
# Digital Video Broadcasting (DVB); Globally Executable MHP (GEM) Specification 1.3 (including OTT and hybrid broadcast/broadband)

**DVB Document A153**

**March 2011**


DVB
Digital Video
Broadcasting

# Contents

# Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for ETSI members and non-members, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

# Foreword

This Technical Specification (TS) has been produced by Joint Technical Committee (JTC) Broadcast of the European Broadcasting Union (EBU), Comité Européen de Normalisation ELECtrotechnique (CENELEC) and the European Telecommunications Standards Institute (ETSI).

NOTE: The EBU/ETSI JTC Broadcast was established in 1990 to co-ordinate the drafting of standards in the specific field of broadcasting and related fields. Since 1995 the JTC Broadcast became a tripartite body by including in the Memorandum of Understanding also CENELEC, which is responsible for the standardization of radio and television receivers. The EBU is a professional association of broadcasting organizations whose work includes the co-ordination of its members' activities in the technical, legal, programme-making and programme-exchange domains. The EBU has active members in about 60 countries in the European broadcasting area; its headquarters is in Geneva.

European Broadcasting Union
CH-1218 GRAND SACONNEX (Geneva)
Switzerland
Tel: +41 22 717 21 11
Fax: +41 22 717 24 81

The Digital Video Broadcasting Project (DVB) is an industry-led consortium of broadcasters, manufacturers, network operators, software developers, regulatory bodies, content owners and others committed to designing global standards for the delivery of digital television and data services. DVB fosters market driven solutions that meet the needs and economic circumstances of broadcast industry stakeholders and consumers. DVB standards cover all aspects of digital television from transmission through interfacing, conditional access and interactivity for digital video, audio and data. The consortium came together in 1993 to provide global standardisation, interoperability and future proof specifications.

# Introduction

## Purpose

The DVB system already provides a comprehensive toolbox to enable interoperable digital video broadcasting systems based on MPEG-2 standards for various transmission media including satellite, cable, terrestrial and microwave. This toolbox also covers interactive services using different kinds of return channels and further supporting functionalities such as service information and many others.

The Globally Executable Multimedia Home Platform (GEM) adds a technical solution for the user terminal that enables the reception and presentation of applications in a vendor, author and broadcaster neutral framework. Here "neutral" includes scenarios that consider legacy infrastructure. Applications from various service providers will be interoperable with different GEM implementations in a horizontal market, where applications, networks, and GEM terminals can be made available by independent providers.

# Aim

The aim of the present document is to encourage implementations of:

- receivers and middleware,
- applications,
- conformance tests.

DVB welcomes feedback from implementers of these. DVB reserves the right to publish a subsequent revision of the present document with (possibly significant) changes based on that feedback.

# 1 Scope

The present document defines the GEM platform. GEM is applicable for specifications and standards based on the GEM APIs, content formats, and semantic guarantees.

The present document is firstly intended to be used by entities writing terminal specifications and/or standards based on GEM. Secondly it is intended for developers of applications that use the GEM functionality and APIs. The GEM specification aims to ensure interoperability between GEM applications and different implementations of platforms supporting GEM applications. This includes interoperability across different middleware specifications, e.g. MHP [1], Blu-ray [13], OCAP [5], ACAP [7], ARIB [6], and the Open IPTV Procedural Application Gateway [8]. Implementers should consult the publisher of specifications which reference GEM regarding conformance.

NOTE: The present document defines the interfaces visible to applications. Application developers should not assume that any related interface is available unless it is specifically listed. Terminal standards or implementations may have other interfaces present.

Since the current business models for delivery of DVB services via broadband IP networks are more operator controlled than was assumed for the original GEM, the present document extends GEM with support for operator supplied applications which run all the time that the GEM environment is running. There is also the possibility that where an operator has subsidised the GEM terminal, applications from that operator may have specially privileges not available to normal applications.

Clauses 1 to 14 specify the applicable technologies and technical definitions in a generic way. Clause 15 provides detailed profile definitions for the following initial profiles:

- Enhanced Broadcasting.
- Interactive Broadcasting.
- Enhanced Packaged Media.
- Interactive Packaged Media.
- IPTV.
- OTT.

which can be extended with future additional profile definitions.

Clause 16 provides a registry of constants and clause 17 describes requirements for internet access clients.

One of the primary goals of the present document is to minimize the number of divergences between GEM and MHP terminal specifications, wherever practical. Divergence is defined in clause 3.1. Where divergences are inescapable, the present document serves as a place to document and control the permitted divergences, so that they will be predictable to terminal manufacturers, broadcasters, and application authors.

# 2 References

References are either specific (identified by date of publication and/or edition number or version number) or non-specific.

- For a specific reference, subsequent revisions do not apply.
- Non-specific reference may be made only to a complete document or a part thereof and only in the following cases:
  - if it is accepted that it will be possible to use all future changes of the referenced document for the purposes of the referring document;
  - for informative references.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

NOTE 1: While any hyperlinks included in this clause were valid at the time of publication ETSI cannot guarantee their long term validity.

NOTE 2: The Java specifications which form the basis of the GEM specification are available at http://www.jcp.org. Additional information about Java TV is available at http://java.sun.com/javame/technology/javatv/index.jsp.

## 2.1 Normative references

The following referenced documents are indispensable for the application of the present document. For dated references, only the edition cited applies. For non-specific references, the latest edition of the referenced document (including any amendments) applies.

[1] ETSI TS 102 727 (V1.1.1): "Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) Specification 1.2.2".

NOTE: Availeble at http://www.etsi.org/deliver/etsi_ts/102700_102799/102727/.

[2] ETSI TS 101 154 (V1.8.1): "Digital Video Broadcasting (DVB); Specification for the use of Video and Audio Coding in Broadcasting Applications based on the MPEG-2 Transport Stream".

[3] JSR 218: "Connected Device Configuration (CDC) 1.1".

NOTE: The latest release of Connected Device Configuration (CDC) is available at http://jcp.org/en/jsr/summary?id=218. At the time of writing, Connected Device Configuration (CDC) 1.1.2 is the latest maintenance release available at. It is strongly recommended to always use the latest release of this JSR.

[4] JSR 217: "Personal Basis Profile (PBP) 1.1".

NOTE: The latest release of Personal Basis Profile (PBP) is available at http://jcp.org/en/jsr/summary?id=217. At the time of writing, Personal Basis Profile (PBP) 1.1.2 is the latest maintenance release. It is strongly recommended to always use the latest release of this JSR.

[5] OCAP 1.1 Profile: "OpenCable Application Platform Specifications".

NOTE: Available at http://www.cablelabs.com/specifications/OC-SP-OCAP1.1.2-090930.pdf.

[6] ARIB STD-B23: "Application Execution Engine Platform for Digital Broadcasting," version 1.1, February 2004.

NOTE: Available at http://www.arib.or.jp/english/html/overview/sb_ej.html.

[7] ATSC A/101A: "Advanced Common Application platform (ACAP)", 12 February 2009

NOTE: Available at http://www.atsc.org/cms/standards/a_101a.pdf

[8] Open IPTV Forum - Release 2 Specification (October 7, 2010): "Volume 6 - Procedural Application Environment, V2.0".

NOTE: Available at http://www.openiptvforum.org/docs/Release2/OIPF-T1-R2-Specification-Volume-6-Procedural-Application-Environment-v2_0-2010-09-07.pdf

[9] ANSI/SCTE 90-1 2004: "SCTE Applications Platform Part 1 OCAP 1.0 Profile".

NOTE: http://www.scte.org/content/index.cfm?pID=1217

[10] Lindholm, T., Yellin, F. (Second Edition): "The Java Virtual Machine Specification".

NOTE: Available at http://java.sun.com/docs/books/jvms/.

[11] Gosling, J., Joy, B., Steele, G. (Third Edition): "The Java Language Specification".

NOTE: Available at http://java.sun.com/docs/books/jls/.

[12] IETF RFC 3073: "Portable Font Resource (PFR) - application/font-tdpfr MIME Sub-type Registration".

NOTE: http://tools.ietf.org/html/rfc3073.

[13] System Description Blu-ray Disc Read-Only Format.

NOTE: http://www.blu-raydisc.info/format_spec.php.

[14] ETSI EN 300 468: "Digital Video Broadcasting (DVB); Specification for Service Information (SI) in DVB systems".

NOTE: Available at http://www.etsi.org/deliver/etsi_en/300400_300499/300468/.

[15] ISO/IEC 10646-1: "Information technology - Universal Multiple-Octet Coded Character Set (UCS)".

[16] JSR 927: "Java TVTM API 1.1".

NOTE: The latest release of JavaTV is always available at http://www.jcp.org/en/jsr/detail?id=927. At the time of writing, JavaTV 1.1.1 is the latest maintenance release with no semantic changes against JavaTV 1.1. It is strongly recommended to always use the latest release of this JSR.

[17] DAVIC 1.4.1 Specification Part 9, Complete DAVIC Specifications, DAVIC. June 1999.

NOTE: Available at http://www.davic.org/Download/Spec1_4_1/141_part09.pdf.

[18] DVB - Digital Video Broadcasting - DVB Identifiers (website).

NOTE: Available at http://www.dvbservices.com/identifiers/index.php.

[19] ETSI ES 200 800: "Digital Video Broadcasting (DVB); Interaction channel for Cable TV distribution systems (CATV)".

[20] ETSI ETS 300 801: "Digital Video Broadcasting (DVB); Interaction channel through Public Switched Telecommunications Network (PSTN)/ Integrated Services Digital Networks (ISDN)".

[21] ISO/IEC 11172-2: "Information technology -- Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s -- Part 2: Video".

[22] ISO 6937: "Information technology -- Coded graphic character set for text communication -- Latin alphabet".

[23] ISO 8859 (all parts): "Information technology -- 8-bit single-byte coded graphic character sets".

[24] CIE 15-2004: Colorimetry, 3rd Edition - ISBN 978 3 901906 33 6.

[25] ISO 3166-1: "Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes".

[26] IETF RFC 822: "Standard for the format of ARPA internet text messages".

[27] Hunt, R.W.G. 1987 "Measuring Colour (Ellis Horwood Series in Applied Science and Industrial Technology)". Ellis Harwood Limited, Chichester, England.

[28] W3C Recommendation 16 August 2006: "Namespaces in XML 1.0 (Second Edition)".

NOTE: Available at http://www.w3.org/TR/REC-xml-names.

[29] ETSI TS 101 162: "Digital Video Broadcasting (DVB); Allocation of Service Information (SI) and Data Broadcasting Codes for Digital Video Broadcasting (DVB) systems".

[30] ETSI TS 101 225 (V1.1.1): "Digital Video Broadcasting (DVB); Home Local Network Specification based on IEEE 1394".

[31] CORBA/IIOP: "The Common Object Request Broker: Architecture and Specification", Object Management Group.

NOTE: Available at http://www.omg.org/cgi-bin/doc?formal/04-03-12.pdf.

[32] ETSI EN 301 192: "Digital Video Broadcasting (DVB); DVB specification for data broadcasting".

[33] GIF 89a: "Graphics Interchange Format (sm)", version 89a.

NOTE: Available at http://www.w3.org/Graphics/GIF/spec-gif89a.txt.

[34] ISO/IEC 10918-1: "Information technology - Digital compression and coding of continuous-tone still images: Requirements and guidelines".

NOTE: Available at http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=18902.

[35] ISO/IEC 11172-3 (1993): "Information technology - Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s - Part 3: Audio".

NOTE: Known as "MPEG-1 Audio".

[36] ISO/IEC 13818-1: "Information technology - Generic coding of moving pictures and associated audio information: Systems".

[37] ISO/IEC 13818-2: "Information technology - Generic coding of moving pictures and associated audio information: Video".

[38] ISO/IEC 13818-6 (1998): "Information technology - Generic coding of moving pictures and associated audio information - Part 6: Extensions for DSM-CC".

[39] IEC 61966-2-1 (1999): "Multimedia systems and equipment - Colour measurement and management - Part 2-1: Colour management - Default RGB colour space - sRGB".

[40] ITU-R Recommendation BT.470-6: "Conventional television systems".

[41] ITU-R Recommendation BT.709-5: "Parameter values for the HDTV standards for production and international programme exchange".

[42] JFIF: "JPEG File Interchange Format". Eric Hamilton, C-Cube Microsystems.

NOTE: Available at http://www.w3.org/Graphics/JPEG/jfif3.pdf.

[43] W3C Recommendation 1 October 1996, Version 1.0: "PNG (Portable Network Graphics) Specification".

NOTE: Available at http://www.w3.org/TR/REC-png-961001.html.

[44] IETF RFC 1321 (1992): "The MD5 Message-Digest Algorithm".

[45] IETF RFC 2616 (1999): "Hypertext Transfer Protocol -- HTTP/1.1".

[46] IETF RFC 2396 (1998): "Uniform Resource Identifiers (URI): Generic Syntax".

[47] IETF RFC 768 (1980): "User Datagram Protocol".

[48] IETF RFC 791 (1981): "Internet protocol".

[49] IETF RFC 793 (1981): "Transmission control protocol".

[50] HAVi, v1.1, This comprises the following documents: HAVi v1.1 Chapter 8, 15-May-2001 HAVi v1.1 Java L2 APIs, 15-May-2001 HAVi v1.1 Chapter 7,15-May-2001.

NOTE: Available at http://www.havi.org.

[51] ITU-T Recommendation X.501 (1993): "Information Technology - Open Systems Interconnection - The Directory: Models".

[52] ITU-T Recommendation X.509 (1997): "Information technology - Open Systems Interconnection - The Directory: Authentication framework".

[53] IETF RFC 2313 (1998): "PKCS #1: RSA Encryption Version 1.5".

[54] ITU-T Recommendation X.680 (1994): "Information Technology - Abstract Syntax Notation One (ASN.1): Specification Of Basic Notation".

[55] IETF RFC 2459 (1999): "Internet X.509 Public Key Infrastructure Certificate and CRL Profile".

[56] ETSI ETS 300 706 (1997): "Enhanced Teletext Specification".

[57] FIPS PUB 180-1 (1995): "Secure Hash Standard".

NOTE: Available at http://www.itl.nist.gov/fipspubs/fip180-1.htm.

[58] IETF RFC 2246 (1999): "The TLS Protocol Version 1.0".

[59] IETF RFC 2045 (1996): "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies".

[60] W3C Recommendation 6 October 2000: "Extensible Markup Language (XML) 1.0 (Second Edition)".

NOTE: Available at http://www.w3.org/TR/2000/REC-xml-20001006.

[61] IETF RFC 1738 (1994): "Uniform Resource Locators (URL)".

[62] IETF RFC 1035 (1987): "Domain names - implementation and specification".

[63] Tiresias: "The RNIB/DTG "Tiresias" font version 8.03".

NOTE: For information on obtaining this font, please contact the DVB project office.

[64] IETF RFC 1034 (1987): "Domain names - concepts and facilities".

[65] IETF RFC 1982 (1996): "Serial Number Arithmetic".

[66] IETF RFC 2181 (1997): "Clarifications to the DNS Specification".

[67] ATSC A/100-5 (2003): " DTV Application Software Environment Level 1 (DASE-1) - Part 5: ZIP Archive Resource Format".

NOTE: Available at http://www.atsc.org/standards/a100/a_100_5.pdf.

[68] IETF RFC 1945 (1996): "Hypertext Transfer Protocol -- HTTP/1.0".

[69] W3C Recommendation 24 December 1999: "HTML 4.01 Specification".

NOTE: Available at http://www.w3.org/TR/html401.

[70] JSR 177 - SATSA, v1.0: "Security and Trust Services API (SATSA) for Java 2 Platform, Micro Edition".

NOTE: Available at http://jcp.org/en/jsr/detail?id=177.

[71] IETF RFC 821 (1982): "Simple Mail Transfer Protocol".

[72] IETF RFC 977 (1982): "Network News Transfer Protocol".

NOTE: Available at http://www.w3.org/Protocols/rfc977/rfc977.

[73] IETF RFC 2818 (2000): "HTTP over TLS".

[74] IETF RFC 2368 (1998): "The mailto URL scheme".

[75] ISO/IEC 14496-18 (2004): "Information technology -- Coding of audio-visual objects -- Part 18: Font compression and streaming".

[76] JSR 172, v1.0: "J2ME Web Services Specification".

NOTE: Available at http://jcp.org/aboutJava/communityprocess/final/jsr172/index.html.

[77] JSR 216: "Personal Profile 1.1".

NOTE: Available at http://jcp.org/en/jsr/detail?id=216.

[78] JSR 219: FP 1.1: "Foundation Profile 1.1 for Java 2 Platform, Micro Edition".

NOTE: Available at http://jcp.org/en/jsr/detail?id=219.

[79] IETF RFC 3268 (2002): "Advanced Encryption Standard (AES) Ciphersuites for Transport Layer Security (TLS)".

[80] ETSI TS 102 034: "Digital Video Broadcasting (DVB); Transport of MPEG-2 Based DVB Services over IP Based Networks".

[81] ETSI TS 102 539: "Digital Video Broadcasting (DVB); Carriage of BCG information over IP".

[82] ETSI TS 102 816: "Digital Video Broadcasting (DVB); PVR/PDR Extension to the Multimedia Home Platform".

[83] IETF RFC 2326: "Real Time Streaming Protocol (RTSP)".

[84] ISO 639 (all parts): "Codes for the representation of names of languages".

[85] ETSI, TS 102 817, “Digital Video Broadcasting (DVB);Digital Recording Extension to Globally Executable Multimedia Home Platform (GEM)” V1.1.1

[86] ETSI, TS 102 817, “Digital Video Broadcasting (DVB);Digital Recording Extension to Globally Executable Multimedia Home Platform (GEM)” V1.1.1

## 2.2 Informative references

The following referenced documents are not essential to the use of the present document but they assist the user with regard to a particular subject area. For non-specific references, the latest version of the referenced document (including any amendments) applies.

[i.1] Application Definition Blu-ray Disc Format - BD-J Baseline Application and Logical Model Definition for BD-ROM, March 2005.

NOTE: http://www.blu-raydisc.com/ under “Technical Info”, “Public Specifications”

[i.2] ETSI EN 301 193 (V1.1.1): "Digital Video Broadcasting (DVB); Interaction channel through the Digital Enhanced Cordless Telecommunications (DECT)".

[i.3] ETSI EN 301 195 (V1.1.1): "Digital Video Broadcasting (DVB); Interaction channel through the Global System for Mobile communications (GSM)".

[i.4] ETSI EN 301 199 (V1.2.1): "Digital Video Broadcasting (DVB); Interaction channel for Local Multi-point Distribution Systems (LMDS)".

[i.5] ETSI TS 101 211 (V1.9.1): "Digital Video Broadcasting (DVB); Guidelines on implementation and usage of Service Information (SI)".

[i.6] ETSI EN 300 472 (V1.2.2): "Digital Video Broadcasting (DVB); Specification for conveying ITU-R System B Teletext in DVB bitstreams".

[i.7] ETSI EN 300 743 (V1.2.1): "Digital Video Broadcasting (DVB); Subtitling systems".

[i.8] ETSI TR 101 201 (V1.1.1): "Digital Video Broadcasting (DVB); Interaction channel for Satellite Master Antenna TV (SMATV) distribution systems; Guidelines for versions based on satellite and coaxial sections".

[i.9] ETSI TR 101 202 (V1.1.1): "Digital Video Broadcasting (DVB); Implementation guidelines for Data Broadcasting".

[i.10] ETSI SR 001 262 (V1.7.1): "ETSI drafting rules".

[i.11] ETSI EN 301 790 (V1.2.2): "Digital Video Broadcasting (DVB); Interaction channel for satellite distribution systems".

[i.12] IETF RFC 1877 (1995): "PPP Internet Protocol Control Protocol Extensions for Name Server Addresses".

[i.13] IETF RFC 1332 (1992): "The PPP Internet Protocol Control Protocol (IPCP)".

[i.14] IETF RFC 1661 (1994): "The Point-to-Point Protocol (PPP)".

[i.15] IETF RFC 1717 (1994): "The PPP Multilink Protocol (MP)".

[i.16] CENELEC EN 50049-1: "Domestic and similar electronic equipment interconnection requirements: Peritelevision connector".

# 3 Definitions and abbreviations

## 3.1 Definitions

For the purposes of the present document, the terms and definitions given in TS 102 034 [80] and the following terms and definitions apply:

Application Program Interface (API): interface between an application and a particular feature, function or resource of GEM

application: functional implementation realized as software running in one or spread over several interplaying hardware entities

application boundary: concise general description of the data elements (HTML documents, code files, images, etc.) used to form one application and the logical locator of the entry point, the application boundary is described by a regular expression over the URL language

NOTE: Where no such boundary is drawn, the default boundary is the entire set of documents that the MHP platform can access.

application instance: unique invocation of an application, i.e. running the same application twice results in two distinct application instances

application manager: entity in GEM that is responsible for managing the lifecycle of the applications in the GEM terminal

NOTE: The application manager manages both the applications and non-applications.

autostart applications: this term has different definitions depending on the application format:

- A DVB-J autostart application is an application that is automatically loaded and executed by the Application Manager as soon as the user selects a service on which the application is signalled as autostart.
- Auto start application a application in a broadcast stream can be signalled as auto start in the same way that other DVB applications can.

NOTE: That it may not actually start providing service until it receives a start trigger.

best effort: implementation dependent approximation which is as close as reasonable in the circumstances concerned to what has been requested

character: specific "letter" or other identifiable symbol, e.g. "A"

character encoding: mapping between an integer input value, and the textual character that is represented by this mapping, e.g. in ASCII value 65 (decimal) is character "A", or shift-JIS for Japanese characters

character set: See character encoding.

communications network: system of interconnected entities providing data interchange between points or from a point to multiple points

Competitive API: API defined in a GEM terminal specification that is used to access functionality (e.g. signalling) that could be reasonably mapped to a GEM API

NOTE: A competitive API would be used to access functionality that is functionally equivalent to a GEM requirement. Competitive APIs are forbidden, unless the functionality is also exposed via the GEM API. See clause 4.1.4.2, "Competitive APIs".

Complementary functional equivalent: functionality (e.g. signalling) in a GEM terminal specification that is functionally equivalent to a GEM functional equivalent, but where another mechanism already exists that satisfies the GEM requirement that a functional equivalent be defined

NOTE: In this case, the two functionally equivalent mechanisms are called complementary functional equivalents

definite: resource is considered of indefinite duration if the exact end point is not known in advance of accessing the resource (e.g. Transport Streams in MPEG-2)

NOTE: A resource is considered of definite duration if in principle the exact length is known (e.g. a file resource).

delegated application: application encoded in a representation that is not directly consumable by an MHP terminal's standard mechanisms

divergence: everything that violates an assertion in a specification and/or a conformance clause

NOTE: A divergence from the GEM specification is when a correctly written conformance test for a GEM specification assertion would fail.

domain of an application: domain of an Xlet characterizes the "space" within which the Xlet is able to execute

NOTE 1: This includes both the "connection" where the Xlet is delivered and other "connections" where an already executing Xlet is allowed to continue executing.

NOTE 2: An application cannot run outside its domain. The maximum lifetime of an application extends from the moment the user navigates to its domain until the moment that the user navigates away from its domain.

NOTE 3: In the broadcast case a "connection" corresponds to a DVB-service. Broadcast signalling indicates which services can load an application and which services allow an already active application to continue.

DVB network: collection of MPEG-2 Transport Stream multiplexes transmitted on a single delivery system, e.g. all digital channels on a specific cable system

DVB-J: Java platform defined as part of the MHP specification

DVB-J API: one of the Java APIs standardized as part of the MHP specification

DVB-J application: set of DVB-J classes that operate together and need to be signalled as a single instance to the Application Manager so that it is aware of its existence and can control its lifetime through a lifecycle interface

enumeration: type that includes in its definition an exhaustive list of possible values for variables of that type

NOTE: In Java, enumerations are not directly supported, but they are often simulated with a set of integer constants.

events: asynchronous communication between applications and the MHP on which they are being executed, they provide communication between solution elements

font: mechanism that allows the specific rendering of a particular character to be specified - e.g. Tiresias, 12 point

NOTE: In practice a font file format will incorporate some aspects of a character encoding.

function: process which conveys or transforms data in a predictable way, it may be effected by hardware, software of a combination of the two

functionally equivalent: functionally equivalent requirement is one that specifies behaviour that performs substantially the same function with substantially the same behaviour as the original specification, as seen from an application's point of view

NOTE: There are several clauses within GEM that do not require literal conformance with the corresponding requirement in the underlying specification, but allow for a compatible substitution.

GEM application: application that is written only to the interfaces and semantic guarantees defined in GEM

NOTE: A suitably signalled GEM application will run on an MHP terminal, or on any terminal that complies to a GEM terminal specification, e.g. on OCAP and the ARIB AE.

GEM terminal: terminal or other device that conforms to a GEM Terminal Specification

NOTE: Examples of GEM terminals include an MHP terminal, an OCAP terminal (including the POD) and a terminal supporting the ARIB AE.

GEM Terminal Specification: specification that includes all normative and selected optional elements of its underlying GEM document, and provides additional specifications that describe functionally equivalent elements for each and every clause of the underlying GEM document where required

hardware entity: independent piece of hardware which forms part of a (multiple) local cluster of elements which as a whole is called a terminal

NOTE: A hardware entity is for example: a set top box, a digital VCR or a conditional access module. A hardware entity includes a number of resources. Each resource provides a number of functions.

hybrid: A combination of more than one GEM target in a terminal.

NOTE: Examples of hybrid GEM terminals include a hybrid Broadcast/Broadband terminal, a hybrid Broadcast/Internet terminal, other combinations are also possible.

indefinite: See definite.

interoperable plug-in: plug-in that only requires standard GEM APIs

interoperability: reception and presentation of applications in a vendor, author and broadcaster neutral framework

java API: standard interface for use by platform independent application software expressed in the Java language

java TV: APIs defined in the packages under the javax.tv. namespace in the present document

lifetime of an application: characterizes the time from which the application is Loaded to the time the application is Destroyed

locator: opaque reference to the location information of objects which are addressable within the Java TV API

NOTE: A given locator may represent a transport independent object and have multiple mappings to transport dependent locators.

Multimedia Home Platform (MHP): GEM-based terminal specification

native signalling: signalling infrastructure for a delegated application that are not part of a GEM terminal's standard mechanisms

navigator: resident application, typically provided by the manufacturer, which the end-user can activate at any time

NOTE: The navigator can be used to select services, applications, and initiate Interoperable applications.

object carousel: repetitively broadcast file system

OID: X.509 Object Identifier

OTT: OTT is a general term for video services delivered over the Open Internet. It's referred to as "over-the-top" because these services ride on top of plain Internet access service and don't require any business or technology affiliations with the network operator.

persistent storage: non-volatile memory available in the GEM terminal which can be read/written to by an application and which may outlive the application's own life

plug-in: set of functionality which can be added to a generic platform in order to provide interpretation of DVB registered, but non-DVB-J, application formats; e.g. HTML3.2 or MHEG-5

privileged application: application from a broadcaster, operator or service provider with permissions not granted to normal signed GEM applications

NOTE: This is the same concept as applications with MonitorAppPermission in OCAP [5].

profile: description of a series of minimum configurations, defined as part of the present document, providing different capabilities of the GEM terminal

NOTE: A profile maps a set of functions which characterize the scope of service options. The number of profiles is small. The mapping of functions into resources and subsequently into hardware entities is out of the scope of the present document and is left to manufacturers.

PSI-only service: MPEG program listed in the PAT of a transport stream but not listed in the SDT or EIT

regular expression: method of capturing a large, possibly infinite set of strings in a compact representation

resident application: application available from non-volatile storage in a GEM device which may be expressed in DVB-J but need not be so

NOTE: Its delivery route is not specified.

resource: is a well defined capability or asset of a system entity, which can be used to contribute to the realization of a service

NOTE: Examples of resources include: MPEG decoder, Graphics system.

return channel: communications mechanism which provides connection between GEM and a remote server

running application: application is considered running if it is in any state other than NOT_LOADED or INVALID

runtime code extension: ability to extend, at runtime, the executable code of an application with code not originally packaged with the application

NOTE: This facility is provided, for example, by org.dvb.lang.DVBClassLoader in DVB-J and by eval() in ECMAscript.

sandbox: unsigned applications and signed applications without a permission file have access to all the APIs for which there is no permission signalling defined, this is commonly called the sandbox

service: sequence of programs under the control of a broadcaster which can be broadcast as part of a schedule

service component: part of a service

NOTE: For a DVB service, service components are normally the MPEG elementary streams listed in the PMT for the service. Where multiple streams of subtitles or MPEG-2 audio representing different languages are carried in the same MPEG elementary stream these are logically service components but are not exposed through the GEM APIs as being distinct and separate.

stand-alone application: application which is not related to a conventional DVB service

NOTE: In this version of the present document, this includes applications running as part of a stored service and applications whose signalling was carried over the interaction channel.

standard definition: MPEG-2 main level at main profile, as defined in TS 101 154 [2]

stream: unidirectional continuous flow of content, for example, MPEG2 video

stored service: set of GEM applications stored within the GEM terminal and treated as a service with regard to life cycle, service selection, etc.

subsidizing service provider: operator who is subsidizing all or part of the cost of a GEM terminal and hence expects special privileges for their applications

system software: software implementation below the API for a specific platform entirely under control of the manufacturer

target: category of GEM terminal specification(s), determined by the mechanism used to transport applications from the content producer to the viewer

EXAMPLE: The broadcast target is for use by GEM terminal specifications in broadcast environments; the packaged media target is used for GEM terminal specifications where the media is packaged onto a physical carrier which is possibly read-only, such as an optical disc.

transport dependent: transport dependent Service or Locator refers to a service or content delivered by a particular delivery mechanism or channel

EXAMPLE: BBC1 delivered by DVB-T would have a different transport dependent Service and Locator from BBC1 delivered by IPTV.

transport independent: transport independent Service or Locator can refer to copies of that service or content regardless of how they are delivered to the GEM terminal

EXAMPLE: The same transport independent Service and Locator could apply to both BBC1 delivered by DVB-T and BBC1 delivered by IPTV.

trigger: event that may cause a change in the behaviour of an application that registers interest in such events

NOTE: Triggers come from the broadcast stream. The trigger may include a reference to time relative to the NPT of a media stream or be asynchronous. It also can carry some semantically significant payload in order to affect changes in an application based on information not available at the time an application was written.

tuning: act of switching between two MPEG transport streams or multiplexes

NOTE: Switching between two DVB services carried in the same transport stream is not tuning.

viewer: end-user of the GEM terminal

xlet: interface used for DVB-J application life cycle control

The following definitions from clause 3.2.1 "Definitions" of OCAP [5] shall apply:

- Abstract service.
- Unbound application.

## 3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

| | |
|---|---|
| AIT | Application Information Table |
| API | Application Programming Interface |
| AV | Audio Video |
| AWT | Abstract Windowing Toolkit |
| BCG | Broadband Content Guide |
| CA | Conditional Access |
| CI | Common Interface |
| CLUT | Colour LookUp Table |
| CoD | Content on Demand |
| CRID | Content Reference Identifier |
| CSS | Cascading Style Sheets |
| DAVIC | Digital Audio Visual Council |
| DCT | Discrete Cosine Transformation |
| DECT | Digital Enhanced Cordless Telecommunications |
| DN | Distinguished Name |
| DOM | Document Object Model |
| DSM-CC | Digital Storage Media - Command and Control |
| DVB | Digital Video Broadcasting |
| DVB-J | DVB Java |
| EPG | Electronic Program Guide |
| ETSI | European Telecommunications Standards Institute |
| GEM | Globally Executable MHP |
| GIF | Graphics Interchange Format |
| GSM | Global System for Mobile communications |
| GUI | Graphical User Interface |
| HTML | Hyper Text Mark-up Language |
| HTTP | Hyper Text Transport Protocol |
| I/O | Input / Output |
| ICC | International Colour Consortium |
| ID | IDentifier |
| IHDN | In Home Digital Network |
| IP | Internet Protocol |
| IPR | Intellectual Property Rights |

| | |
|---|---|
| IPTV | IP television |
| ISDN | Integrated Services Digital Network |
| ISO | International Organization for Standardization |
| ITU | International Telecommunication Union |
| JDK | Java Development Kit |
| JFIF | JPEG File Interchange Format |
| JMF | Java Media Framework |
| JPEG | Joint Picture Expert Group |
| LMB | Live Media Broadcast |
| LMDS | Local Multipoint Distribution System |
| MHEG | Multimedia Hypermedia Expert Group |
| MHP | Multimedia Home Platform |
| MPEG | Moving Picture Expert Group |
| MSO | Multiple System Operator |
| NPT | Normal Play Time |
| OC | Object Carousel |
| OCAP | Open Cable Application Platform |
| OOB | Out of Band |
| OS | Operating System |
| OSD | On Screen Display |
| OTT | Over the Top TV |
| PFR | Portable Font Resource |
| PMT | Program Map Table |
| PNG | Portable Network Graphics |
| POD | Point of Deployment |
| PSI | Program Specific Information |
| PSTN | Public Switched Telephone Network |
| PVR | Personal Video Recorder |
| RAM | Random Access Memory |
| RGB | Red, Green Blue |
| SCART | The connector known as "Syndicat des Constructeurs d'Appareils Radiorecepteurs et Televiseurs" also Peritel and Euroconnector and Cenelec EN 50049-1 [i.16]. |
| SD&S | Service Discovery and Selection |
| SI | Service Information |
| SMATV | Satellite Master-Antenna Television |
| TCP | Transmission Control Protocol |
| TS | Transport Stream |
| UCS | Universal Multiple-Octet Coded Character Set |
| UDP | User Datagram Protocol |
| UI | User Interface |
| URL | Uniform Resource Locator |
| UTF | UCS Transformation Coding |
| UTF8 | Universal Transformation Format 8 |
| UU | User to User |
| VM | Virtual Machine |
| WSS | Wide Screen Signalling |
| XML | eXtensible Markup Language |

# 4 General considerations and conventions

## 4.1 General considerations

### 4.1.1 Purpose

The GEM document is not intended, and should not be used, as a complete terminal specification. It is a framework upon which a GEM terminal specification can be created.

A GEM-based specification, the Multimedia Home Platform (MHP) middleware standard, defines a comprehensive platform that enables interactive television services to be deployed that are interoperable across any manufacturer's implementations of the standard. MHP is a comprehensive specification of a receiving device (an MHP terminal). MHP terminals receive digital video broadcasting services based on standards for various transmission media including satellite, cable, terrestrial, microwave and TCP/IP. The transport layer may be DVB-T/T2, DVB-C/C2, DVB-S/S2 or an IP transport.

One element of the MHP standard is a description of the terminal facilities that can be exploited by applications that form a part of a broadcast service. These facilities may be exposed via APIs (Application Programming Interfaces); such APIs carry semantic guarantees. Similarly, receiver functionality can be exposed with a declarative content format that contains semantic guarantees. Another element of the MHP standard is the specification of the terminal hardware and signalling infrastructure that allows it to be connected to any compatible network.

In some regions, markets and/or networks, it is impractical to adopt the full MHP specification. For example, in the United States, there is a significant investment in infrastructure that cannot be easily converted. In Japan, the terrestrial broadcasting standard, while very similar to DVB-T, is not the same, and contains elements that make the adoption of the full MHP standard for terminals impractical.

Despite these regional differences, it is desirable to be able to execute a GEM application as part of a service that is carried over different network infrastructure. Such interoperability can be achieved, as long as the middleware standard supports the same APIs and semantic guarantees.

The present document for the Global Execution of MHP services (GEM) defines the APIs, semantic guarantees, and content formats that can be relied upon in all interactive television standards and specifications that support globally-interoperable GEM applications. Any such specification based on GEM shall normatively reference the GEM specification in its entirety, and shall fulfil the normative requirements of GEM.

The present document does not provide a complete specification sufficient to implement a device. Additional normative elements are required.

NOTE: The present document covers the broadcast, packaged media, IPTV and OTT targets and defines the common core which is mandatory. There are a few APIs that are not required when only the broadcast target is implemented. Similarly, there are a few APIs that are not required when only the IPTV, OTT or packaged media target is implemented. However, when one or both targets are implemented, it is mandatory that all the required APIs and other definitions are included as specified in this recommendation.

### 4.1.2 Format

The present document does not state how the receiver has to be built or what network infrastructure has to underlie the implementation; it is limited to specifying the behaviour and interfaces that globally interoperable applications may rely on.

### 4.1.3 Inclusion of GEM features

#### 4.1.3.1 Subsetting prohibited

Specifications that reference GEM shall include it in its entirety. It is prohibited to base any specification on GEM if the referencing document does not require all normative requirements of the present document.

#### 4.1.3.2 Supersetting permitted

If a GEM terminal specification wishes to include APIs, signalling or behaviours defined in MHP [1] that are not required by GEM, it may do so as described in clause 15.6, "Functional equivalents".

### 4.1.4 Addition of non-GEM interfaces

Specifications based on GEM may add public interfaces, provided that they are added in a namespace that does not conflict with GEM. For example, OCAP [5] defines extensions in the Java package `org.ocap`.

GEM terminal specifications and GEM terminals shall not require that such extension interfaces be called by GEM applications in order to enable behaviour that is normatively required by the present document.

#### 4.1.4.1 DVB-J enumerations

A GEM terminal specification shall not add new values to an enumeration that is returned from a method defined by the present document.

NOTE: For example, the interface `org.dvb.net.rc.RCInterface` defined in annex R introduces an enumeration that is returned by the method `getType()`. This enumeration includes the values `TYPE_CATV`, `TYPE_DECT`, etc. It is not permissible to attempt to subdivide one of these types by introducing new enumeration values in a different namespace. See also the example in clause W.2, "Example of enumeration extension".

#### 4.1.4.2 Competitive APIs

A GEM terminal specification may include functionality that is functionally equivalent to functionality required by GEM, and thus could reasonably be mapped to a GEM API. This functionality (e.g. signalling) might be in addition to functionality that satisfies the GEM requirement that a functional equivalent be defined. In other words, the two mechanisms may be complementary functional equivalents.

In this case, the GEM terminal specification shall define a mapping to the GEM API for both mechanisms, that is, for both functional equivalents. All such mappings shall comply with the GEM requirements for the functional equivalents.

As with all functional equivalents, complementary functional equivalents may include features beyond what is required by GEM, and that can not reasonably be exposed via a GEM API. In this case, it is of course allowable for the GEM terminal specification to define additional APIs to expose these features, in a non-GEM package namespace.

NOTE: Consider, for example, the "Carousel" functional equivalent. A GEM terminal specification might provide two different signalling schemes for delivering the files and associated events for a GEM application. If this is done, GEM-compliant bindings have to be defined for the APIs and other requirements of GEM for both signalling schemes. This is illustrated in the following two figures.

##### 4.1.4.2.1 Illustration of Complementary Functional equivalents

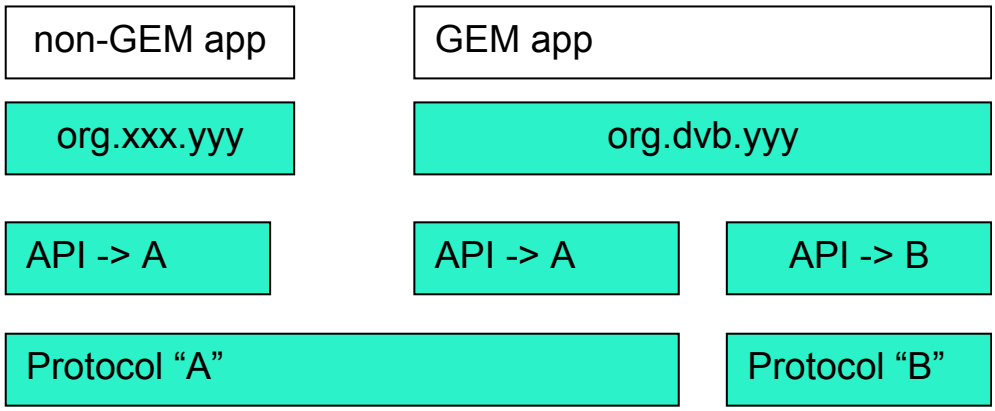



**Figure 1: Complementary Functional equivalents**

Figure 1, showing complementary functional equivalents A and B, both mapped to the GEM API org.dvb.yyy. Mechanism A includes additional features, which are exposed to non-GEM applications via the API org.xxx.yyy. This situation is normal and does not conflict with GEM clause 4.1.4.2.

#### 4.1.4.2.2 Illustration of Competitive APIs

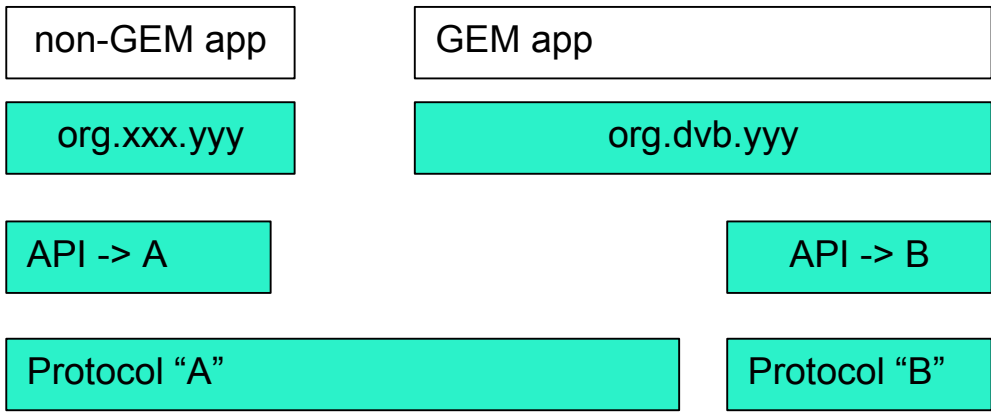



**Figure 2: Competitive APIs**

Figure 2, showing complementary functional equivalents A and B. Only mechanism B is mapped to the GEM API; A is only exposed via the non-GEM API org.xxx.yyy. In this case, org.xxx.yyy is a competitive API, and is in conflict with clause 4.1.4.2, "Competitive APIs".

## 4.1.5 Application areas

GEM defines four targets: Broadcast, Packaged Media and IPTV and OTT. The first two of these, Broadcast and Packaged Media, are further subdivided into two application areas: Enhanced Broadcasting and Interactive Broadcasting. Enhanced Broadcasting combines digital broadcast of audio/video services with downloaded applications which can enable local interactivity; it does not need an interaction channel. Interactive Broadcasting enables a range of interactive services associated or independent from broadcast services; it requires an interaction channel.

A target is a category of GEM terminal specification(s), determined by the mechanism used to transport applications from the content producer to the viewer.

Currently GEM defines four targets:

- The broadcast target, which is used by GEM terminal specifications in a broadcast environment, such as MHP [1], OCAP [5], ATSC [7] and ARIB [6].
- The packaged media target where the media is packaged on a physical medium which is possibly read-only, such as an optical disc as Blu-ray [13].
- The IPTV target, where media is transmitted over a bidirectional broadband connection, such as in MHP [1] and in the OpenIPTV Forum [8].
- The OTT target, where media is transmitted over a bidirectional broadband connection without QoS guarantee.

Combinations of the GEM targets into a combined target is explicitly permitted, following the GEM rules for combined profiles as defined below:

- If a feature is present in one of the targets, it shall be present in the combined target.
- The combined target is the unification of all features of the individual target.
- If a feature is optional or not required in the combined target, a valid implementation may omit it.
- If a feature is optional in one target and mandatory in the other, it shall be mandatory in the combined target.

#### 4.1.5.1 Broadcast target

The broadcast target is available since GEM 1.0.x and is defined for broadcast TV using cable, terrestrial or satellite.

#### 4.1.5.2 Packaged Media target

The packaged media target allows to combine audio/video content and application into a self contained medium, such as an optical disc. The GEM packaged media target is the basis of the BD-J Blu-ray execution environment.

#### 4.1.5.3 IPTV target

The GEM-IPTV target is defined to enable audio/video services over managed broadband networks. These networks allow management of the broadband bandwith and can maintain a guaranteed QoS.

Since these networks can provide both unicast and multicast network services, typical services can include LMB (based on multicast) and unicast Content on Demand (CoD) services.

#### 4.1.5.4 OTT target

According to the service profiles defined in [80] for delivery of media contents over IP networks, OTT is restricted to Content on Demand (CoD) type of services and LMB over unicast connections.

NOTE: Multicast based LMB is typically not possible within OTT, even if live (unicast) contents can be enjoyed via OTT.

Content on Demand (CoD) service is a service where users can select the individual content items they want to watch from the list of available content. Consumption of the content is started upon user request.

2 types of CoD services are addressed within OTT target:

- Streamed CoD services, where content is consumed while the content itself is being delivered (real-time streaming)
- Download CoD services, where the whole content has to be downloaded first to the local storage in the receiver before consuming it. Consumption is then independent of the delivery.

#### 4.1.5.5. Typical Hybrid profiles

The following combined targets are most useful for certain markts:

- OTT+Broadcast: a combiation of OTT with the Broadcast Interactive Profile used in hybrid broadcast/broadband receivers
- OTT+Packaged: a combination of OTT with the Packaged Media Interactive Profile
- OTT+IPTV: a combined IP profile for use in managed and unmanaged networks
- OTT+Broadcast+Packaged: a combination of OTT with the Broadcast Interactive Profile with Packaged Media Interactive Profile, in all-in-one hybrid broadcast/broadband receivers/players

### 4.1.6 Profiles

As not all GEM implementations will be able to support all application areas and as there is a further evolution expected over time, different profiles of GEM are considered. For this release of the GEM specification, profiles are mapped to the above mentioned application areas, yielding five profiles.

Figure 3 shows the six GEM profiles. These are described in greater detail in clause 15, "Detailed platform profile definitions".

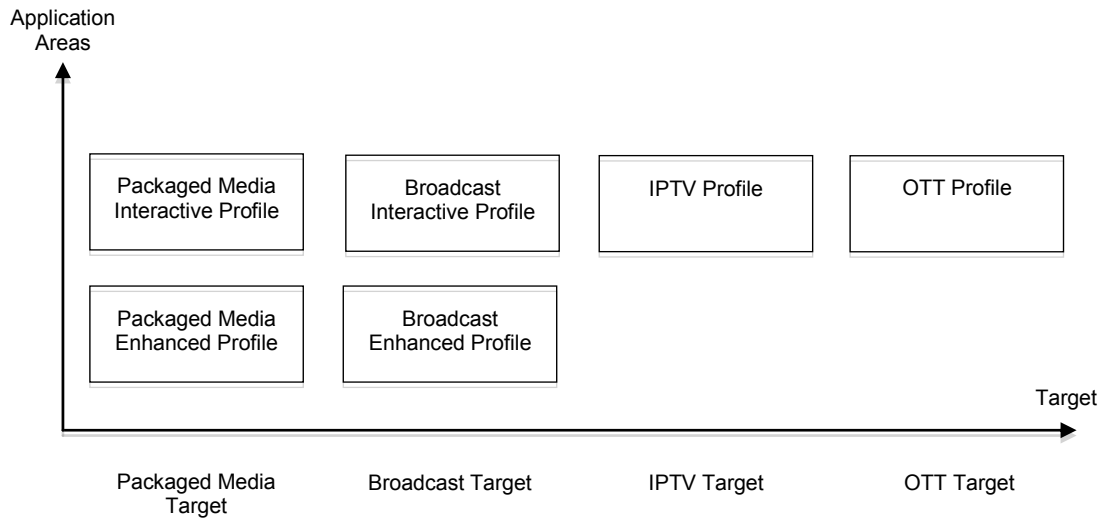



**Figure 3: GEM platform profiles**

### 4.1.7 Full conformance with the present document

To be fully conformant with the present document, GEM terminals shall conform to at least one of the profiles defined in clause 15, "Detailed platform profile definitions".

Further, GEM terminals shall be conformant with any one of the following specifications:

- MHP 1.0., MHP 1.1, MHP 1.2, MHP 1.2.2. [1] or later.

- OCAP 1.0, OCAP 1.1 [5] or later.
- ARIB STD-B23[6]: "Application Execution Engine Platform for Digital Broadcasting," version 1.1, February 2004 or later.
- ATSC A/101[7]: "Advanced Common Application platform (ACAP)" or later.
- SCTE 90-1 [9].
- Blu-Ray Disc Player Specification [13].
- PAE 1.0 specification of the Open IPTV forum [8] or later.
- Any other specification that implements the IPTV target, for the purposes of presenting GEM services delivered via an IP transport. It is permissible if such an IPTV specification is not publicly available

  NOTE 1: All realizations of GEM functional equivalents and other information about the Blu-Ray Disc Player Specification, of particular interest to application authors, is available in the BD-J Baseline Application and Logical Model Definition for BD-ROM [i.1].

- Any other specification that implements the IPTV target, for the purposes of presenting GEM services delivered via an IP transport. It is permissible if such an IPTV specification is not publicly available.

For avoidance of doubt, equipment which is fully conformant with the entire present document apart from the above clause is not fully conformant with the present document.

NOTE 1: It is expected that a future version of OCAP will adopt GEM 1.2.x.

GEM terminal specifications shall contain a normative requirement that its terminals conform with GEM. GEM terminal specifications shall further require that, in the event of a conflict between GEM and the GEM terminal specification, GEM shall take precedence. Any errata to GEM shall be in a designated section of GEM errata, which shall only contain errata agreed for publication in a subsequent version of GEM. The text of the errata may be in the GEM terminal specification, but there shall be a statement referring to a document under the control of the DVB that confirms that these are agreed errata.

NOTE 2: As an example, this can be implemented with language like the following, drafted for a fictional GEM terminal specification called "A GEM Terminal Specification" (AGTS).

"7.1 Compliance with GEM

AGTS terminals shall comply in full with GEM [ref GEM]. The present document adopts the MHP definition of the following functional equivalents, as specified in GEM [ref GEM], clause 15.6:

- Arch
- Carousel
- Text Wrapping

For avoidance of doubt, in the event of a conflict between GEM [ref GEM] and this specification, the normative guarantees of GEM [ref GEM] shall take precedence except as detailed in clause 7.1.1, "GEM errata".

7.1.1 GEM errata

Following are errata to GEM [ref GEM]. The changes presented have been agreed by the appropriate DVB subgroup for publication in a subsequent version of GEM [ref GEM]".

## 4.2 Conventions

Unless otherwise specified, the BNF notation in the present document shall follow the definitions of section 2.1 of RFC 2616 [45].

The text "Void" in a clause title or in a reference designates a clause or reference that existed in a previous version of the present document that has been removed and replaced with "Void" to preserve the numbering of the text that remains.

The inclusion of a normative reference does not imply the inclusion of all of that document. The clauses of a normative reference that apply to the present document are identified where each normative reference is used.

The javadoc accompanying the present document shows the maximum set of APIs and their dependencies. Not all of these are mandatory in all GEM terminals. (For example, org.dvb.service.DvbSIManager is shown as implementing CRIDAccess and SDSRecordAccess even though CRIDAccess is defined in clause 11.6.7, "Integration between protocol independent SI API and TV-Anytime" and hence is only required when the broadband content guide is supported.) Implementations not including an API should omit the reference to it. The present document has been designed to ensure that these omissions happen at clean break points.

### 4.2.1 Void

### 4.2.2 Void

### 4.2.3 Void

### 4.2.4 Conventions within the present document

#### 4.2.4.1 GEM

Use of the term "GEM" within a normative clause of the present document shall be interpreted as referring to the present document.

#### 4.2.4.2 Resident navigator

The present document uses the terms "navigator" and "resident navigator". It is noted that in GEM terminal specifications, it is permissible for some of the functions of the navigator to be delegated to an entity that is not part of the resident software of the terminal, e.g. the OCAP [5] monitor application.

Downloaded or other resident applications that perform some of the policy decisions or functionality of the navigator shall implement a policy that is consistent with the requirements of the present document.

#### 4.2.4.3 DVB service

For the purposes of the present document, references within the present document to DVB services shall be interpreted as meaning any services that may carry GEM applications.

### 4.2.5 References to OCAP

Where the words "shall be supported" or "may be supported" are used referring to a clause from OCAP [5], the following terms in the referenced clause shall be re-defined as follows:

- References to OCAP-J applications or application instances shall be interpreted as references to GEM applications.
- In the case of broadband networks, references to the following XAIT descriptors shall be interpreted as references to the descriptors of the same name defined in clause 10.16.2 "XAIT" of MHP [1].
    - Abstract Service Descriptor.
    - Unbound Application Descriptor.
    - Privileged Certificate Descriptor.

- Requirements on and references to manufacturer applications or host device manufacturer applications or receiver manufacturer applications shall be ignored as they are outside the scope of the present document.
- References to MSO shall be interpreted as references to network operators in general.
- References to the monitor application or monitor applications shall be interpreted as references to privileged applications.
- References to OCAP implementations shall be interpreted as references to GEM implementations.

# 5 Basic architecture (informative)

GEM does not mandate a basic architecture. The section below describes an informative example of one possible architecture for GEM terminal specifications.

## 5.1 Context

At its simplest level, GEM is set in the following context (see figure 4). The software of GEM has access to flows of streams and data, and may write some data to storage. The platform may be able to route streams and data outwards to a sink or store.

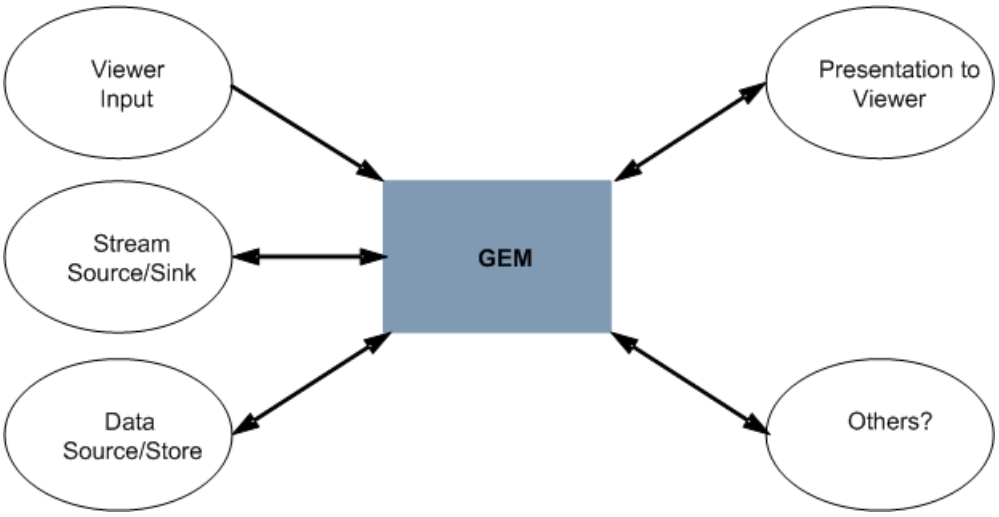



**Figure 4: GEM context**

The platform will receive inputs from Viewer input devices and output communications through a screen or other outputs like loudspeakers to present to the viewer. The platform may have access to communications with remote entities.

The diagram in figure 5 shows a possible set of external interfaces between a GEM terminal and the outside world. This is one example only but it serves to illustrate a series of principles.

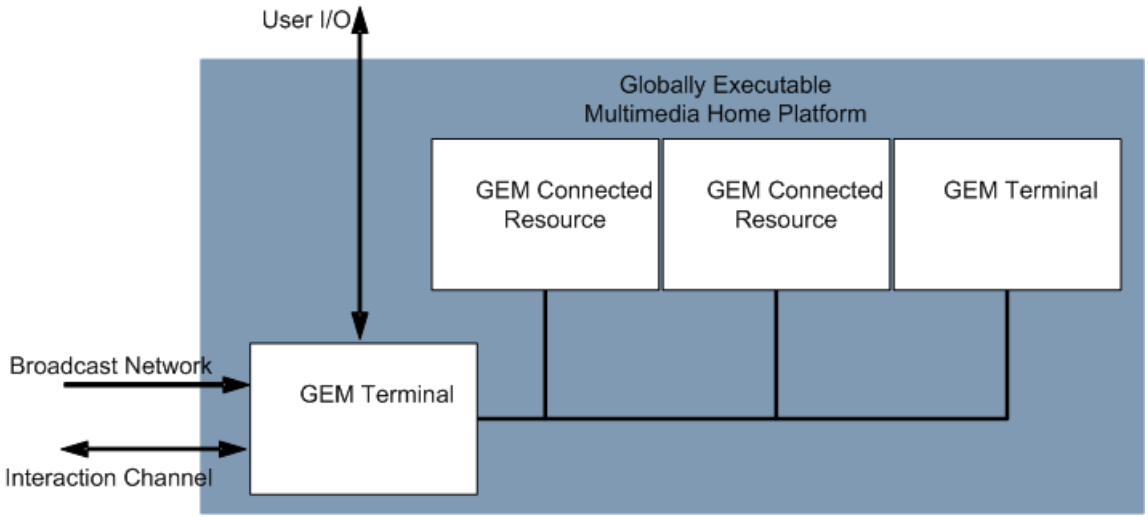



**Figure 5: External interfaces between a GEM terminal and the outside world**

The resources of the GEM terminal, accessible by an application, may be contained in a series of different but connected physical entities.

The local cluster may connect a number of GEM terminals and resources.

A cluster may also include resources which are not part of the GEM infrastructure and are not available to the application.

The local cluster is understood to be consistent with the DVB IHDN specification [30]. The detailed description of GEM in the local cluster is not in the first version of the specification.

## 5.2 Architecture

The Architecture describes how the GEM software elements are organized.

The GEM model considers 3 layers (figure 6):

- Resources.
- System software.
- Applications.

The API lies between the Applications and the System Software seen from the perspective of an application.

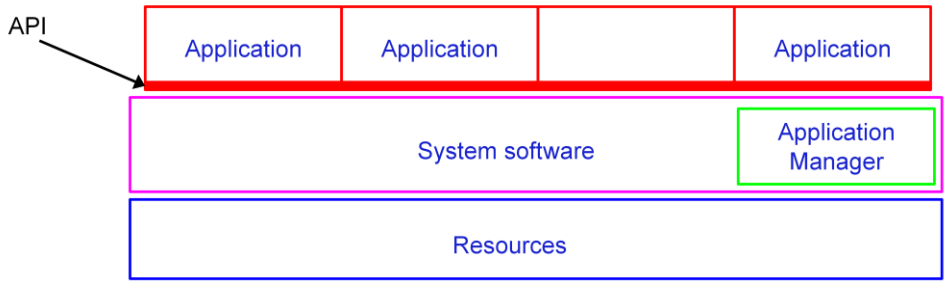



**Figure 6: Basic architecture**

### 5.2.1 Resources

The hardware entities in the platform include a number of functions. They are represented by hardware or software resources. There is no assumption about how they are grouped. The model considers that there can be more than one hardware entity in the total Platform.

From an abstract point of view it makes no difference if the logical resources are mapped into one or several hardware entities. What is important is that resources are provided to the GEM terminal transparently. An application should be able to access all locally connected resources as if they were elements of a single entity.

### 5.2.2 System software

Applications will not directly address resources. The system software brings an abstract view of such resources. This middle layer isolates the application from the hardware, enabling portability of the application.

The implementations of the Resources and System software are not specified in the present document.

#### 5.2.2.1 Application Manager

The system software includes an application management function, which is responsible for managing the lifecycle of all applications, including Interoperable ones.

### 5.2.3 Application

Applications implement interactive services as software running in one or more hardware entities. The interface for GEM applications is a top view from application to the system software.

Figure 6 illustrates an idealized architecture model of the processes which will occur in a GEM terminal. A hierarchy of control is assumed in which each layer controls the processes in adjacent layers. The top layer is responsible for the control of the operation via interactive applications. The Application Manager is part of the System Software and as such is implementation specific. It interacts with all applications.

The System Software implements the API by presenting an abstract model of:

- Streams played from different sources and pipes for conducting them.
- Commands and events.
- Data records or files.
- The hardware resources.

The API provides the associated services to applications.

In fact there are many APIs which implement distinct services and interfaces. These are described in detail in the present document, either by reference to external documents, or by detailed specification.

The present document describes the interfaces between the network, the application and the system software of the GEM terminal. The implementation of any of these three is not specified in the present document.

## 5.3 Interfaces Between a GEM Application and the GEM Terminal

Application(s) use the API to access the actual resources of the receiver, including: databases, streamed media decoders, static content decoders and communications. These resources are functional entities of the receiver and may be finally mapped onto the hardware of the receiver in some manner.

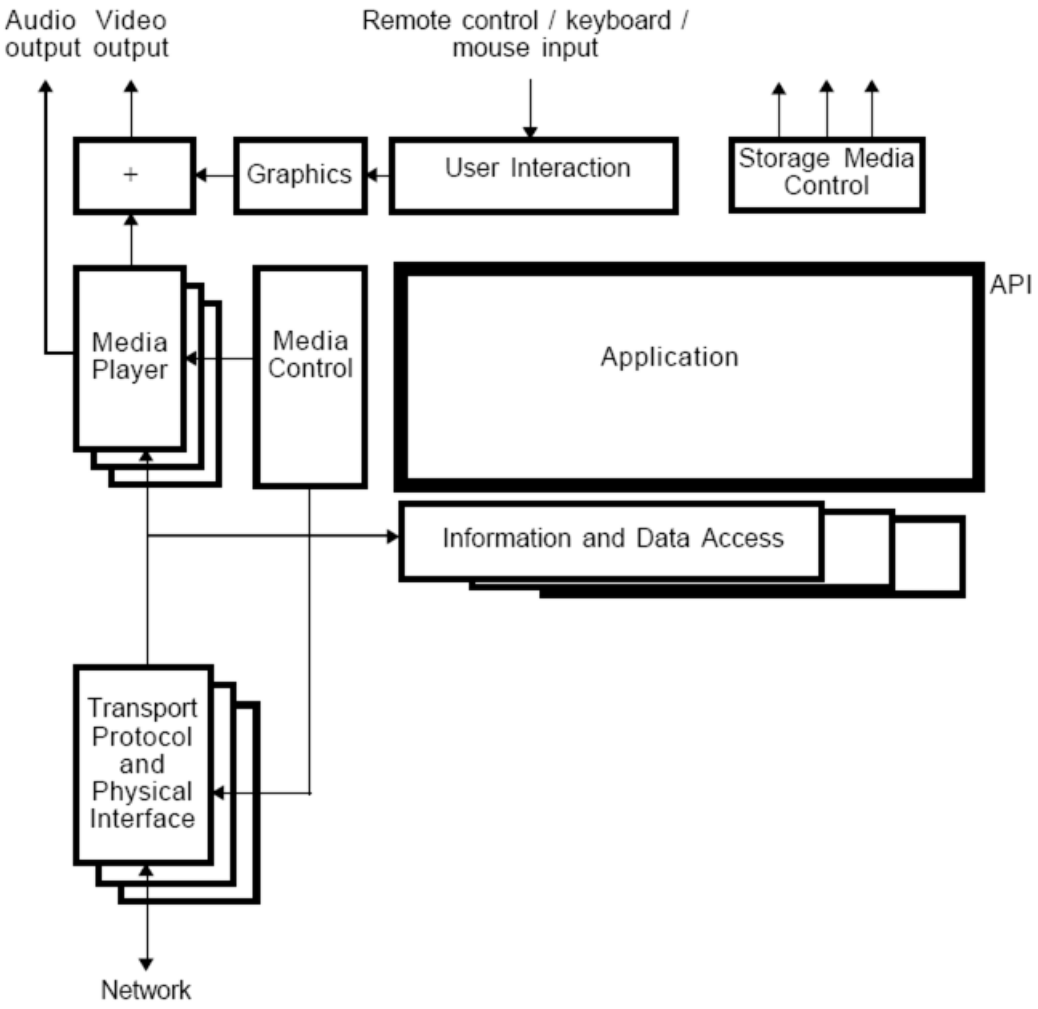



**Figure 7: Interfaces between a GEM application and the GEM system**

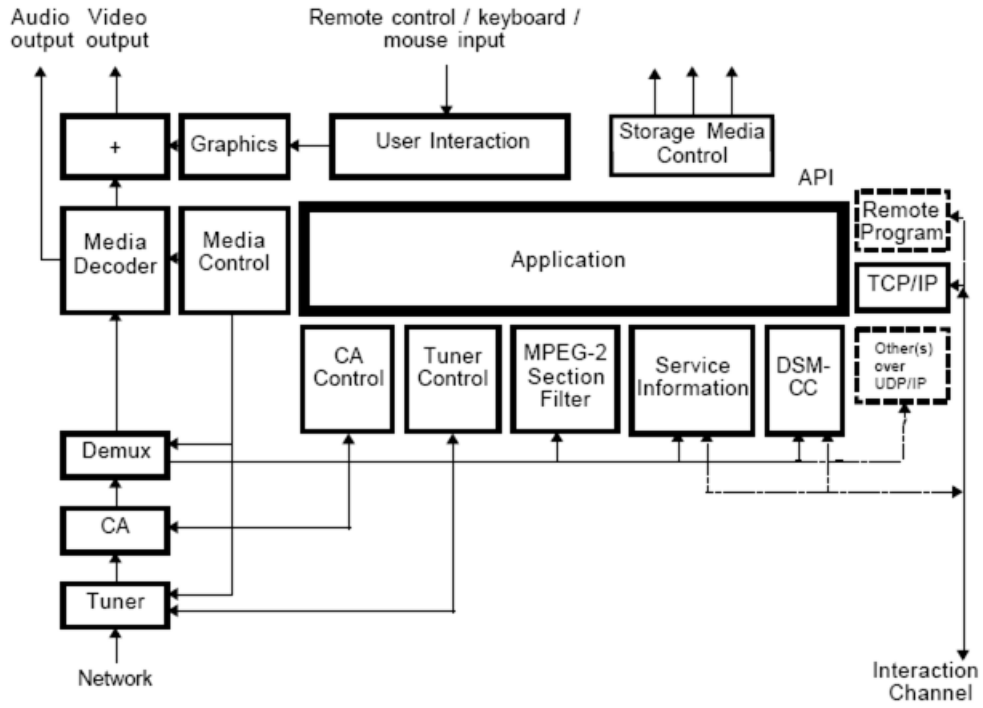



**Figure 8: Interfaces between a GEM application and the GEM platform**

The diagrams in figures 7 and 8 show these interfaces and their relationships to media and information flows within a GEM system - excluding any local cluster issues. The first diagram shows a generic GEM, the second a specific instance of an enhanced broadcast or interactive TV profile system, with optional additions shown.

In figures 7 and 8, only the border between the application and the rest of the system is in the scope of the present document. All the rest of each diagram is implementation dependent and shown for information purposes only.

## 5.4 Plug-ins

A "plug-in" is a set of functionality that can be added to a generic platform in order to provide interpretation of application and content formats not specified by the present document to be included in GEM terminals.

NOTE: Those organizations concerned with interoperation between the standard GEM platform and other platforms need to specify the plug-in properly for such platforms.

The choice of which plug-ins to use must be in the hands of the end-user in order that he can have a choice of sources of service. This option can be exercised in a number of ways, including the purchase of equipment with "built-in" plug-in functionality, the positive selection of a download, or the automatic selection of a download where there is no memory resource limitation.

The plug-in may stay resident where the design of the platform implementation allows. The GEM terminal including the plug-in must behave, once the plug-in is loaded and operational, in the same way as a platform supporting the format of the delegated applications without the use of a plug-in.

Figure 9 illustrates the position of two types of Plug-in in the GEM terminal.

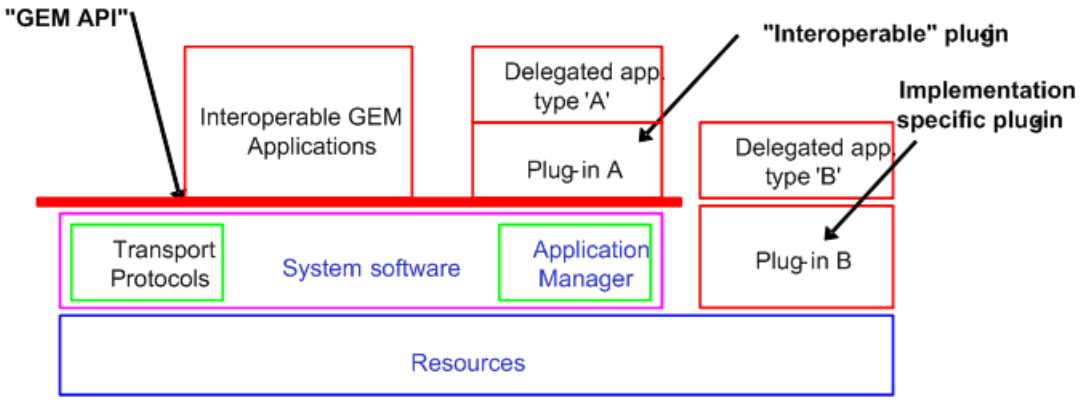



**Figure 9: Illustrative plug-in implementation options**

There are two possible types of Plug-in implementation:

- Using implementation-specific code (e.g. in native code, or using implementation-specific Java APIs). This is called an implementation- specific plug-in, illustrated as plug-in "B" in the figure.
- A GEM application. This is called an interoperable plug-in, illustrated as plug-in "A" in the figure.

The internal specification of both plug-ins ("A" and "B") is outside of the scope of the present document. They are illustrated to show their relationship to the platform.

There are two varieties of inter-operable plug-ins depending on the signalling used for the delegated applications.

- Ones which work with delegated applications signalled with GEM signalling and which implement `org.dvb.application.plugins.Plugin`. These delegated applications are visible to the GEM receiver and are managed by the GEM application manager.
- Ones which work with delegated applications signalled with native signalling and only look like a GEM application to the GEM receiver. These do not implement the plug-in interface. These delegated applications are invisible to the GEM receiver and totally managed by the plug-in.

### 5.4.1 Security Model

Plug-ins must have sufficient access to the resources in the platform to implement the specification concerned. An implementation-specific plug-in may have access to many of the resources of the platform irrespective of the GEM security model. All plug-ins are responsible for managing the security of the applications they execute.

If a plug-ins needs privileged access to resources not available to all downloaded applications (i.e. not in the "sand box") in order to provide equivalent function to the "legacy" supported they will require the appropriate authentication.

# 6 Transport protocols

## 6.1 Introduction

For broadcast targets, in order to be able to talk to the external world, a GEM terminal has to communicate through different network types.

Broadcast only services are provided on systems consisting of a downstream channel from the Service Providers to Service consumers.

Interactive services are provided on systems consisting of a downstream channel together with interaction channels.

For packaged media targets, the GEM terminal can navigate AV streams on the physical carrier and communicate with the external world through different IP-based networks. Wherever GEM refers to broadcast-related services or protocols, a packaged media target uses streams and formats from the packaged media. Note that the MHP/GEM term "interaction channel" refers to IP-based connectivity.

## 6.2 Broadcast channel protocols

For broadcast targets, this clause deals with DVB defined or referenced broadcast channel protocols. This clause does not consider other protocols and the APIs that would provide access to them.

Other protocols and their APIs are considered as extensions to the present document, see annex H.

Figure 10 shows the broadcast channel protocol stack for GEM. As some of the protocols are not required by the present document, not all elements of this figure necessarily apply. See clause 15.6, "Functional equivalents".

The full details of APIs that provide DVB-J applications with access to broadcast protocols are in clause 11, "DVB-J platform".

NOTE: For the packaged media profiles, these definitions might not be used.

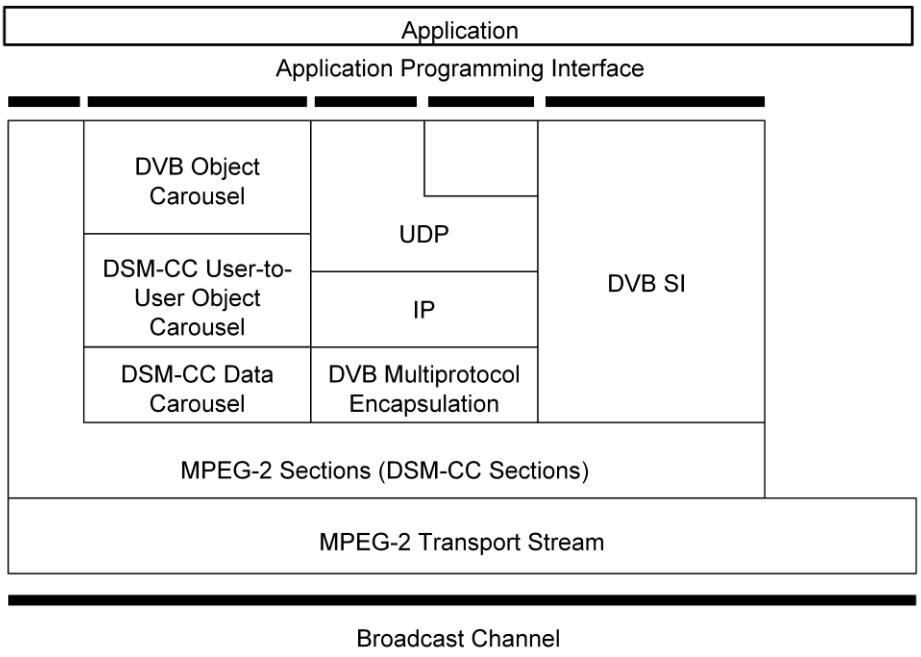



**Figure 10: Broadcast Channel Protocol Stack**

Except in the case of MPEG-2 sections (see clause 6.2.2, "MPEG-2 sections"), when a GEM application attempts to access conditional access scrambled data through one of these broadcast channel protocols, the GEM terminal shall attempt to initiate descrambling of this data without the application needing to explicitly ask for it. Attempts to access

conditional access scrambled data at the level of MPEG-2 sections shall not happen without the application explicitly asking for this. This requirement applies to GEM terminal specifications that include the functional equivalent named "Conditional Access" as specified in clause 15.6, "Functional equivalents".

### 6.2.1 MPEG-2 transport stream

MPEG-2 Transport Stream is defined in ISO/IEC 13818-1 [36].

### 6.2.2 MPEG-2 sections

MPEG-2 private sections as defined in ISO/IEC 13818-1 [36] is based on the MPEG-2 Transport Stream protocol in clause 6.2.1.

### 6.2.3 DSM-CC private data

DSM-CC Private Data protocol as defined in ISO/IEC 13818-6 [38].

### 6.2.4 DSM-CC data carousel

DSM-CC Data Carousel as defined in ISO/IEC 13818-6 [38].

### 6.2.5 Object carousel

DSM-CC User-to-User Object Carousel protocols as defined in ISO/IEC 13818-6 [38] with the restrictions and extensions as defined in EN 301 192 [32], TR 101 202 [i.9] and annex B, "Broadcast filesystem and trigger transport".

Use of the Object Carousel protocol is not required for GEM terminal specifications. GEM terminal specifications that do not include the functional equivalent named "Object Carousel" as specified in clause 15.6, "Functional equivalents" shall specify a functional equivalent that satisfies the requirements of the API specified in Annex P, "Broadcast transport protocol access".

#### 6.2.5.1 Void

#### 6.2.5.2 Void

#### 6.2.5.3 Loss of carousel behaviour

Under some conditions, carousel data streams servicing broadcast file systems may become unavailable. The conditions for temporary disconnection and reconnection of carousel are defined in clause 9.1.5, "Persistence of Applications Across Service Boundaries".

DSM-CC User-to-User Object Carousels as described in clause 6.2.5, "Object carousel" become permanently unavailable due to changes in the signalling including the following:

- The component signalled as carrying the DSI is removed from the PMT.
- The value of carousel ID associated with the carousel changes.
- The program disappears from the PAT.
- After an implementation dependent time general failure of the signalling (e.g. non-transmission of the PMT).

Additionally, carousels also become permanently unavailable when loss of connection to a temporarily disconnected carousel becomes permanent.

The conditions for permanent loss of a carousel may be specified differently in GEM terminal specifications that do not include the DSM-CC User-to-User Object Carousel as described in clause 6.2.5, "Object carousel". However, GEM terminal specifications shall specify conditions for permanent loss of a carousel. The conditions for temporary

disconnection and reconnection of a carousel defined in clause 9.1.5, "Persistence of Applications Across Service Boundaries" apply to all GEM terminal specifications.

When carousel data streams become unavailable, implementations may continue to provide data from carousels which have been lost where they have that data cached. The extent of this is clearly implementation dependent (but limited by clause B.5, "Caching"). It is also implementation dependent how this changes with time. Permanently lost carousels shall never be restored automatically. Temporary disconnections and reconnections are automatic and are largely invisible to the application. However, implementations must preserve Java IO semantics. For example, the `InputStream.available()` method should accurately report the number of bytes available without blocking.

Data not in such a cache shall be unavailable to applications. When applications attempt to access unavailable data from permanently lost carousels, the operation shall fail. The failure mode shall be one appropriate to the content format and the mechanism being used to access the data.

Failure modes for DVB-J applications are defined in clause 11.5.1.3, "Behaviour following loss of a broadcast carousel".

When an application attempts to access unavailable data from a temporarily disconnected carousel, the operation shall block until the data becomes available, or it is interrupted. This includes any operations that indirectly cause synchronous loading operations, as stated in clause 11.5.1.1, "Constraints on the java.io.File methods for broadcast carousels".

Upon reconnection to a carousel, the system shall start fetching any outstanding data. Any blocked I/O operations shall return once the data is received.

A temporarily disconnected service may become permanently lost if the system determines that the loss of connection is irrecoverable - as stated in annex P "(normative): Broadcast Transport Protocol Access", org.dvb.dsmcc.ServiceDomain. The cases in which this may occur are:

- The carousel becomes permanently unavailable, as stated above.
- The period since the carousel became disconnected is at least 60 seconds.

If this occurs, any blocked I/O operations shall terminate with `InterruptedIOException` and the `ServiceDomain` shall enter the detached state. An implementation may choose to use longer timeouts or other strategies to determine when to permanently lose carousels, within the constraints listed above. For example, boot carousels for running applications may be kept in preference to carousels mounted by applications.

### 6.2.6 Protocol for delivery of IP multicast over the broadcast channel

DVB Multiprotocol Encapsulation as defined in EN 301 192 [32] provides support for IP and is based on the DSM-CC Private Data protocol. EN 301 192 [32] only defines support for carriage of multicast IP in MPE and not support for carriage of unicast IP.

Use of the DVB Multiprotocol Encapsulation protocol is not required for GEM terminal specifications. If, however clause 11.5.2, "Support for Multicast IP over the Broadcast Channel" is supported, some mechanism for delivery and signalling of IP multicast over the broadcast channel shall be specified.

NOTE: This feature is optional in all profiles of GEM.

### 6.2.7 Internet Protocol (IP)

Internet Protocol as defined in RFC 791 [48].

### 6.2.8 User Datagram Protocol (UDP)

User Datagram Protocol as defined in RFC 768 [47].

### 6.2.9 Service information

Use of DVB service information is not required for GEM terminal specifications, however some mechanism for delivery of service information that is sufficient for the SI access mechanisms required by GEM shall be specified.

DVB Service Information as defined in EN 300 468 [14] and ETR 211 [i.5] may be taken as an informative example of such a mechanism for GEM terminal specification that do not include the functional equivalent named "SI" as specified in clause 15.6, "Functional equivalents".

### 6.2.10 IP signalling

IP Notification Table (INT) as defined in EN 301 192 [32].

Use of this signalling is not required for GEM terminal specifications, however a functional equivalent that satisfies the requirements of clause 11.5.2, "Support for Multicast IP over the Broadcast Channel" is required if IP over the broadcast channel is supported.

NOTE: This feature is optional in all profiles of GEM.

## 6.3 Interaction channel protocols

This clause deals with the DVB defined or referenced interaction channel protocols. This clause does not consider other protocols and the APIs that would provide access to them. Other private protocols and possibly APIs are not precluded and are outside of the scope of GEM.

Figure 11 illustrates the set of DVB defined interaction channel protocols that are accessible by GEM applications in some or all profiles (see clause 15, "Detailed platform profile definitions"). The full details of the APIs that provide access to these interaction protocols are in clause 11, "DVB-J platform".

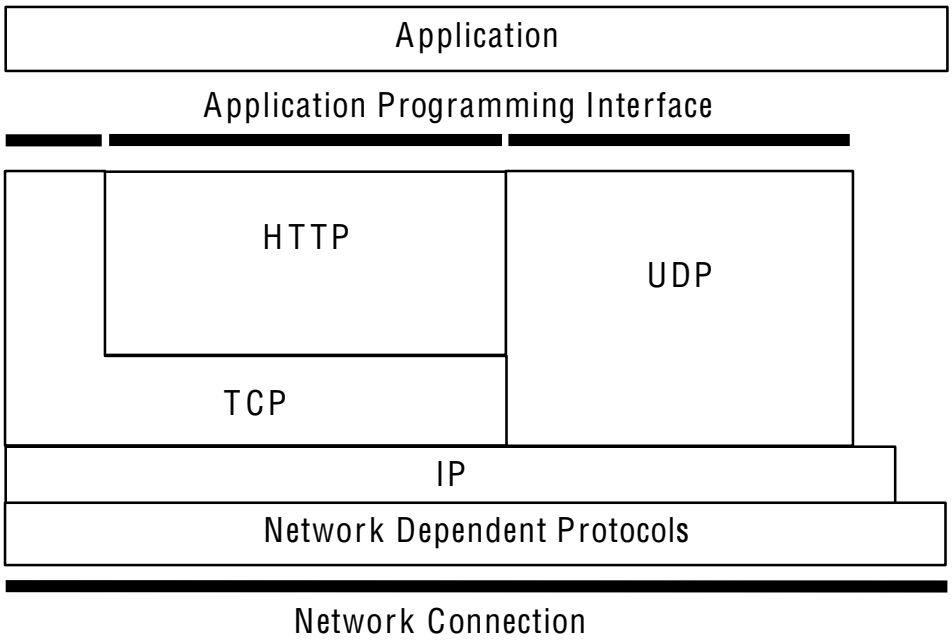



**Figure 11: Interaction Channel Protocol Stack**

### 6.3.1 Network Dependent Protocols

As defined in ES 200 800 [19], ETS 300 801 [20], EN 301 193 [i.2], EN 301 195 [i.3], EN 301 199 [i.4], TR 101 201 [i.8], EN 301 790 [i.11] respectively for CATV, PSTN/ISDN, DECT, GSM, LMDS SMATV and satellite networks.

For connections based interaction channels, the PPP protocol is used as defined in RFC 1332 [i.13], RFC 1661 [i.14], RFC 1717 [i.15]. Network supplied DNS server addresses shall be supported as in RFC 1877 [i.12].

NOTE: The protocols listed in this clause are not explicitly required in any GEM profile.

### 6.3.2 Internet Protocol (IP)

Internet Protocol as defined in RFC 791 [48].

### 6.3.3 Transmission Control Protocol (TCP)

Transmission Control Protocol as defined in RFC 793 [49].

### 6.3.4 UNO-RPC

The UNO-RPC consists of the Internet Inter-ORB Protocol (IIOP) as specified in CORBA/IIOP [31].

### 6.3.5 UNO-CDR

The UNO-CDR as defined in CORBA/IIOP [31].

### 6.3.6 DSM-CC User to User

DSM-CC User-to-user as defined in ISO/IEC 13818-6 [38] with the restrictions and extensions as defined in EN 301 192 [32] and TR 101 202 [i.9].

### 6.3.7 Hypertext Transfer Protocol (HTTP)

#### 6.3.7.1 HTTP 1.1

Hypertext Transfer Protocol as defined in RFC 2616 [45].

NOTE: HTTP 1.1 support as specified is not required in the broadcast, packaged media and IPTV target of GEM. This does not preclude a GEM terminal specification using a different profile of HTTP.

#### 6.3.7.2 GEM profile of HTTP 1.0

HTTP 1.0 is defined in RFC 1945 [68]. GEM terminals supporting HTTP 1.0 are required to support persistent connections as defined below. GEM terminals implementing profiles where HTTP 1.0 is required are also allowed to implement subsequent versions of the HTTP specification (e.g. HTTP 1.1, RFC 2616 [45]) as long as these are backwards compatible and persistent connections or their equivalent in the subsequent version of the specification are supported.

With no support for persistent connections, a separate TCP connection has to be established to fetch each URL, increasing the load on HTTP servers and causing congestion. The use of multiple separate image files and other associated data, for example often require a client to make multiple requests to the same server in a short amount of time. Therefore an MHP terminal implementation shall include support for persistent connections.

The following section explains how persistent connection is to be used within GEM. The original form of HTTP 1.0 persistent connections is defined as informational text in RFC 2616 [45]. The normative specification text included here is intended to be fully compatible with this RFC.

##### 6.3.7.2.1 HTTP 1.0 persistent connections

When connecting to a server, a GEM terminal shall send the Keep-Alive connection-token:

```
Connection: Keep-Alive
```

An HTTP server (1.0 or 1.1) would then respond with the Keep-Alive connection token and the client may proceed with an HTTP 1.0 (Keep-Alive) persistent connection.

#### 6.3.7.2.2 The Keep-Alive Header

When the Keep-Alive connection-token has been transmitted with a request or a response, a Keep-Alive header field may also be included. The Keep-Alive header field takes the following form:

```
Keep-Alive-header = "Keep-Alive" ":" 0# keepalive-param
keepalive-param = param-name "=" value
```

where the "0#" notation means that the "keepalive-param" field may be repeated 0 or more times and they are separated by a comma character "," if the field is included more than once.

The Keep-Alive header itself is optional, and is used only if a parameter is being sent. The present document does not define any parameters.

If the Keep-Alive header is sent, the corresponding connection token must be transmitted. The Keep-Alive header must be ignored if received without the connection token.

#### 6.3.7.2.3 GEM and proxies

A GEM terminal must not send the Keep-Alive connection token to a HTTP 1.0 proxy server, as HTTP 1.0 proxy servers do not obey the rules of HTTP 1.1 for parsing the Connection header field.

If a GEM terminal sends Keep-Alive to a proxy server that doesn't understand Connection, which would then erroneously forward it to the next inbound server, which would establish the Keep-Alive connection and result in a hung HTTP 1.0 proxy waiting for the close on the response. The result is that the GEM terminal must be prevented from using Keep-Alive when talking to proxies that doesn't understand Connection.

#### 6.3.7.2.4 Version compatibility

An HTTP 1.1 server may also establish persistent connections with a GEM terminal upon receipt of a Keep-Alive connection token. However, a persistent connection with a GEM terminal can't make use of the chunked transfer coding, and therefore must use a content-length for marking the ending boundary of each message.

For servers used for GEM applications, it is recommended that if HTTP 1.1 servers are used, they should support the HTTP 1.0 persistent connections when initiated by an HTTP 1.0 client using the Keep-Alive connection token.

### 6.3.7.3 HTTPS

This is described in RFC 2818 [73].

## 6.3.8 User Datagram Protocol (UDP)

User Datagram Protocol as defined in RFC 768 [47].

## 6.3.9 DNS

GEM terminals shall implement at least the DNS resolver protocols that enable forward translation of fully qualified domain names to IP addresses as defined by RFC 1034 [64] and RFC 1035 [62] and clarified by RFC 1982 [65] and RFC 2181 [66].

In connection based return channels, where DNS server addresses are provided both from the network as part of connection setup and from a GEM application, those provided from the network shall take precedence.

## 6.3.10 Additional Transport Protocols

Additional transport protocols that are proprietary and/or specific to a network service provider may be supported through the registration of interaction channel service providers (see org.dvb.spi.ict.InteractionChannelTransportProvider).

# 6.4 Transport protocols for application loading over the interaction channel

Three scenarios are envisaged:

- The file system is completely implemented in the broadcast channel (the classic MHP 1.0 model which is not described further here).
- File system implemented only via the interaction channel.
- Hybrid between broadcast stream and interaction channel.

## 6.4.1 File system implemented only by the interaction channel

This clause addresses the case where the interaction channel presents the only file system, e.g. in MHP [1], where the protocol ID is 0x0003.

GEM terminal specifications may define other protocol ID values that use this mechanism.

### 6.4.1.1 File system logical structure

The list of items signalled in the transport protocol descriptor(s) (see clause 15.6.1.2.1, "Transport protocol descriptor") are considered to form a single name space.

The GEM terminal when trying to locate a file specified by an incomplete (relative) filename, shall attempt to fetch a file from each of the items in that list in the order in which they are found in the list until the file is found or the list is exhausted.

The items in the list shall either be references to .zip files (see ZIP [67]) or base URLs ending in "slash" (/) onto which the path of the requested file shall be concatenated. Any items in the list which are not one of these two types shall be ignored. Errors in discovering whether a specific file can be reached through a specific list item shall be ignored. The list must contain at least one item.

As an example, consider the GEM platform retrieving the "dvb.fontindex" file for an application from an interactive channel for which the AIT File includes two transport protocol descriptors with the following contents:

**Table 1: Example transport descriptor selector byte payload**

| URL base | URL extension |
|---|---|
| "http://www.dvb.org" | "applications/application1.zip" |
| | "graphics/" |
| | "shared/utils.zip" |
| "http://www.ebu.ch" | "general/misc.zip" |
| "http://www.dvb.org" | "other_stuff/we_dont_use_this_very_often.zip" |

Then the search order for "dvb.fontindex" would be the following:

a) In the contents of http://www.dvb.org/applications/application1.zip.

b) From the URL http://www.dvb.org/graphics/dvb.fontindex.

c) In the contents of http://www.dvb.org/shared/utils.zip.

d) In the contents of http://www.ebu.ch/general/misc.zip.

e) In the contents of http://www.dvb.org/we_dont_use_this_very_often.zip.

The search order for "abc/dvb.hashfile" would be the following:

a) For "abc/dvb.hashfile" in the contents of http://www.dvb.org/applications/application1.zip.

b) From the URL http://www.dvb.org/graphics/abc/dvb.hashfile.

c) For "abc/dvb.hashfile" in the contents of http://www.dvb.org/shared/utils.zip.

d) For "abc/dvb.hashfile" in the contents of http://www.ebu.ch/general/misc.zip.

These file system semantics shall be followed for all files fetched by or on behalf of a GEM application. This includes the files whose names and locations are standardized by the present document, e.g. 'dvb.fontindex', 'dvb.hashfile', 'dvb.signature.x' and dvb.certificate.x'.

For DVB-J, with the following application location descriptor:

a) base_directory "/app1/starting".

b) classpath_extension "/shared;/app1/later".

c) initial_class "org.dvb.demo.myXlet".

The search order for a class would be the following:

a) For "app1/starting/org/dvb/demo/myXlet.class" in the contents of http://www.dvb.org/applications/application1.zip.

b) From the URL http://www.dvb.org/graphics/app1/starting/org/dvb/demo/myXlet.class.

c) For "app1/starting/org/dvb/demo/myXlet.class" in the contents of http://www.dvb.org/shared/utils.zip.

d) For "app1/starting/org/dvb/demo/myXlet.class" in the contents of http://www.ebu.ch/general/misc.zip.

e) other_stuff/we_dont_use_this_very_often.zip.

f) For "/app1/starting/shared/org/dvb/demo/myXlet.class" in the contents of http://www.dvb.org/applications/application1.zip.

g) From the URL http://www.dvb.org/graphics/app1/starting/shared/org/dvb/demo/myXlet.class.

h) For "/app1/starting/shared/org/dvb/demo/myXlet.class" in the contents of http://www.dvb.org/shared/utils.zip.

i) For "/app1/starting/shared/org/dvb/demo/myXlet.class" in the contents of http://www.ebu.ch/general/misc.zip.

j) other_stuff/we_dont_use_this_very_often.zip.

k) For "/app1/starting/app1/later/org/dvb/demo/myXlet.class" in the contents of http://www.dvb.org/applications/application1.zip.

l) From the URL http://www.dvb.org/graphics/app1/starting/app1/later/org/dvb/demo/myXlet.class.

m) For "/app1/starting/app1/later/org/dvb/demo/myXlet.class" in the contents of http://www.dvb.org/shared/utils.zip.

n) For "/app1/starting/app1/later/org/dvb/demo/myXlet.class" in the contents of http://www.ebu.ch/general/misc.zip.

o) other_stuff/we_dont_use_this_very_often.zip.

#### 6.4.1.2 File transfer

Files to be transferred from an "http" URL shall be transferred using either HTTP 1.1 as defined in clause 6.3.7.1, "HTTP 1.1" or the profile of HTTP specified under clause 6.3.7.2, "GEM profile of HTTP 1.0".

Files to be transferred from an "https" URL shall be transferred as defined in clause 6.3.7.3, "HTTPS".

For other URL schemes, if a registered interaction channel transport protocol service provider exists for the relevant URI scheme, that provider shall be used.

NOTE: That transport protocol service providers may not be registered for the http: or https: URI schemes (i.e. service providers cannot override the GEM implementation's internal implementation of the HTTP and HTTPS protocols).

#### 6.4.1.3 Class encoding

The classes of DVB-J applications can be delivered either as discrete class files or within ZIP files (see ZIP [67]).

NOTE 1: This contrasts to the broadcast case where object carousel files are simply class files.

The classes of DVB-J applications shall be delivered as discrete class files at the level of the file system.

NOTE 2: This means that DVB-J class files are allowed to be held on an HTTP server either as discrete class files or in ZIP files (see ZIP [67]) where ZIP files form part of the overall file system.

#### 6.4.1.4 Directory listing in this file system

For the authentication (see clause 12.4.1.5, "Special authentication rules"), this file system shall be considered to be one not supporting directory listing.

For listing the files in a directory (see clause 14.7, "Files and file names"), this file system shall be considered to be one where the MHP terminal is never certain of the complete contents of a directory.

### 6.4.2 Hybrid between broadcast stream and interaction channel

In the hybrid case all directory information is provided in the broadcast stream but some, or possibly all, of the file contents are provided via the interaction channel.

#### 6.4.2.1 File transfer

##### 6.4.2.1.1 Broadcast file delivery

In GEM terminals that use the DSM-CC User-to-User Object Carousel, the file contents are carried via a `BIOP::File` as is the normal case for a DSM-CC User-to-User Object Carousel as described in clause 6.2.5, "Object carousel". The IOR from the file binding to the file contents uses either a `BIOPProfileBody` or a `LiteOptionsProfileBody`.

##### 6.4.2.1.2 Interaction channel delivery

In GEM terminals that use the DSM-CC User-to-User Object Carousel, the IOR from the file binding to the file contents uses the `HTTPProfileBody` (see table 2) to identify the location of the file contents on the interaction channel.

This form of IOR shall only be used for `BIOP::File` objects.

See clause 6.3.7.2, "GEM profile of HTTP 1.0" for the profile of HTTP that is used to retrieve the contents of the file.

##### 6.4.2.1.3 HTTPProfileBody

The HTTP Profile body specifies `host`, `port` and `path_segments`. In HTTP requests these are concatenated to form a URL of the form:

```
http://host:port/path_segments
```

**Table 2: HTTP Profile Body syntax**

| Syntax | bits | Type | Value | Comment |
|---|---|---|---|---|
| HTTPProfileBody { | | | | |
| profileId_tag | 32 | uimsbf | 0x44564200 | |
| profile_data_length | 32 | uimsbf | * | |
| profile_data_byte_order | 8 | uimsbf | 0x00 | big endian byte order |
| version.major | 8 | uimsbf | 0x01 | protocol major version 1 |
| version.minor | 8 | uimsbf | 0x00 | protocol minor version 0 |
| host_data_length | 8 | uimsbf | N1 | |
| for (k=0; k<N1; k++) { | | | | |
| host_data | 8 | uimsbf | + | |
| } | | | | |
| port | 16 | uimsbf | | |
| objectKey_length | 16 | uimsbf | N2 | |
| for (k=0; k<N2; k++) { | | | | |
| objectKey_data | 8 | uimsbf | + | |
| } | | | | |
| } | | | | |
| NOTE: The tag values 0x44564201 to 0x44564201 have been reserved for future use by the DVB. 0x44 is ASCII for "D", 0x56 is "V" and 0x42 is "B". | | | | |

version: These 8-bit fields indicate the version of the protocol that the server will use to deliver the file specified. The version value 1.0 indicates that the transport protocol is as defined in clause 6.3.7.2, "GEM profile of HTTP 1.0".

host_data: These bytes convey the identifier of the Internet host to which messages may be sent. It may be a fully qualified domain name or Internet standard "dotted decimal" form (e.g. "192.231.79.52").

port: This 16 bit unsigned integer is the TCP/IP port number at the specified host where the target agent is listening for requests.

objectKey_data: These bytes form a string which carries the path_segments portion of the URL which uniquely identifies the object on the server RFC 2396 [46].

## 6.5 IPTV protocols

GEM terminal specifications supporting the IPTV target shall define protocols for the delivery of GEM services and service information. GEM terminal specifications may mandate support for any of the protocols specified by the following subclauses.

NOTE: GEM-IPTV protocols may rely on the provider interfaces, defined in clause 9.11, "Providers".

### 6.5.1 Transport protocols

#### 6.5.1.1 Service Discovery and Selection

The protocols for unicast and multicast delivery of service discovery and selection information are defined in clause 5.4 of TS 102 034 [80].

#### 6.5.1.2 Broadband Content Guide

The protocols for unicast and multicast delivery of broadband content guide information are defined in clause 4 of TS 102 539 [81].

#### 6.5.1.3 Real Time Protocol (RTP)

The use of RTP is defined in clause 7 of TS 102 034 [80].

#### 6.5.1.4 Real Time Streaming Protocol (RTSP)

The use of RTSP is defined in clause 6 of TS 102 034 [80].

#### 6.5.1.5 Internet Group Management Protocol (IGMP)

The use of IGMP is defined in clause 7 of TS 102 034 [80].

### 6.5.2 Service information and metadata protocols

#### 6.5.2.1 IP service discovery

This is defined by clause 5.2 of TS 102 034 [80] excluding clause 5.2.6.6.

GEM terminals shall monitor and detect changes in this signalling however there is no requirement to do this more frequently than once per day.

#### 6.5.2.2 Broadband content guide

This is defined by clauses 5, 6 and 7 of TS 102 539 [81] and clause 5.2.6.6 of TS 102 034 [80].

## 6.6 OTT Protocols

### 6.6.1 Protocols for streaming

Unicast streaming using HTTP 1.1 [45] shall be supported for the OTT target.

NOTE: HTTP 1.1 support the range header in a GET request, which allows partial retrieval for use in cases such as trick play or seek operations to reduce unnecessary network usage.

To optimize the streaming user experience over best-effort broadband lines, the receiver shall implement proper buffering and playback strategies to cope with varying network conditions. The details of such strategies are implementation dependant. Adaptive streaming is one possibility to address bandwidth variations.

#### 6.6.1.1 Adaptive Streaming

The user experience with Streamed CoD based on a single instance (single coding rate) of a given content may be adversely affected by the different and varying conditions of each customer's unmanaged broadband access. Different because of possible different access line profiles (either commercial or technical); varying because of the continuously changing end-to-end network conditions.

Several solutions have been recently announced or introduced into the market (e.g. Adobe, Microsoft, Apple, Widevine, 3GPP, MPEG, OIPF), but there is not yet a standardized and DVB-endorsed solution for adaptive streaming.

A common mechanism for adaptive streaming is when a service provides a content item in multiple bitrates in a way that enables a terminal to adapt to variations in the available bandwidth by seamlessly switching from one version to another, at a higher or lower bitrate, while receiving and playing the content.

This is achieved by encoding a content item in alternative representations of different bitrates and segmenting these representations into temporally aligned and independently encoded segments.

Typically adaptive streaming solutions provide a streaming manifest file describing the available multimedia content segments at various bit-rates. This information is used within the JMF player to select the appropriate segments for the current network condition.

To optimize the user experience over broadband lines, the receiver must implement buffering and playback to cope with varying network conditions. Buffering and adaptation to changing network conditions should be handled at platform and protocol level, but applications are able to provide customers with some rough technical information about content and/or network and/or client characteristics/status in the context of Streamed CoD (see Monitoring API in Annex N.2).

Since GEM is protocol-agnostic, no assumptions are made on the particular manifest file supported by a given platform. GEM does not mandate a specific adaptive streaming protocol.

### 6.6.2 Protocols for download

If content download is supported, HTTP 1.1 [45] shall be supported. The content download mechanism is agnostic to the file formats that are transferred.

NOTE: HTTP 1.1 supports the range header in a GET request, which allows partial retrieval for continuing failed or interrupted download requests.

# 7 Content formats

NOTE: This clause contains definitions referenced from other parts of the present document. Use of these formats may be optional, or it may be possible to replace them with a functional equivalent.

## 7.1 Static formats

### 7.1.1 Bitmap image formats

#### 7.1.1.1 Image encoding restrictions

Any indications in the transmitted image with respect to pixel scaling, colour space or gamma are to be ignored in the presenting of the image. One image pixel shall be mapped to one graphics pixel in the current graphics configuration, unless otherwise scaled by the application directly.

See also clause 7.5, "Colour representation".

#### 7.1.1.2 JPEG

JPEG as defined in ISO/IEC 10918-1 [34] using the JFIF [42] file exchange format.

Only coding using sequential DCT-based mode or progressive DCT-based mode is required to be supported by implementations.

Specifically, lossless and hierarchical modes and arithmetic coding of DCT coefficients need not be supported.

The thumbnail feature of JFIF [42] need not be supported. GEM terminals not supporting thumbnails shall skip them if present and continue decoding the rest of the image.

See also clause 15.3, "JPEG - restrictions" for other possible restrictions.

#### 7.1.1.3 PNG

PNG is defined as in PNG Version 1.0 [43] with the following notes and modifications:

- Later versions of PNG introduce additional chunk types, which are not required to be supported
- See also clause 15.1, "PNG - restrictions".
- GEM terminal specifications may allow or require processing of colour space or gamma information in image transformations.

#### 7.1.1.4 GIF

GIF is defined as in GIF 89a [33].

### 7.1.2 MPEG-2 I-Frames

MPEG-2 I-Frames are defined as in ISO/IEC 13818-2 [37].

The payload of a file delivering an MPEG-2 I frame shall:

- Be a valid video_sequence() including a sequence_extension().
- Contain one I frame only, i.e. one picture_header(), one picture_coding_extension(), and one picture_data() encoded as an intra coded frame, with picture structure = "frame".

That is the structure is:

```
sequence_header()
sequence_extension()
extension_and_user_data(0)
optional group_of_pictures_header() and extension_and_user_data(1)
picture_header ( picture_coding_type = "I frame")
picture_coding_extension ( picture_structure = "frame picture")
extension_and_user_data(2)
picture_data()
sequence_end_code()
```

MPEG I-frames shall be presented with no DecFC (except for any scaling) i.e. the raster's encoded aspect ratio and any AFD shall be ignored.

### 7.1.3 MPEG-2 Video "drips"

The drip feed mode consists of letting an application progressively feed the MPEG-2 video decoder with chunks of an MPEG-2 video stream. In this mode, it is only required for the decoder to handle I and P frames (i.e. not B frame). Each chunk shall contain one frame and a certain number of syntactic elements (as described in ISO/IEC 13818-2 [37]) such as `sequence_header()` or `group_of_picture_header()`.

Firstly, the content of each of the chunks of bytes fed to the decoder shall comply with the following syntax:

```
optional {
        sequence_header()
        sequence_extension()
        extension_and_user_data(0)
        optional {
                group_of_pictures_header()
                extension_and_user_data(1)
        }
}

picture_header ( picture_coding_type = "I frame" or "P frame")
picture_coding_extension ( picture_structure = "frame picture")
extension_and_user_data(2)
picture_data()
optional {
        sequence_end_code()
}
```

In addition, the overall concatenation of chunks over time shall respect the authorized combinations of syntactic elements described in ISO/IEC 13818-2 [37] to build a legal MPEG-2 video stream. The following diagram, extracted from ISO/IEC 13818-2 [37], reflect the rules defined in that standard:

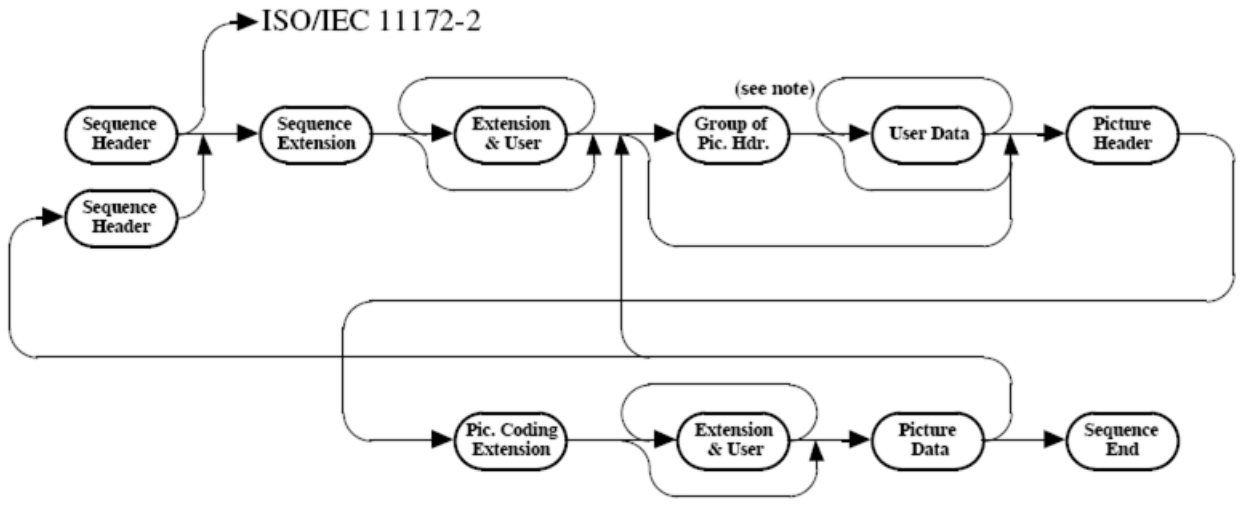



**Figure 12: MPEG-2 Video Stream**

NOTE 1: After a GOP the first picture is an I-picture.

The following restrictions are applied to P-frames:

- The P-frame shall contain no prediction information (i.e. no motion vector shall be present in macroblock elements).
- The allowed macroblock_types for P-frames are:
  - "Intra" (i.e. VLC code 0001 1).
  - "Intra, Quant" (i.e. VLC code 0000 01).
  - "No MC, Coded" (i.e. VLC code 01).
  - "No MC, Coded, Quant" (i.e. VLC code 0000 1).

NOTE 2: The standard semantics for P-frames allow `macroblock_escape` and `macroblock_address_increment` to signal skipped macroblocks. This allows P-frames to be very sparse, only carrying macroblocks positioned at certain locations on the screen. This contrasts with semantics for an I-frame where macroblocks are required to fill the full screen.

If invalid content is fed to the MPEG-2 video decoder, the content is discarded and there are no guarantees when subsequent valid chunk of byte fed to the decoder will be displayed (unless the decoder is restarted).

This mode requires the decoder to be in the "low delay" mode as defined in ISO/IEC 13818-2 [37].

This mode can be used by connecting a `org.dvb.media.DripFeedDataSource` instance to a Player representing a MPEG-2 video decoder. See annex N, "Streamed media API extensions".

MPEG video drips shall be presented with no DecFC (except for any scaling) i.e. the raster's encoded aspect ratio and any AFD shall be ignored.

### 7.1.4 Monomedia format for audio clips

The format for audio clips is MPEG-1 Audio (Layer 1 and 2) ES data as defined as in ISO/IEC 11172-3 [35] and constrained in TS 101 154 [2].

Each "file" of audio content is a binary data file carrying Audio elementary stream data. Each "file" delivers an integer number of audio access units and the first byte of each file is the first byte of an audio access unit. The MPEG Audio data in all other respects conforms to the specifications provided in TS 101 154 [2].

Implementations decoding audio clips can assume that they have an approximately constant number of bytes per second. If this not true then the behaviour is implementation dependent.

Use of the MPEG-1 format as defined above is not required for GEM terminal specifications for packaged media targets. This clause shall apply to GEM terminal specifications that include the functional equivalent named "Audio Clips" as specified in clause 15.6, "Functional equivalents". GEM terminal specifications that do not include this definition shall specify a functional equivalent that satisfies the requirements of the APIs mentioned in the table as included in clause 15.2, "Minimum media formats supported by DVB-J APIs".

### 7.1.5 Monomedia format for text

Java modified UTF-8 as defined in the specification of the method `java.io.DataOutput.writeUTF`.

NOTE: Based on ISO/IEC 10646-1 [15] but modified with respect to the encoding of the character code zero.

#### 7.1.5.1 Built-in character set

See annex E, "Character set".

## 7.2 Media streaming formats

### 7.2.1 Audio

At least one format for streaming audio shall be specified in a GEM terminal specification.

### 7.2.2 Video

At least one format for delivering standard definition streaming video shall be specified in a GEM terminal specification.

### 7.2.3 Subtitles

Support for DVB subtitles, teletext subtitles or any other form of subtitles is optional in the present document.

NOTE: OCAP [5] does not include support for subtitles. It does include US closed-captioning which is somewhat similar, but has different regulatory requirements and usage models.

The content formats supported for subtitles are:

- DVB Subtitles.
- Teletext.

See clause 13.5, "Subtitles".

In the event that both DVB Subtitles and DVB Teletext are available then DVB Subtitles will take precedence (i.e. if a stream is flagged as having both DVB Subtitles and Teletext Subtitles then the DVB Subtitles will be displayed).

Teletext Subtitles conform to the same display model as DVB subtitles.

Application control and detection of subtitles, whether they be DVB Subtitles or Teletext Subtitles, will be through JMF. The application will have no knowledge of the delivery/presentation protocol being used to provide subtitles.

No APIs will be provided to access Teletext data packets and no timing model is provided for the decoding of Teletext subtitle. Text subtitles will be decoded as soon as the data becomes available.

#### 7.2.3.1 DVB Subtitles

DVB Subtitles are defined as in EN 300 743 [i.7].

#### 7.2.3.2 Teletext

Transmission of the text is as defined in EN 300 472 [i.6]. The data format is as defined in ETS 300 706 [56] but restricted to presentation level 1,5 or lower. Signalling of the Teletext subtitle page will be via the Teletext descriptor as defined in EN 300 468 [14].

Within the GEM specification Teletext is only supported as an alternative content format for delivery of subtitles. The GEM specification does not address its possible use as a navigable content format.

NOTE: Manufactures remain free to implement full Teletext support based on regulatory requirement or market demand. Such support would be implemented outside of the GEM environment, by VBI re-insertion of the non-subtitle text or through a native Teletext Decoder. The user interface integration is then an issue for the manufacturer to resolve.

It is envisaged that broadcasters will use GEM applications to deliver navigable text services providing a greater level of interactivity and enhanced graphics.

### 7.2.4 Containers

A Container is what in MPEG is called the “system layer”, i.e. a means for multiplexing and synchronizing multiple media streams with associated system information (metadata).

Examples of widely used containers are MPEG-2 TS on the broadcast side and MP4 on the internet for streaming purposes.

At least one container format shall be specified in a GEM terminal specification.

### 7.2.5 Streaming Manifest

A “streaming manifest” is a file, which can be used to tell the receiver that several alternative representations are available on the server for the same content, e.g. more files corresponding to encodings at different bit rates of the same content which the receiver can seamlessly switch to based on varying network conditions.

A GEM terminal, which supports adaptive streaming, SHALL select or define at least one file format for the straming manifest. URLs which reference manifest files whose formats are supported by the implementation SHALL be accepted when creating a JMF player.

If a streaming manifest file is referenced in the URL used by an application to create a JMF player which does support that particular manifest format, the JMF player will perform all the relevant operations (manifest parsing, switching from one representation to another based on certain conditions, …) without any application involvement.

GEM does not put any restrictions on the format of the streaming manifest.

## 7.3 Resident fonts

The inclusion of resident fonts is not required for GEM terminal specifications for packaged media targets.

This clause shall apply to GEM terminal specifications that include the functional equivalent named "Resident Fonts" as specified in clause 15.6, "Functional equivalents".

For GEM terminal specifications that do not include the resident font there shall be signalling to indicate a font file packaged with an application that will be available under the logical name "SansSerif", e.g. returned by `java.awt.Toolkit.getFontList()` or provided to the `java.awt.Font` constructor.

NOTE: `java.awt` requires that a default font be available, but does not specify what it is. It is recommended that applications explicitly specify a known font for consistent results.

See clause G.4, "Resident fonts and text rendering".

See also annex D, "Text presentation".

## 7.4 Downloadable fonts

A GEM terminal specification may define a way to check the name, style and size parameters of a `FontFactory.createFont()` call on validity with respect to the data present in the font file from which the font should be created.

A GEM terminal specification may require this check for a `FontFactory` constructed for a single font file.

A GEM terminal specification may require this check for `FontFactory` constructed for a font index file in addition to the check for support of the given name, style and size in the given font index file.

If such a check is not defined for a GEM terminal specification then the parameters used in a `createFont()` call shall not be checked against data present in the font file.

## 7.4.1 PFR

Use of the PFR downloadable font format as defined here is not required for GEM terminal specifications for packaged media targets.

This clause shall apply to GEM terminal specifications that include the functional equivalent named "Downloadable Fonts" as specified in clause 15.6, "Functional equivalents". GEM terminal specifications that do not include this definition shall specify a functional equivalent that enables a font to be packaged with GEM applications. This clause is an example of such a format, and is informative for these GEM terminal specifications.

PFR0 (Portable Font Resource version 0) is defined as in DAVIC 1.4.1p9 [17] as the coding format for fonts. Receivers implementing a GEM specification that requires PFR support are only required to provide support for the outline version of the font.

The `charCode` value in the PFR `charRecord` and `bmapCharRecord`, and the `charCode1` and `charCode2` fields of the pairKernData() structure, shall be the ISO/IEC 10646-1 [15] code for the glyph. If these fields are 16-bit, they are encoded using UCS-2 and shall not be a high-surrogate or a low-surrogate code value. If these fields are 8-bit, then the one-byte character code shall be the least significant 8 bits of a 16-bit UCS-2 code, where the most significant 8 bits are all zero.

For fonts in the PFR format, the font name shall be the `fontID` field in the font file. The `fontID` field in the font file shall be encoded using the Java modified UTF-8 encoding (see clause 7.1.5, "Monomedia format for text").

Each font in a PFR file can have auxiliary data in its physical font record. If present this auxiliary data shall adhere to the syntax as specified in table 3. It shall consist of a number of blocks terminated by a block of length 0 and type 0.

**Table 3: PFR auxiliary data**

| | No.of Bits | Identifier |
|---|---|---|
| auxDataFontRecord() { | | |
| do { | | |
| blockLength | 16 | uimsbf |
| blockType | 16 | uimsbf |
| switch (blockType) { | | |
| case 2: | | |
| for (i = 0; i < 10; i++) { | | |
| reserved | 8 | uimsbf |
| } | | |
| ascent | 16 | tcimsbf |
| descent | 16 | tcimsbf |
| reserved | 32 | uimsbf |
| externalLeading | 16 | tcimsbf |
| for (i = 0; i < (blockLength - 24); i++) { | | |
| reserved | 8 | uimsbf |
| } | | |
| break; | | |
| default: | | |
| for (i = 0; i < (blockLength - 4); i++) { | | |
| reserved | 8 | uimsbf |
| } | | |
| break; | | |
| } | | |
| } while (blockLength > 0 \|\| blockType != 0) | | |
| } | | |

An auxiliary data font record consists of a number of blocks terminated by a block of length 0 and type 0.

`blockLength`: The total number of bytes in this block, including the blockLength and the blockType.

`blockType`: A number that defines the type of data in the block. 0x0000 to 0x7fff are reserved, 0x8000 to 0xffff are user defined types.

`ascent`: Represents the font ascent, which is the distance from the base line to the top of most alphanumeric characters.

descent: Specifies the descent (units below the base line) of most alphanumeric characters.

externalLeading: Specifies the amount of extra leading (space) that the application adds between rows. The designer may set this member to zero.

GEM terminals shall ignore the contents of the reserved fields within block type 2 for the purposes of computing font metrics. GEM terminal behaviour for block types other than 2 is implementation dependent and these should not be used by GEM applications.

### 7.4.2 OpenType

OpenType is standardized in ISO/IEC 14496-18 [75] as the font format for MPEG-4, and is a full-featured font format that supports advanced typographic features, multi-lingual text processing and international character sets.

ISO/IEC 14496-18 [75] defines different MPEG text profiles and levels of functionality. GEM receivers shall support OpenType fonts with TrueType outlines, as defined by Advanced Simple Text Profile at Level 2 in ISO/IEC 14496-18 [75], section 6.2.5. OpenType fonts with TrueType outlines enable the highest quality of text rendering on low-resolution displays.

For OpenType fonts, the font name used by the GEM application shall be the full font name specified in the Naming ('name') table of OpenType font with the `NameID = 4`.

See also clause D.2.2, "Downloaded fonts".

## 7.5 Colour representation

### 7.5.1 Background (informative)

The method of colour encoding is critical to how consistently the colours in an image can be reproduced across different systems. The description must be cast in a way which is independent of the mechanisms by which it will finally be reproduced for the viewer.

The International Colour Consortium (ICC) has proposed a thorough solution to the precise communication of colour in open systems. However the ICC profile format is somewhat over-specified for GEM. The ICC mechanism for ensuring that a colour is correctly mapped from an input to the output colour space is by attaching a profile for the input colour space to the image in question. This is appropriate for high end systems, especially those in the print media. However, a primarily CRT based home platform neither needs, nor has the processing power and available bandwidth, to handle an embedded profile mechanism. It would also require some sophistication on the part of the end consumer to set up properly.

Fortunately by adopting a single default colour space that can be processed as an implicit ICC profile the advantages of the ICC approach are gained, and the system is later scalable to a full colour management system with a clear relationship to existing ICC colour management systems while minimizing software and support requirements in a GEM terminal today.

A colour space is a model for representing colour numerically in terms of three or more coordinates or tristimulus values. An RGB colour space represents colours in terms of Red, Green and Blue coordinates. The MHP format shall use the specific RGB encoding for colour imagery, sRGB as defined in IEC 61966-2-1 [39]. This is suitable for a wide range of presentation environments including TV's and has become widely adopted in the computer environment and WWW. It is, for example, compatible with CCIR Recommendation ITU-R Recommendation BT.709-5 [41] standard for colour encoding in HDTV. This format has the advantage of device independence without a great deal of additional overhead.

For sRGB, the goal is to communicate the appearance of colours as displayed on a reference monitor in terms of 8-bit digital code values for each coordinate. sRGB colour values represent colour appearance with respect to a defined reference viewing environment.

For colour stimuli viewed in the reference viewing environment, sRGB values are defined by a series of simple mathematical operations from standard CIE colourimetric values.

The sRGB format is a good match for 24 bit colour on most CRTs. In devices where a great deal of damage is done to the colour space it may not give consistent results. For example dithering to a 4-bit per primary colour map will violate the gamma assumptions.

## 7.5.2 Specification

All images transmitted shall be within the gamut encompassed by the sRGB colourspace. Where possible this should be coded so that the terminal does not have to translate. Where this is impractical the sRGB image may be transcoded into a different colourspace provided the gamut assumption is not violated (i.e. to be consistent with JFIF, JPEG images shall be sent in the region of the $YC_rC_b$ colourspace that overlaps with the sRGB gamut).

NOTE: That the presentation of images using colours outside of the sRGB gamut is platform dependent.

Images created in GEM will be in the sRGB colourspace by default, although manufacturers are free to provide support for other colour spaces if they choose. All MHPs shall support transformations from sRGB to the colour spaces allowed by the MPEG-2 definition (e.g. ITU-R Recommeandations BT.709-5 [41] and BT.470-6 [40]) and vice versa, manufacturers may choose to support transformations to and from other colour spaces.

### 7.5.2.1 The sRGB Reference Viewing Environment

The reference display conditions and viewing environment for sRGB are partly described in table 4. A reference viewing environment must be provided to allow for the unambiguous definition of colour, the sRGB reference viewing environment corresponds to conditions typical of indoor viewing of CRTs - further details can be found in the IEC 61966-2-1 [39].

The sRGB reference conditions therefore provides a well defined reference compatible with ITU-R Recommendation BT.709-5 [41].

**Table 4: sRGB reference Display conditions**

| Condition | sRGB |
|---|---|
| Viewing flare | 1,0 % |
| Reference Background | 20 % |
| Display model Offset | 0,055 |
| Display Gun/Phosphor Gamma | 2,4 |
| Display white point | x = 0,3127 y = 0,3290 (D65 Hunt, R.W.G. [27]) |
| Ambient Lighting | 64 lx |
| Display Luminance level | 80 $cd/m^2$ |

### 7.5.2.2 Colourimetric Definitions and Encodings

sRGB tristimulus values can be computed as follows, firstly linear sRGB tristimulus are computed as linear combinations of the 1931 CIE XYZ (CIE 15 [24]) values using the following relationship.

$$\begin{bmatrix} R_{sRGB} \\ G_{sRGB} \\ B_{sRGB} \end{bmatrix} = \begin{bmatrix} 3,2410 & -1,5374 & -0,4986 \\ -0,9682 & 1,8760 & -0,0416 \\ 0,0556 & -0,2040 & -1,0570 \end{bmatrix} \begin{bmatrix} X_{D65} \\ Y_{D65} \\ Z_{D65} \end{bmatrix}$$

In the encoding process, negative sRGB tristimulus values, and sRGB tristimulus values greater than 1,00 are not retained. The luminance dynamic range and colour gamut of sRGB is limited to the tristimulus values between 0,0 and 1,0 by simple clipping. This gamut, however, is large enough to encompass most colours that can be displayed on CRT monitors.

For comparison, the CIE chromaticities for the red, green, and blue ITU-R Recommendation BT.709-5 [41] and ITU-R Recommendation BT.470-6 [40] reference primaries, and for CIE Standard Illuminant D65 (IEC 61966-2-1 [39]), are given in tables 5 and 6. From these primaries the $YC_bC_r$ transmitted values are computed by similar relationships.

Therefore ITU-R Recommendation BT.709-5 [41] $YC_bC_r$ colourspace and similar video colour spaces can be converted to sRGB and vice versa by way of CIE XYZ (CIE 15 [24]). Chromaticities for other video formats allowed in MPEG streams can be found in their respective standards.

**Table 5: ITU-R Recommendation BT.709-5 [41] reference primaries and CIE standard illuminant**

| | Red | Green | Blue | D65 |
|---|---|---|---|---|
| **x** | 0,6400 | 0,3000 | 0,1500 | 0,3127 |
| **y** | 0,3300 | 0,6000 | 0,0600 | 0,3290 |
| **z** | 0,0300 | 0,1000 | 0,7900 | 0,3583 |

**Table 6: ITU-R Recommendation BT.470-6 [40] reference primaries and CIE standard illuminant**

| | Red | Green | Blue | D65 |
|---|---|---|---|---|
| **x** | 0,6700 | 0,2100 | 0,1400 | 0,3100 |
| **y** | 0,3300 | 0,7100 | 0,0800 | 0,3160 |
| **z** | 0,0100 | 0,0800 | 0,7800 | 0,3740 |

The linear sRGB tristimulus values are next transformed to non linear sR'G'B' values. This process closely approximates the effect of a "gamma" curve of 2,2 with a slight offset. This makes sRGB consistent with legacy systems and images.

if $R_{sRGB} > 0{,}00304$ and $G_{sRGB} > 0{,}00304$ and $B_{sRGB} > 0{,}00304$

then

$R'_{sRGB} = 1{,}055 \times R_{sRGB} \text{ ^ } (1{,}0/2{,}4) - 0{,}055$

$G'_{sRGB} = 1{,}055 \times G_{sRGB} \text{ ^ } (1{,}0/2{,}4) - 0{,}055$

$B'_{sRGB} = 1{,}055 \times B_{sRGB} \text{ ^ } (1{,}0/2{,}4) - 0{,}055$

else

$R'_{sRGB} = 12{,}92 \times R_{sRGB}$

$G'_{sRGB} = 12{,}92 \times G_{sRGB}$

$B'_{sRGB} = 12{,}92 \times B_{sRGB}$

end if

Finally, the non-linear sR'G'B' values are converted to digital code values. This conversion scales the sR'G'B' values by using the equation below where WDC represents the white digital count and KDC represents the black digital count.

- $R_{8bit} = ((WDC - KDC) \times R'_{sRGB}) + KDC.$
- $G_{8bit} = ((WDC - KDC) \times G'_{sRGB}) + KDC.$
- $B_{8bit} = ((WDC - KDC) \times B'_{sRGB}) + KDC.$

The current sRGB specification uses a black digital count of 0 and a white digital count of 255 for 24-bit (8-bits/channel) encoding, and the MHP shall adopt the same convention.

NOTE: Some digital video signals may use a black digital count of 16 and a white digital count of 235 in order to provide a larger encoded colour gamut.

Details of the reverse transformation from sRGB to CIE XYZ are given in IEC 61966-2-1 [39], mappings from ITU-R Recommendation BT.709-5 [41] and ITU-R Recommendation BT.470-6 [40] to CIE XYZ are given in ISO/IEC 13818-2 [37].

# 7.6 MIME types

**Table 7: File type identification**

| MIME type | Extension (note) | Definition of content |
|---|---|---|
| "image/jpeg" | ".jpg" | As defined in clause 7.1.1.2, "JPEG". |
| "image/png" | ".png" | As defined in clause 7.1.1.3, "PNG" possibly with a constrained profile as defined in clause 15.1, "PNG - restrictions". |
| "image/gif" | ".gif" | As defined in clause 7.1.1.4, "GIF". |
| "image/mpeg" | ".mpg" | As defined in clause 7.1.2, "MPEG-2 I-Frames". |
| "video/mpeg" | ".mpg" | As defined in clause 7.2.2, "Video". |
| "video/vnd.dvb.mpeg.drip" | ".drip" | As defined in clause 7.1.3, "MPEG-2 Video "drips"". |
| "audio/mpeg" | ".mp2" | As defined in clause 7.1.4, "Monomedia format for audio clips" or as defined in clause 7.2.1, "Audio". |
| "text/vnd.dvb.utf8" | ".txt" | As defined in clause 7.1.5, "Monomedia format for text". |
| "text/xml" | ".xml" | Not defined in the present document. |
| "text/css" | ".css" | Not defined in the present document. |
| "text/plain" | ".txt" | As defined in clause 7.1.5, "Monomedia format for text". |
| "text/ecmascript" | ".js" | Not defined in the present document. |
| "image/vnd.dvb.subtitle" | ".sub" | As defined in clause 7.2.3, "Subtitles". |
| "text/vnd.dvb.subtitle" | | |
| "text/vnd.dvb.teletext" | ".tlx" | As defined in clause 7.2.3.2, "Teletext". |
| "application/vnd.dvb.pfr" | ".pfr" | As defined in clause 7.4, "Downloadable fonts". |
| "application/vnd.dvbj" | ".class" | A DVB-J class file. |
| “application/vnd.dvb.ait” | “.ait” | AIT File as defined in clause 10.4.5, “AIT File” |
| “application/vnd.dvb.ait+xml” | “.aitx” | XML encoding of the AIT as defined in Annex AR, “XML encoding for AIT” |
| "multipart/vnd.dvb.service" | ".svc" | An MPEG Program (DVB Service) conforming to DVB norms. |
| NOTE: Future formats may use more characters in the extension. | | |

GEM terminal specifications may replace the entry "application/vnd.dvb.pfr" with "application/font-tdpfr" as defined in RFC 3073 [12] if they wish.

Earlier versions of GEM and derived specifications defined the DVB MIME types without the vnd. prefix. These type names are deprecated and their use is strongly discouraged. If a GEM terminal gets a deprecated MIME type, it shall behave, as if the vnd. prefix was present.

NOTE: The entries for "image/vnd.dvb.subtitle", "text/vnd.dvb.subtitle", "text/vnd.dvb.teletext" and "multipart/vnd.dvb.service" refer to content types for which support is not required by the present document.

## 7.6.1 Rationale

The MIME types are defined to reserve a name space for the possible future support of downloadable JMF players.

The file name extensions shall be included in broadcasts to assist receivers identify the type of the content. For DVB-J applications, this is described in table 30, "Return types of URL.getContent()".

Not all MIME and filename extensions defined in the above table are actually used in the present document. For DVB-J, the APIs which consume media types are described in clause 15.2, "Minimum media formats supported by DVB-J APIs". With MIME types whose corresponding media is not listed for a particular profile, access to that content type from files is not defined for that profile.

# 8 Void

# 9 Application model

## 9.1 Service-bound GEM applications

### 9.1.1 Basic lifecycle control

The basic control of the lifecycle of service-bound GEM applications is through the selection of services. Selection of a service can be initiated by the user of the GEM terminal using the navigator as well as by GEM applications offering service selection functionality. For broadcast and IPTV applications, service selection functionality usually takes the form of an EPG; EPG functionality is not typical for packaged media.

The unit for the presentation and execution of content in the GEM specification is the service. A service in GEM represents a group of pieces of content which are intended to be presented together to the end-user. A service may consist of the contents of a service, including audio/video streams, data streams and all the service information, applications and application signalling that is being broadcast. For packaged media targets, a service is made up of elements that are typically stored on a storage medium, including audio/video streams, data streams, service information, applications and application signalling. Other forms of service are possible, such as a stored service (see clause 9.6.2, "Stored services").

Every service that gets presented by a GEM platform is presented within a service context. These form one of the foundations for the runtime environment and the execution model. A service context is an "environment" in which a service gets presented. It defines the boundaries of the service (letting the platform and applications identify which of the pieces of content that are being presented make up a given service). It also enables that service to be addressed and controlled as a single entity.

In a DVB-J application, a service context is represented by an instance of the `ServiceContext` class. Where multiple DVB-J applications are being presented in the same service context, the number of `ServiceContext` objects representing a service context is implementation dependent, but each application sees only one such instance. Changes made by one application to the `ServiceContext` object that it has are visible to the `ServiceContext` objects representing the same service context in other applications. DVB-J applications may obtain a reference to the service context within which they are executing through using the method `getServiceContext(XletContext)` on the `ServiceContextFactory` class.

The means by which the navigator of a GEM terminal supports selection of services is implementation dependent. However, where a GEM application is using the numeric keys of the remote control, the navigator shall not respond to the user pressing these keys by causing service selection. Hence the user pressing the numeric keys to enter his pincode does not cause service selection.

A service context can present only one service at any one time. Selecting a service to be presented in a service context causes any previous service being presented in that service context to stop being presented. Any content part of the previous service which is not part of the new service shall stop being presented. GEM terminals may limit the number of broadcast service contexts which can be presented simultaneously.

- In a DVB-J application, selecting a service corresponds to calling the `select()` method on an instance of `javax.tv.service.selection.ServiceContext`.
- Host Control tune requests from a CI module cause service selections. Host Control replace / clear_replace has an equivalent effect to using `javax.tv.media.MediaSelectControl`.

Support for host control tune requests is not mandatory in the present document, thus the language in this clause relating to these tune requests only applies if such control is present in the terminal specification.

### 9.1.2 Starting applications

When a service is selected, applications which are listed in the Application Description of the service and identified as auto-start shall be launched as described in clause 10.4.3.1 without explicit intervention of the user. Applications which are started after the selection of a service will be controlled by signalling associated with that service. The GEM

terminal shall monitor that signalling for changes made by the broadcaster. These changes may include the termination of particular applications as well as the addition of new auto-start applications.

Applications which are not identified as auto-start in the Application Description shall not be automatically launched by the GEM terminal, but require explicit launching. This explicit launching can be done by the resident navigator on the GEM terminal or by a GEM application. For example, the user can launch such applications after they have been offered a choice of applications through some user interface. Since the resident navigator is not required to provide a mechanism for this explicit launching, services wishing to have applications started must provide an application which is identified as auto-start.

Where the currently selected service in a service context includes multiple GEM applications, any running applications may be able to launch other applications from that set. The launched applications shall be presented inside that same service context.

A DVB-J application is able to achieve this using the application listing and launching API.

## 9.1.3 Support for execution of multiple simultaneous applications

The set of applications that are signalled within a service can be presented and executed concurrently.

GEM terminals shall be able to support applications from that set (using the same screen) at least as defined in annex G, "Minimum platform capabilities". GEM terminals are required to support execution of the set of such applications for each service which they permit to be presented simultaneously.

Broadcasters should ensure that simultaneous running of the set of applications for a service is comprehensible to the user and does not yield perceptible interference problems.

## 9.1.4 Stopping applications

GEM applications may stop themselves voluntarily using the GEM APIs or may be stopped by the GEM terminal in a number of situations. Examples of situations where this shall be allowed include:

### 9.1.4.1 A new service being selected replacing a previously selected one

When a new service is selected and replaces a previously selected one, applications from the former service shall only continue to execute where they are signalled in the new service. If an application is not signalled in that signalling then it will be stopped by the GEM terminal. Where an application is known to be bound to a single service, the broadcaster can identify that application as service bound using the `service_bound_flag` in the Application Description. Such applications shall be stopped as soon as possible by the GEM terminal and without needing the Application Description for the new service to be available. This allows the autostart application(s) of the new service to be started earlier than would otherwise be the case.

### 9.1.4.2 The stopping of an application by another application

Subject to the security policy of a GEM terminal, one application may request to stop another application. In such a case, the resident Application Manager, after a successful security check, kills the application; otherwise that application shall continue running, without interruption.

### 9.1.4.3 Changes in the application signalling to request a particular application be stopped

The broadcaster may request a GEM terminal to stop an application using the control codes in the Application Description. The precise semantics of these are dependent on the application format.

### 9.1.4.4 Stopping by the GEM terminal due to a shortage of resources

Where a GEM terminal has insufficient resources (e.g. memory, CPU and resources managed by the resource management API) to continue the execution of one of the running applications, the GEM terminal is allowed to decide to stop a GEM application without user intervention.

NOTE: The precise resources of a GEM terminal are implementation dependent.

### 9.1.5 Persistence of Applications Across Service Boundaries

Where a running application is signalled in both the new service and the former service, and is not signalled as service bound in the former service, it shall continue to run and shall not be restarted. In this case, the running application shall become controlled by the application signalling of the new service where it is signalled and not the signalling of the former service. Hence the GEM terminal shall monitor the Application Description of the new service and shall stop responding to the Application Description of the former service.

If the application is signalled as service bound in the former service then it is terminated in the normal way as the new service is selected. If it is signalled as auto-start in the new service it will restart with no volatile context from the previous instantiation.

If an application survives a service selection operation, it will not automatically gain access to any new broadcast file system on the new service. It remains logically attached to any broadcast file systems it has already attached to, although it may be temporarily disconnected from the broadcast carousels feeding those file systems. The behaviour of the broadcast file system under these circumstances is defined in clause 6.2.5.3, "Loss of carousel behaviour".

If temporarily disconnected, the broadcast carousel shall be reconnected upon selection of a service where an identical carousel is available. This includes both the original service from which the application was downloaded (assuming the carousel is still present in that service) and other services containing an identical carousel.

### 9.1.6 Management of autostarting

The receiver shall launch autostart applications under the following conditions:

- The signalling indicates that the application can be supported by the receiver, as defined by the application profile and versioning information contained in the Application Description.
- Only a single application with a given Application identification is allowed to run at any time in a single service context.
- The application is a newly introduced autostart application or has newly been given autostart status.

So:

- When a service is selected the receiver shall launch at most one instance of each autostart application that it can support.
- If after service selection an autostart application that the receiver can support is introduced or a previously listed supportable application gains autostart status then the application shall be launched subject to normal resource limitations, etc.

However, if an autostart application terminates itself, it shall not be restarted unless it again becomes a new autostart application. An application becomes a new autostart application in the following cases:

- The receiver navigates away from the service and then selects a service where the application is autostart.
- The application is removed from the Application Description and then is re-introduced.
- The autostart status of the application is reset then set again.

NOTE 1: In summary the autostart status of an application is in effect an edge trigger rather than level trigger signal.

NOTE 2: These semantics for the autostart behaviour address "de-bouncing" the case where an autostart application terminates voluntarily. They do not address the case where the receiver terminates the application.

In this case the platform may attempt to re-start the application however this is implementation dependent.

NOTE 3: The present document does not describe in detail the timing required for the signalling to renew the autostart status of an application.

### 9.1.7 Tuning without service selection

GEM applications may cause tuning to another transport stream by mechanisms other than service selection. Usage of these mechanisms does not constitute service selection and therefore no applications from the target transport stream or service shall be started either by the GEM terminal or by GEM applications. The GEM terminal shall continue monitoring the Application Description of the logically selected service where this is available on the target transport stream. Where the Application Description of the selected service is not available, the application shall continue executing as described in clause 10.4.2, "Visibility of Application Description and tuning". The service being presented in the service context shall not be changed by an application using these mechanisms:

- DVB-J tuning APIs.

This clause only applies to GEM terminal specifications with a tuner.

### 9.1.8 GEM Applications and Service Selection

GEM applications may select services using the service selection API. The service selection API includes a class `ServiceContext` to represent environments in which services may be presented. Calling the `select()` method on a `ServiceContext` causes a new service to be presented in that context and any former service being presented in that context will be stopped.

Where one GEM application uses the application listing and launching API to successfully start a second GEM application, the second GEM application shall be considered as executing inside the service context of the first GEM application. The number of `ServiceContext` objects representing a service context is implementation dependent, but each application sees only one such instance. Changes made by one application to its associated `ServiceContext` object are visible to the `ServiceContext` objects representing the same service context in other applications.

GEM applications started in response to a service selection operation are considered to be executing "inside" a service context. They may obtain a reference to the service context within which they are executing through using the method `getServiceContext(XletContext)` on `javax.tv.service.selection.ServiceContextFactory`.

### 9.1.9 Cached applications

When an application is signalled in the Application Description of a service in the usual way, and that Application Description entry also contains an `application_storage_descriptor`, the terminal may use files that have been stored on the terminal to accelerate the loading of that application. These files may have been stored on the terminal by any way allowed by clause 9.9, "Stored and cached applications", including proactive caching, or an explicit request to store or cache the application.

Stored broadcast related applications shall take the lifecycle model of service-bound applications. For example:

- The application can only be started when it is listed in the Application Description of the currently selected service.
- Following a service change, the application behaviour will depend on the Application Description signalling in the newly selected service.

In summary, the behaviour is that of a service-bound application, except that the loading is possibly accelerated due to the file system cache provided by the storage.

When a cached application (other than one with `launchable_completely_from_cache` set to "1") executes, it shall see the same filesystem as if it was being run directly from broadcast. The set of files stored in response to being listed in the Application Description file (see clause 9.6.1, "Applications loaded from the interaction channel") shall always override the same files from the broadcast.

Broadcasters should not remove a file listed in the ADF from the broadcast without amending the ADF and incrementing the application version number. If a cached file is removed from the broadcast, but is cached by the terminal, it shall still be readable. `File.list()` shall always return the current contents of the broadcast, so will not show such files.

When starting an application which has `launchable_completely_from_cache` set to "1", the behaviour depends on whether or not all critical files are cached. If they are not, the application will be started as described in the previous

paragraph (or, if "not_launchable_from_broadcast" is "1", the application will not be started at all). If all critical files are cached, the application will be started without mounting the carousel. Only the files stored in the cache shall be available. Note that, in this case, if the permission request file is not listed in the application description file (or does not have a priority of critical and has not been cached), the permission request file is missing and hence no additional permissions are requested.

NOTE: The application provider is responsible for managing any dependencies between those files which can be stored and those files which are not required to be stored.

#### 9.1.9.1 Version management

If the application has not_launchable_from_broadcast set to '0' then the service-bound version shall be used regardless of the value of is_launchable_with_older_version. If the application has not_launchable_from_broadcast set to '1' and:

- is_launchable_with_older_version is set to '0', it shall not be launched at all.
- is_launchable_with_older_version is set to '1', the cached/stored version shall start.

Applications that use the application caching API are responsible for updating the cached version of an application when a new version is released. If not_launchable_from_broadcast is set to '1' and:

- is_launchable_with_older_version set to '0', a launchable application must be provided to manage the update (although a GEM terminal may perform the update autonomously).
- is_launchable_with_older_version set to '1', the cached/stored application is responsible to manage the update.

Terminals shall allow different versions of an application to be cached/stored simultaneously. (For example, if two broadcasters use the same application, they may not upgrade to the latest version at the same time.) Where multiple versions of an application are stored and one of them is to be started, the one started shall be the one whose version number is less than the version number of the transmitted application by the smallest amount.

NOTE: GEM terminals may refuse to update, or keep stored, applications where the version numbers change too frequently. Broadcasters should ensure that frequently changing files are not listed in the "Application description file". The meaning of "too frequently" is terminal-specific. However, an application version changing twice a day is not expected to happen too frequently.

When running a cached application where multiple versions of that application are cached, the terminal shall only return files from the cache that were stored in response to an application description file signalled with the same organisation ID, application ID, and version number as the version of the application that is being run.

#### 9.1.9.2 Proactive caching

If a storable application is signalled, terminals are allowed to proactively store any files that are indicated in the ADF. Terminals do not have to honour the requested priority when proactively caching files.

In particular, note that proactive caching is not required to store all files with critical priority (but in this case the application shall not be reported as cached by the GEM APIs, and shall not be started if it is signalled with not_launchable_from_broadcast set to "1").

Terminals are responsible for handling version updates to applications that were proactively cached.

## 9.2 DVB-J Model

GEM terminal specifications based on the packaged media profile or the IPTV profile may not support the dynamic update of a service's Application Description. In such a case, corresponding language in clause 9 and its subclauses does not apply.

### 9.2.1 Starting DVB-J Applications

DVB-J applications may be started by any of the means defined for general GEM applications. The application listing and launching API defined in annex S, "Application listing and launching" allows one GEM application to start another GEM application subject to security policy. The `start()` method of the `AppProxy` interface will then cause the Application Manager to start the new GEM application subject to normal resource limitations.

The Xlet interface is defined in the `javax.tv.xlet.Xlet` interface Java TV [16]. DVB-J applications provide a class implementing this interface and reference that class in the DVB-J application location descriptor. In order to start a DVB-J application, the application manager shall call the constructor of this class, the `initXlet()` and the `startXlet()` methods of this interface.

### 9.2.2 Stopping a DVB-J Application

DVB-J applications may stop for any of the reasons listed for general GEM applications. An application shall be able to notify that it is stopped by finishing its execution and informing the Application Manager through the `notifyDestroyed()` method on the `javax.tv.xlet.XletContext` interface. This interface also includes other methods to allow a DVB-J application to request or notify changes in its state.

The application listing and launching API allows an application to indirectly control the lifecycle of another application subject to security policy. This control is indirect because an application cannot invoke an Xlet state method directly, but goes through this API. This ensures that the resident Application Manager can always keep track of all the applications that are running.

When a DVB-J application is stopped by a GEM terminal, the `destroyXlet` method of the signalled Java class implementing the Xlet interface, i.e. the initial class of the application, shall be called by the application manager. In the case of the application being stopped due to a service selection operation, the stopping of the application shall be unconditional. This method call gives applications their last opportunity to save state before their execution stops. Applications which wish to survive the user of the GEM terminal zapping away from their service (e.g. during an advertising break) must save their state and reload that state when they are re-started if the user returns to that service later.

### 9.2.3 DVB-J Application Lifecycle

#### 9.2.3.1 Introduction

This clause describes the Xlet lifecycle model for the DVB-J API. This describes the capabilities of the Xlet in each state and the methods by which the application manager influences the life cycle state.This clause is not directly related to other aspects of a system, such as graphics or shared resource allocation/management.

NOTE: The Java platforms define a number of application models that have their own lifecycles associated with them. In general, they are designed to address specific issues on that platform. For instance, the Applet was designed to provide support for executable content in web pages. However, none of the existing application technology fully addresses the specific requirements of television receivers. The application lifecycle defined in this clause is meant to be compatible with existing Java platforms and virtual machine technology.

This clause defines the lifecycle of an instance of a DVB-J application. The lifecycle of a DVB-J application and of an application instance are the same except that when an application instance is destroyed, the application is only transiently destroyed and then becomes not loaded. See `org.dvb.application.AppProxy.NOT_LOADED` in annex S, "Application listing and launching".

#### 9.2.3.2 Lifecycle state machine for DVB-J application instances

The Xlet state machine ensures that the behaviour of an Xlet is close to the behaviour that television viewers expect, specifically:

- The perceived start-up latency of an Xlet can be very short.
- It is possible to put an Xlet into a state where it is not providing its service.

- It is possible to destroy an Xlet at any time.

The figure 13 shows the state machine model for Xlets. The Xlet states are defined in more detail in table 8.

The different influences that can cause an Xlet to change state include:

- The application manager uses the Xlet API to signal these changes to the Xlet.

  Various factors may stimulate the application manager to act in this way, for example:

  - Broadcast signalling (e.g. a change in the state of the application_control_code parameter carried by the Application Description (see clause 10.4, "Application Description")).

  - User selection of an application in a host provided UI.

- The Xlet itself "decides" to change state.

  The application uses the `XletContext` Object to communicate or request such changes to the application manager.

- Another Xlet acts via the application launching API (see clause 11.7.2, "Application discovery and launching APIs").

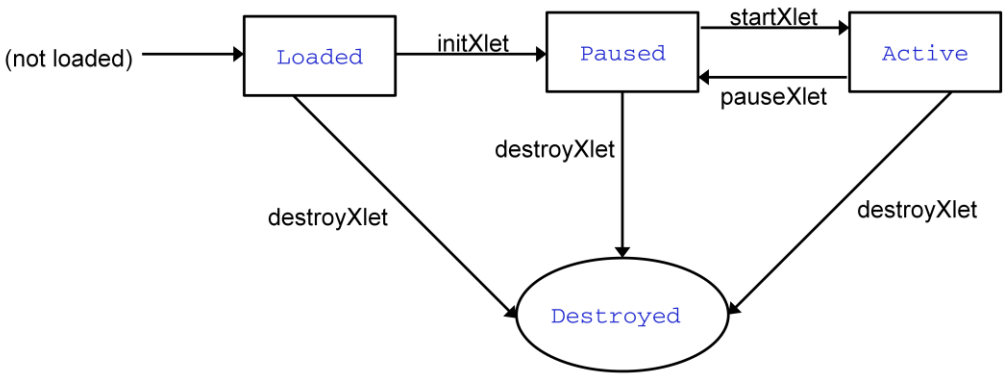



**Figure 13: Xlet lifecycle state machine diagram**

**Table 8: Valid lifecycle states of DVB-J application instances**

| State Name | Description |
|---|---|
| `Loaded` | The DVB-J application instance has been loaded and has not been initialized.<br><br>The signalled Java class used to initiate a DVB-J application instance must implement the `javax.tv.xlet.Xlet` interface. Otherwise, the class (and hence the application instance) may be ignored. An instance of the signalled Java class is created by the application manager e.g. using the `Class.newInstance` method. Therefore a DVB-J application instance must have a public "default constructor". Otherwise, the class (and hence the application instance) shall immediately enter the `Destroyed` state. If the default constructor returns without throwing an uncaught exception, then the application instance is considered to be "`Loaded`", otherwise the application instance immediately enters the `Destroyed` state and is discarded. Once the application instance has been successfully loaded and instantiated, the application manager can transition the application instance to the Paused state by invoking the `initXlet` method on the signalled class (implementing the Xlet interface).<br><br>If the `initXlet()` method throws an `XletStateChangeException` then the application instance shall remain in the Loaded state. The only possible state transition for such an application instance is into the destroyed state. The application instance can request this itself or wait for the application manager to cause this transition.<br>(See notes 1 and 2.) |
| `Paused` | A Paused DVB-J application instance should minimize its usage of resources if it wants to maximize its probability of survival. This does not imply that it cannot be holding any resources, but in such a case, it would have a lower priority as concerns access to resources than it had when it was in the `Active` state.<br><br>When entering the paused state, an application should ensure that any `HScene` instances it is using are invisible. See also clause 9.10, "Lifecycle interactions between GEM and resident applications".<br><br>While an application is paused, it should not attempt to make any `HScene` instances visible, and it should not attempt to draw to the screen. Any application that does this while paused may be considered uncooperative, and may be terminated by the application manager. Applications should also respect the rules of "television viewing mode" as defined in clause 11.4.1.4.<br><br>This state is entered:<br>• From the Loaded state after the `Xlet.initXlet()` method returns successfully when invoked by the application manager (the first time). Other invocations of this method do not cause the change of state.<br>(See note 3)<br>• From the `Active` state after the `Xlet.pauseXlet()` method returns successfully when invoked by the application manager. Other invocations of this method do not cause the change of state.<br>• From the `Active` state upon entering the `XletContext.notifyPaused()` method. |
| `Active` | The DVB-J application instance is functioning normally and providing service.<br>This state is entered:<br>• From the Paused state after the Xlet.startXlet() method returns successfully. |
| `Destroyed` | The DVB-J application instance has released all of its resources and terminated.<br>This state is entered:<br>• When the Xlet's destroyXlet() method returns successfully. The destroyXlet() method shall release all resources held and perform any necessary clean up so the Xlet may be garbage collected.<br>• Upon entering the XletContext.notifyDestroyed() method. The Xlet performs its clean up actions before calling the notifyDestroyed() method.<br>(See note 4) |
| NOTE 1: Application initialization is intended to occur in the initXlet method, rather than in the default constructor.<br>NOTE 2: This state is entered only once per instance of an DVB-J application.<br>NOTE 3: The application manager shall only call initXlet() once per instance of a DVB-J application.<br>NOTE 4: This state is entered only once per instance of an DVB-J application instance. | |

Every time a DVB-J application instance is started (i.e. the constructor of the object implementing Xlet is called), it shall logically run in its own new virtual machine instance. See clause 11.2.1, "Basic Considerations".

Only the DVB-J application can determine if it is able to provide the service for which it was designed. As such, in some respects an application manager cannot guarantee whether an DVB-J application can, or is, providing its service; and application manager can only indicate that the DVB-J application is able to do so. A typical sequence of DVB-J application execution is:

| Application Manager | DVB-J application |
|---|---|
| The application manager creates a new instance of an Xlet. | The Xlet's default (no argument) constructor is called, it is in the `Loaded` state. |
| The application manager creates the necessary context object for the DVB-J application to run, and initializes the Xlet. | The DVB-J application uses the context object to initialize itself. It is now in the `Paused` state. |
| The application manager has decided that it is an appropriate time for the DVB-J application to perform its service, so it signals it to enter the `Active` state. | The DVB-J application acquires any resources it needs and begins to perform its service. |
| The application manager no longer needs the DVB-J application to perform its service, so it signals the DVB-J application to stop performing its service. | The DVB-J application stops performing its service and might choose to release some resources it currently holds. |
| The application manager has determined that the DVB-J application is no longer needed, or perhaps needs to make room for a higher priority application in memory, so it signals the DVB-J application that it is a candidate to be destroyed. | If it has been designed to do so, the DVB-J application saves state or user preferences and performs clean up. |

## 9.2.4 Xlet API

The Xlet API provides GEM application developers with an API that provides life cycle signalling. The Xlet API uses the callback approach to signal state changes.

This API is specified in clause 11.7.1, "APIs to support DVB-J application lifecycle".

### 9.2.4.1 Xlet State Change Semantics

An Xlet's state can change either by having one of the methods on its Xlet Interface called, or by making an internal state transition and notifying the application manager via the `XletContext` Object. The semantics of when that state change actually happens are important:

- Calls to `Xlet`: this interface indicates a successful state change only when the call successfully returns.
- Calls to `XletContext`: the `notifyDestroyed()` and `notifyPaused()` methods indicate a state change on entry. The `resumeRequest()` method indicates no state change, instead only a request to change state.

If a method on the Xlet interface throws an `XletStateChangeException`, the Xlet shall remain in the state it was in immediately prior to the call of the method throwing the exception unless otherwise specified. In the present document, the only exception to this rule is the `destroyXlet` method when the `unconditional` flag is `true` where throwing the `XletStateChangeException` is specified to have no effect. For the case of `initXlet` which may only be called once, the application manager may choose to transition the Xlet to the destroyed state (without calling `destroyXlet`) some implementation specific time later.

### 9.2.4.2 Xlet state change requests

The following table defines the previous states in which calls to the methods on `XletContext` relating to state management are valid;

NOTE: These methods are called after the Xlet has changed its state.

**Table 9: States for valid state management calls**

| Call | State |
|---|---|
| notifyDestroyed | all states |
| notifyPaused | active only |
| resumeRequest | paused only |

Calls to these methods when an Xlet is in any other state shall have no effect.

## 9.2.5 Multiple application environment support

The DVB-J platform allows for the simultaneous execution of several DVB-J applications.

Allowing several DVB-J applications to run simultaneously implies that some rules be defined for these DVB-J applications to share the resources of the MHP, and in particular for them to share the Input Focus and the Output Focus.

### 9.2.5.1 Control of DVB-J applications by other DVB-J applications

The present document provides support for control of the lifetime of a DVB-J application by another DVB-J application. This feature enables service providers to write their own "Launcher applications" that take care of the presentation to the user of the availability of DVB-J applications, and that enables eventually the user to launch DVB-J applications.

NOTE: The actual control of the lifetime of an DVB-J applications is done by the Application Manager only. The present document only provides APIs that enable DVB-J applications to ask the Application Manager to start, stop, pause and resume DVB-J applications.

See clause 11.7.2, "Application discovery and launching APIs".

### 9.2.5.2 Input Focus management

The input focus is defined as follows:

- The application that has input focus is in principle able to receive user-input events.
- Other applications not having the input focus can request to receive a subset of user-input events via a dedicated API. See clause 11.3.2.2, "org.dvb.event".

A DVB-J application has input focus if and only if the `java.awt.Component` having focus belongs to the component tree of that application.

### 9.2.5.3 Other resources management

The APIs defined in the present document provide support for resource allocation/revocation and resource revocation notification. The semantics of the APIs, however does not define under which circumstances an access to a resource is granted or revoked. While it is well understood that in most cases, it is up to the GEM implementation to define its own policy in terms of resource management, this clause defines the basic rules that a GEM implementation has to follow.

The GEM specification describes a multi-application environment. Hence several applications may be competing for access to the same atomic resource. The resource notification API described in clause 11.7.5, "Resource notification" provides a common way for applications to negotiate access to scarce atomic resources when such competition happens. This API allows for the GEM terminal to inform the application that currently holds the resource that another application wants to access this resource. It also provides a means for the owner of the resource and to the requester of the resource to communicate by private means. This private communication is reflected by the `request_data` object that the requester may pass to the owner. The semantics of this object is private and has to be known by both applications.

Some existing and general purposes java APIs that were developed before the GEM work was started do not use this general resource sharing mechanism. Hence access to resources addressed by these APIs are not subject to negotiation. For example, when an application holds a JMF player, if another application was to create a JMF player for the same content-type, the GEM terminal has to decide by itself whether it withdraws the resource underlying the JMF player from the current owner and grants it to the requester.

It is also possible for applications to use the inter-application communication API to establish private communication channels enabling them to negotiate access to resources.

As an optimisation, GEM implementations may make available additional resources (e.g. tuners) to GEM applications which do not ask for them explicitly. Any such additional resources must be successfully reserved on behalf of the application before use and released afterwards. If the reservation fails, the same resource shall be used as on a GEM implementation without those additional resources, e.g. the tuner associated with the service context in which the application is running. If the additional resources are later removed, the GEM implementation shall revert to using the expected resources (e.g. the tuner associated with the service context in which the application is running) transparently to the application.

#### 9.2.5.4 VM implementation

Where there are multiple DVB-J applications being executed as part of the same service, GEM terminals are allowed implement these in a single actual virtual machine instance. Regardless this shall conform to clause 11.2.1, "Basic Considerations".

## 9.3 Void

## 9.4 Inter-application resource management

Some downloaded resident applications specified as extensions to the present document may perform some of the functions of the navigator, e.g. the monitor application defined OCAP [5]. In this case, such downloaded software shall implement a policy that is consistent with the requirements of the present document.

### 9.4.1 Application instances running in the same service context

Where there is a resource conflict between two applications signalled as part of the same service and running in the same service context, this shall be resolved using the priority signalled in the `application_priority` field in the Application Description for each application. When comparing two applications, the one with the higher `application_priority` value shall be considered the more important to preserve.

- Where there is a resource conflict between two applications signalled as part of the same service and running in the same service context, this shall be resolved using the priority signalled in the `application_priority` field in the Application Description for each application or the external application authorization descriptor depending on which descriptor the application is currently signalled by.

- Unless stated otherwise, only the application which owns a resource has the right to modify the state / settings / configuration of that resource. If a resident application (e.g. the navigator) makes changes to the state / settings / configuration of a resource, the GEM terminal shall inform the GEM application which formerly owned the resource that the GEM application has lost ownership of the resource.

NOTE: The only exception to this rule is where all applications running inside a service context together own the service component handlers of that service context and not any single one of them. See clause 11.6.2, "Service selection API".

When a `ServiceContext` is in the presenting state, it shall keep reserved the resources needed to present the service currently being presented. In particular, when the service being presented is received via a tuner represented by a `org.davic.net.tuning.NetworkInterface`, that tuner shall be kept reserved. It is implementation dependent whether such tuners are reserved when not being used for the presentation of the currently selected service. Any resources reserved by the implementation of `ServiceContext` but not used for the presentation of the currently selected service shall be released if any other application requests them.

If an application running inside a `ServiceContext` attempts to reserve a resource being used for presenting the currently selected service in that `ServiceContext`, it shall be given a higher priority than the implementation of `ServiceContext`. e.g. an application running in a TV service running in a `ServiceContext` shall be able to reserve the `NetworkInterface` currently used to present that TV service and tune away to another transport stream.

### 9.4.2 Application instances not running in the same service context

The present document intentionally does not specify resource management policy or algorithms other than as above. The service level resource priority (as set by the `setPriority()` method on `UnboundApplicationService` or `DvbServiceContext.setPriority()`) is one input to the resource management algorithm. Another input may be the identity of the application or applications that the user is interacting with. How these inputs are used by the implementation resource management algorithm is not defined by the present document.

NOTE: Specifically the present document intentionally does not require support for the platform-wide application resource priority defined in clause 10.2.2.5 "Application Priority" of OCAP [5] or the policies and mechanisms defined in clause 19 of OCAP [5].

## 9.5 Void

## 9.6 Services and applications not related to conventional services

Clause 9.1, "Service-bound GEM applications" describes applications whose lifecycle is controlled through services. The present document additionally supports applications whose lifecycle is not bound to that of services. These include:

- Applications signalled through the interaction channel (see clause 9.6.1, "Applications loaded from the interaction channel").
- Applications running from storage in the GEM terminal (see clause 9.6.2, "Stored services").

### 9.6.1 Applications loaded from the interaction channel

In this scenario the application signalling comes from the interaction channel rather than the broadcast channel. In overview:

- Data equivalent to the service-bound Application Description is provided from the interaction channel in the AIT File. The connection to the server that provides the data is analogous to a connection to a service.
- A locator specifying the location of the AIT File is analogous to the locator specifying a service.
- Navigation to this locator is analogous to selection of a service. The service selection could either be from the GEM terminal's intrinsic navigator or via an application using the service selection API.
- As with navigation between services when an interaction channel locator is selected as the current service the GEM terminal retrieves and decodes the AIT File. Decisions about which currently running applications are allowed to remain running and which new applications are launched are based on the information in the AIT File just as they would be when navigating between services.
- When a locator specifying an AIT File is successfully used in a service selection operation, the applications database shall be populated with the information from that AIT File.

### 9.6.2 Stored services

Stored services are an encapsulation of one or more applications. The lifecycle of a stored service shall obey the same rules as defined for a service in clause 9.1, "Service-bound GEM applications". So, for example, currently running applications will be stopped if they are not listed to be runable in the stored service's Application Description information.

Hence there is no requirement to execute a stored services simultaneously with a service or application.

Stored applications running as part of a stored service shall:

- Execute with the Application identification that was stored with them when they were stored.
- Be able to access a file system using GEM APIs where:
  - Any file that is present shall have the same name, relative path as it did in the broadcast file system.
  - Any API providing byte level access to the data of a file shall return the same data as was broadcast.
  - The set stored may be a superset of the files listed in the Application description file.
  - That file system is read-only to the GEM application.

NOTE 1: This does not preclude the implementation also storing an optimized representation of files that would be presented to certain APIs. For example, this allows class files to be precompiled for subsequent presentation at the class loader.

The results of an application attempting to access files or directories above the directory that originally contained the Application description file are implementation dependent but shall either succeed or fail using the failure modes specified for the mechanism used.

When a stored service is started, any previously running streamed media decoders shall continue to run. It is the responsibility of applications to stop them if so required. Any resources used by such streamed media decoders shall have the lowest possible priority in any resource conflicts.

A stored service has the same lifecycle semantics as other services types (such as broadcast). So, when a stored service is selected the stored Application Description information is inspected to determine which (if any) applications should auto start and which (if any) currently running applications are allowed to continue running by virtue of being included in the stored service.

NOTE 2: Currently, no API has been defined that would allow the addition of externally authorized applications to a stored service.

A stored service is built up by identifying applications from another service. It is not a copy of a service.

When an application is added to a stored service, the following information shall be stored:

- The storage_property and version information from the Application storage descriptor (see clause 10.8.2, "Application storage descriptor").
- Sufficient information to be able to reconstruct the application_descriptors_loop (e.g. so that it can be returned by the `org.dvb.application.AppAttributes.getProperty()` method).
- The application type.
- The Application identification.
- At least all those files listed in the Application description file with priority of "critical".
- Implementations may store files in addition to those listed in the Application description file.
- Sufficient information to be able to construct the set of permissions to be granted to the application including the certificate files needed for any credentials.

Implications of the above are:

- If any non-critical files are needed for the application functionality then the application is responsible for obtaining them in the event that they are not stored.
- No information from Application Description common loop descriptors is required to be stored.
- Information from the transport protocol descriptors is only relevant during the storing of the application. When the application is executing from storage it is responsible for creating the transport connections it requires.

If applications within a stored service are also listed in a currently selected service they are runable with the same constraints as other broadcast applications in the context of that service. So, as with "Cached applications", the consequence of storage is to allow better performance.

If a stored application service is removed while it is being presented then presentation of that service will stop with a `PresentationTerminatedEvent.SERVICE_VANISHED`. The navigator may select another service in an implementation-dependent manner. (For example, a stored application may choose to remove the service it is stored in, including removing itself. This is not an error.)

Stored application services which contain no autostart applications should not be presented to the end-user, either by the navigator or by GEM applications such as EPGs. GEM terminals should delete stored application services which contain no applications when/if no running GEM applications have references to that service.

### 9.6.3 DVB-J Model

This clause is applicable only if clause 9.6.1 or 9.6.2 is required by the GEM terminal specification.

For DVB-J applications, APIs required to include some support for these applications include the following.

- Service selection API (see clause 11.6.2).
- "Content referencing" (see clause 11.7.6) as described in clause 11.11.12, "Support for the HTTP Protocol in DVB-J".
- "Protocol independent SI API" (see clause 11.6.5) as described in clause 11.11.12, "Support for the HTTP Protocol in DVB-J".

When a stored service is successfully created using the method `StoredApplicationServiceFactory.createStoredApplicationService`, this service shall appear in the list of services maintained by the `SIManager`. It shall be returned by `filterServices` both when passed an instance of `ServiceTypeFilter` constructed with the type `StoredApplicationServiceType.STORED_APPLICATION_SERVICE` and when passed null to list all known services. The method `ServiceContext.select` shall support selecting stored services as identified by instances of `StoredApplicationService` created or returned using all these mechanisms. This is described in more detail in clause 11.12.2.2, "Modified behaviour of GEM 1.0 APIs".

In profiles where application signalling over the interaction channel is supported, the method `SIManager.getService` shall accept instances of `javax.tv.locator.Locator` whose external form is valid "http" and "https" URLs and return a `Service` object. The method `ServiceContext.select` shall support selecting services identified by such service objects. This is described in more detail in clause 11.11.12, "Support for the HTTP Protocol in DVB-J".

### 9.6.4 Common behaviour

This clause is applicable only if clause 9.6.1 or 9.6.2 is required by the GEM terminal specification.

The environment in which stand-alone GEM applications are presented or executed shall be the same as for applications related to a conventional service except as detailed below.

- Any existing video and audio running from before the application was started shall continue running.
- Applications shall not have a mechanism for attaining the identity of any such existing video or audio.
- Applications wishing to hide any such existing video may cover the full screen area with graphics.
- Applications have no ability to hide any such existing audio.
- GEM terminals shall assign the lowest possible resource priority to the presentation of any such existing video and audio.

# 9.7 Lifecycle of internet access applications

## 9.7.1 General issues

GEM terminals supportinginternet access applications shall support at least the execution of one internet access application at one time. There is no requirement to support the execution of more than one internet access application at one time. There is no requirement to support the execution of internet access applications and GEM applications at the same time. GEM terminals where internet access applications and GEM applications can be in memory and executing code at the same time but cannot share access to the screen are allowed. Such GEM terminals shall be considered as supporting internet access applications and GEM applications at the same time in the rest of the present document. In these GEM terminals, any GEM applications in the active state may be put in the paused state when they lose access to the screen to an internet access application. The present document is intentionally silent about priorities for access to resources like memory in the event of a conflict between GEM applications and internet access applications.

NOTE: This means that when running internet access applications and GEM applications simultaneously, if memory in the GEM receiver runs low, some implementations may choose to terminate the GEM applications and others may choose to terminate the internet access applications.

## 9.7.2 Starting internet access applications from GEM applications

Where GEM terminals support simultaneous execution of internet access applications and GEM applications at the same time, then the internet access application shall start when requested. The internet access application shall be considered to be a peer of the GEM application and not a child of it. The lifecycle of the internet access application is not constrained by the lifecycle of the GEM application which started it.

Where GEM terminals do not support this simultaneous execution then the GEM application shall be destroyed / killed as part of the starting of the internet access application. In this case, when the end user of the GEM terminal exits the internet access application for reasons other than selecting a "dvb:" link to a service, the GEM terminal shall return to presenting the service (if any) which it was presenting before the internet access application was started. Any GEM applications in this service will be re-started without any previous state using the normal mechanisms defined in the present document.

The APIs to enable DVB-J applications to manipulate internet access applications are defined in annex AH, "Internet client APIs".

## 9.7.3 Selecting DVB services from internet access applications

In GEM, internet access applications shall support end user starting of DVB services (i.e. using the 'dvb:' URL) and GEM applications signalled over the interaction channel (i.e. using an application description file referenced by an HTTP URL). This shall be handled by the normal service selection mechanism used elsewhere in the GEM specification (e.g. in the implementation of the service selection API).

On GEM terminals which support running internet access applications and GEM services at the same time, the new GEM service shall start running if and only if the service context in which the internet access application is running can support the running of GEM services. If the service context in which the internet access application is running cannot support the running of GEM services then the service selection operation shall fail and this will be reported to any GEM applications listening for service selection events on that service context. If the internet access application was started by a GEM application using the `org.dvb.internet` API, that GEM application is still running, and that GEM application has registered for `InternetClientEvents`, then if the service selection fails, that application will also receive an `InternetClientFailureEvent` including the locator of the DVB service which failed to be selected.

NOTE: It is up to the GEM application receiving an `InternetClientFailureEvent` to extract the locator for the DVB service which failed to be selected and, if it so desires, select it in its own service context. If it doesn't do this then the end-user's starting of the DVB service fails.

If the internet access application was started directly from the navigator and not by a GEM application, the service context used to present any DVB service selected is implementation dependent but shall be one which can support the selection of DVB services.

On GEM terminals which do not support running internet access applications and GEM applications at the same time, the internet access application shall behave as if the end user of the GEM terminal had asked to exit that application. This may result in a platform specific dialogue with the end user, e.g. to ask about saving partly composed email messages as a draft. Depending on the nature of any such dialogue, the selection of the DVB service may fail if the end user of the GEM terminal does not wish to exit the internet access application concerned.

There is no requirement to remember the internet access application which selected a DVB service or to return to that internet access application when leaving the DVB service which it selected. This is the opposite of starting internet access applications from GEM applications.

## 9.8 Plug-ins

Attention is drawn to the general rules in clause 4.2.

Clause 5.4, "Plug-ins" of the present document defines the concept of plug-ins for use in migration situations, for the support of applications in existing content formats. Services and networks using such an approach to migration have a number of choices defined by the present document. Where a service carries one or more applications in an existing content format (referred to as delegated applications), those applications may be signalled either using the native signalling defined for that existing content format or using the Application Description based signalling defined in the present document or both. Two types of plug-ins are defined: inter-operable plug-ins and implementation specific-plug-ins. The present document is intentionally silent about the operation and signalling of implementation specific plug-ins. Signalling of inter-operable plug-ins shall be done as defined in clause 10.7.2, "GEM signalling scenario".

The lifecycle of inter-operable plug-ins shall obey the semantics of DVB-J applications concerning service selection operations as defined in clause 9.1.4.1, "A new service being selected replacing a previously selected one" of the present document.

Where the Application Description is used to signal delegated applications, the maintainer of the specification for that format shall register with the DVB project to obtain a value for the "application type" field of the Application Description (see table 12, "Application types"). Having done this, the maintainer of the specification for the existing content format is responsible for defining any additional descriptors for the Application Description which are required for the support of their content format.

## 9.9 Stored and cached applications

This section describes the common storage behaviour that applies to all stored applications. See clause 9.1.9, "Cached applications" and clause 9.6.2, "Stored services".

### 9.9.1 Storing files

When a file that forms part of an application is stored, the following metadata shall be stored with the file:

- The application type.
- The Application identification.
- The version information from the Application storage descriptor (see clause 10.8.2, "Application storage descriptor").
- If the Application description file lists any GEM security file (such as a signature file or a CRL) then these files shall be stored in the normal way (with due regard to the priority, etc.). However, it is not necessary to list the security files in order to retain the security information as the method of retention of this information is an implementation detail.
- Sufficient information about the certificate chain that authenticated the application to be able to determine if any of those certificates have been revoked when it comes to run the application.
- Sufficient information to fully implement the method `DSMCCObject.getSigners` for all signed files which are stored.

Implications of the above are:

- It is not required to store file MIME type as carried by the optional Content type descriptor. The application developer is responsible for ensuring that stored files can be correctly interpreted without relying on this information.
- It is not required to store cache priority information from the Caching priority descriptor. The application developer is responsible for ensuring that dynamic data is not listed in the Application description file.

### 9.9.2 Version management

To be eligible for any kind of application storage, an application must be signalled with an `application_storage_descriptor`. This contains the version number for the application.

A stored application is uniquely identified by an `organisation_id`, `application_id`, version number triple. Terminals shall support storing multiple versions of the same application. (For example, different service providers may use the same third-party application but may update to a new version at different times.)

Application authors are responsible for ensuring that their application is always broadcast with the correct version number.

If the broadcast version number of an application changes whilst that application is being stored or cached (including as part of pro-active caching by the terminal) any files stored since the last time the Application Description was received with the old version number shall be erased. If this storing or caching was requested via the GEM APIs and is not pro-active caching, the first time this happens, the platform shall automatically attempt to store or cache the new version of the application instead of the old version. If the broadcast version changes again whilst the same application is still being stored/cached, the download shall fail with an `ApplicationDownloadException`.

### 9.9.3 Removing stored applications

If an application is running (i.e. in any state other than `DESTROYED` or `NOT_LOADED`), the terminal shall not remove it from storage or cache.

The API methods to remove an application from storage or cache shall work normally and shall not fail just because the application is running. When the application terminates, only then shall it be removed from storage or cache.

For applications that are running as part of a stored service, removing them from that stored service shall cause them to be stopped as if they were removed from the Application Description (i.e. the `destroyXlet()` method shall be called as normal, and that `destroyXlet()` method shall be able to access stored files).

### 9.9.4 Interrupted downloads

Storing or caching an application may be interrupted, e.g. if the application that requested the storing or caching terminates or is terminated. (For example, if the user zaps to a service whilst a download is in progress.)

If storing or caching an application is interrupted before all critical files are downloaded, terminals are recommended to keep the partially-downloaded application stored in an implementation-specific cache, and if the terminal is asked to download the same version of that application again, then the terminal should resume where it left off and not download the already-stored files again.

If storing or caching an application is interrupted after all critical files are downloaded, the storing or caching has succeeded, even if there are non-critical files which could still be downloaded and have not been.

### 9.9.5 Dynamic behaviour

If an application that was not previously stored becomes stored, the following shall apply:

- An `AppsDatabaseEvent` with event id `APP_ADDED` shall be sent for that application.
- If the application was signalled in the current service as both `AUTOSTART` and `not_launchable_from_broadcast`, it shall be started.

- Subsequent calls to `AppsDatabase.getAppAttributes` for that application will return an instance of `ExtendedAppAttributes` whose `isStored()` method will return true until/unless the application is later removed from storage.

NOTE: For applications with `not_launchable_from_broadcast` set, the return value of the `AppAttributes.isStartable()` method will change at the same time.

# 9.10 Lifecycle interactions between GEM and resident applications

Application authors should be aware that applications may be made invisible without having other resources withdrawn, and may not be put in the paused state.

# 9.11 Providers

## 9.11.1 Introduction (informative)

Providers is a term used in Java SE for extensions to platform facilities exposed via standardised APIs. These are often adaptors between implementations of standard APIs and market specific protocols. They use standard APIs and are independent of any particular implementation and hence can be deployed by operators directly without a system software update or similar process.

The present document defines a framework for these. Example providers enabled by that framework include the following:

- provider for the PKCS #11 protocol for connecting smart cards to return channel security;
- providers for connecting the standard GEM SI APIs to proprietary SI formats and protocols;
- providers for connecting the standard GEM media presentation APIs to various protocols for control of media presentation in IPTV, both proprietary and variations on standards.

Providers may be restricted in scope to a single GEM application or may apply to all GEM applications in a GEM terminal. The first of these (called Xlet bound) are available to normal signed GEM applications. The second of these (called System bound) are only available to privileged applications.

Three models for provider class distribution are supported:

- Provider classes are distributed as part of the application using them. They are loaded by the application's Classloader and without any special extension to the signalled classpath. The provider classes need not be stored in the GEM terminal. Each application using the provider has its own copy of the provider classes.
- Provider classes are distributed as part of an application. This application declares (via signalling) that it contains the classes for that provider. When an application using that provider starts, the classes for the provider are used from the application that contains it without an instance of the containing application being started. These provider classes may be loaded from the network or from storage if the containing application has been stored, e.g. using the mechanisms defined in clause 9.9, "Stored and cached applications" of the present document.

  They are loaded either by the application's Classloader (with an extended classpath) or by a parent of that Classloader not shared by any other application. Each application using the provider has its own copy of the provider classes.

NOTE: The application containing the provider classes is used as a container for those classes and may never be instantiated as an application instance in its own right.

Such applications may be signalled with application control code `DISABLED`. `StoredApplicationServices` which only contain applications signalled with this application control code will not be selectable as defined by `StoredApplicationService.isSelectable()`.

- Provider classes are distributed as part of a different application from the one(s) using them. They are installed by an instance of that application. They are loaded by the Classloader of that application. They are never visible to any other application. There is a single copy of those classes.

The first two of these models are only used for `XletBoundProviders` and the third is only used for `SystemBoundProviders`.

### 9.11.2 Lifecycle of xlet bound providers

When an instance of an application with one or more provider usage descriptors in its Application Description starts, the implementation shall attempt to find a provider for each signalled provider name by matching the provider names. For each matching provider, the following shall be done:

- The classes of the application containing the provider shall be added to the classpath of the application being started (or added to the classpath of a parent classloader of the application being started which is not shared with any other application).

- An instance of the class whose name is signalled in the `provider_class_name` shall be created and registered with the `ProviderRegistry` as an `XletBoundProvider`. This shall be created before the constructor of the Xlet class is called.

If no provider is found for a signalled provider name, the implementation shall make provision in the classloader of the application instance in case a provider of that name is installed during the lifetime of that application instance. If a provider of that name is installed during the lifetime of that application instance, the steps listed above shall be followed and the provider shall become available to the application.

Providers may be registered and unregistered during the lifetime of an application instance however any providers which are still registered when the application instance terminates shall be unregistered at that time.

While any application instance is using a provider exported by a second application, that second application shall not be removed from storage, see clause 9.9.3, "Removing stored applications". If a request to remove the second application succeeds while providers it exports are in use, it shall only be removed once all of the application instances using providers it exports have terminated. Once such a request has succeeded, no new providers shall be installed from the removed application. If providers of that name are to be used, a new application must be stored which exports that a provider of that name. Once such a new application has been stored, that application shall be used for new providers even though the former application may not actually have been removed yet.

### 9.11.3 Lifecycle of system bound providers

These are registered by the application containing the provider. The Xlet which registered the provider may unregister it. Otherwise the provider will be unregistered when the Xlet which contains it terminates.

NOTE: Xlets containing system bound providers would typically be unbound applications without any user interface.

## 9.12 Impact of graphics constraints on the application model

### 9.12.1 Impact on generic applications

The graphics constraints descriptor (clause 10.4.3.3, "Graphics constraints descriptor") shall be used in the following ways:

a) Selecting the default graphics configuration for an application when an instance of that application starts or deciding not to start the application.

b) Defining whether an application can continue to show a user interface (or continue to run) when the set of available graphics configurations changes.

When an instance of an application starts, its requested graphics configuration(s) shall be compared against the available graphics configurations for the context in which the application is being started.

- If exactly one graphics configuration requested by the application is available then that one shall become the default graphics configuration of the application.
- If more than one graphics configuration requested by the application is available then the most preferred available configuration shall become the default graphics configuration of the application.
- If no graphics configuration requested by the application is available then the application instance shall either run without a visible UI or not run at all depending on the value of the `can_run_without_visible_ui` flag.

While an application instance is running, if the set of available graphics configurations changes then the following shall apply:

- Running applications signalled with `handles_configuration_changed` set to 0 shall be killed if their default graphics configuration is not available after the change.
- Running application signalled with handles_configuration_changed set to 1 where none of the requested graphics configurations are available shall either be killed or run without a visible UI according to the value of their `can_run_without_visible_ui` flag.
- Running applications signalled with `handles_configuration_changed` set to 1 where at least one of the requested graphics configurations are available shall run with their default graphics configuration being the most preferred which is available after the change. If this is different from the graphics configuration used before the change then this may result a change in the size and/or the HGraphicsDevice of their HScene.

Table 10 specifies the behaviour of applications in the same service as video when an application external to the service context where the video is presented takes control of video presentation.

**Table 10: Application behaviour under external control of video presentation**

| handles_externally_controlled_video | can_run_without_visible_ui | Video already under external control when application starts | Application already running when video comes under external control |
|---|---|---|---|
| 0 | 0 | Fails to start | Killed except for applications with no visible UI which continue to run. |
| 0 | 1 | Starts but shall be unable to obtain a visible UI | UI becomes invisible |
| 1 | 0 | Starts with visible UI | Continues to run |
| 1 | 1 | Starts with visible UI | Continues to run |

## 9.12.2 Impact on DVB-J applications

The default graphics configuration of the application shall be used for the HScene returned by the implementation of the following methods:

- `org.havi.ui.HSceneFactory.getDefaultHScene()`
- `org.havi.ui.HSceneFactory.getDefaultHScene(HScreen)`
- `javax.tv.graphics.TVContainer.getRootContainer()`
- `javax.microedition.xlet.XletContext.getContainer()`

NOTE: Regardless of the signalled default graphics configuration, applications may use the other facilities provided by the `org.havi.ui` package to discover HGraphicsConfigurations and attempt to select them subject to being able to reserve the corresponding resource.

For a DVB-J application, running without a visible UI shall mean the following:

- Already running applications shall have the visibility of their HScene set to `false`. Attempts to change this shall fail silently while the application remains running without a visible UI.

- Applications starting without a visible UI shall have any attempts to obtain a default HScene fail for methods with a defined failure mode. For methods without a defined failure mode, an HScene shall be returned but attempts to set the HScene visibility to true or to call the show method shall fail silently.

In table 10, "Application behaviour under external control of video presentation" DVB-J applications shall be considered to have no UI if they do not have any HScene instances except for ones which have been disposed.

# 9.13 Unbound Applications

This clause is optional in all profiles of GEM. GEM terminal specifications may include all or part of the specification elements of this clause, or may include none at all. If specification elements are included, functional equivalents may be specified where necessary.

## 9.13.1 Introduction to unbound applications (informative)

### 9.13.1.1 Scope

Unbound applications are ones whose lifecycle is not bound to the presentation of a particular piece of content, either a service or content provided on demand. This is in contrast to broadcast GEM applications (see clause 9.1, "Service-bound GEM applications") which are bound to one or more TV channels by signalling and only run while that TV channel is being presented. The effect of unbound applications can be achieved in GEM 1.0 by signalling them as part of all TV channels in a network with the result that they run regardless of which TV channel or service is selected.

OCAP [5] adds support for 3 different versions of unbound applications:

- Host device manufacturer applications installed with the OCAP implementation.
- Applications that are signalled through the XAIT.
- Applications that are registered through the Monitor Application.

The second and third of these are in the scope of the present document. The first of these is outside that scope of the present document.

### 9.13.1.2 Divergences from OCAP Solution

The GEM design is based as closely as possible on OCAP [5] however some technical changes were found to be necessary to address different market requirements. These are as follows:

- Service bound applications are more important in GEM than in OCAP. One specific concern with unbound applications is their impact on the testing of service bound applications. The GEM design diverges from OCAP in order to minimize the impact on testing of service bound applications.
- GEM unbound applications will be used in IPTV where signalling is XML based rather than MPEG-2 based. The GEM design diverges from OCAP in its use of XML for signalling in an IPTV environment.
- Even though IPTV deployments today are based around a single network operator owning the receiver, the DVB commercial requirements for IPTV include scalability to future situations where services from multiple IPTV operators can be received. The GEM design diverges from OCAP in order to permit this scalability.
- GEM does not support CableCARD or the out of band channel (OOB) hence aspects of OCAP related to or relying on these cannot be re-used directly in GEM.
- Some aspects of OCAP are a direct consequence of specific US regulatory requirements. In particular the absence of support for operator applications built in to the box at the time of manufacture. This constraint would not apply to GEM-IPTV.

#### 9.13.1.3 Overview

The present document defines a single environment within which there are multiple service contexts. These may include:

- Service contexts presenting broadcast services.
- Service contexts presenting stored application services.
- Service contexts presenting return channel signalled services.
- Service contexts presenting internet client services (in GEM terminals supporting the internet access profile).
- Service contexts presenting recorded services (in GEM terminals supporting the GEM PVR/PDR extension).
- Service contexts presenting abstract services (containing unbound applications).

When the GEM environment is initialised, abstract services are created based on the signalling in the XAIT (see clause 9.13.4, "Initialization of GEM Environment" below). Abstract services may also be created by GEM applications on GEM terminals supporting the privileged application extension.

Where there is a subsidising service provider:

- they can control the single environment using the features of the privileged application option;
- they are responsible for managing the set of unbound applications. Other service providers can only deploy unbound applications with their agreement.

In the absence of control by a subsidising service provider, the controlling role is performed by the GEM implementation as is the case in previous versions of GEM.

### 9.13.2 Service model

Clause 10.2.2.2 of OCAP [5] "OCAP Service Model" may be supported.

### 9.13.3 Application lifecycle

Clause 10.2.2.3 OCAP [5] "Application Signaling and Lifecycle" may be supported.

### 9.13.4 Initialization of GEM Environment

When the GEM environment is initialized, (either from scratch or after a re-start) the following steps shall apply:

- The XAIT shall be loaded and parsed including updating the service list and the applications database.
- The most recent initial monitor application (if any) shall be identified (either signalled in the XAIT and present in the network or already stored in the GEM terminal). If one is identified then it shall be launched and the implementation shall wait for it to signal that it is ready by calling `org.ocap.OcapSystem.monitorConfiguredSignal`, (see clause 11.15.5, "OCAP annex O - the org.ocap package" in the present document).
- For each abstract service in the service list that is signalled as "`auto_select`" a service context shall be created and the abstract service shall be selected in that service context. If the version of a XAIT signalled application in application storage matches that defined in the XAIT, then the stored version may be launched. Otherwise, the signalled version shall be launched.

NOTE: The contents of this clause are aligned with clause 20.2.2 of OCAP [5] however the number of dependencies on CableCARD in that clause make it impractical to reference that text.

# 10 Application signalling

## 10.1 Introduction

This clause covers the following topics:

- Identification and launching of applications associated with a service.
- Requirements on the signalling that enables a broadcast to manage the lifecycle of applications.

MHP [1] contains a model of signalling that fulfils the requirements of GEM, but other signalling is possible. Broadly speaking, GEM places requirements on both the format of an application and requirements underlying its signalling. GEM does not, however, define the signalling that shall be used or the packaging of applications; this is left for GEM-based specifications to define.

### 10.1.1 Summary of requirements on common signalling

The minimum signalling requirements for any GEM application are summarized as follows:

- Some form of Application Description (see clause 10.4) with information sufficient to:
  - identify the source of the application code and other assets;
  - identify the application's application ID and organisation ID;
  - identify the name of the application.

### 10.1.2 Summary of additional signalling for DVB-J applications

The minimum additional signalling requirements for DVB-J applications are summarized as follows:

- A DVB-J Specific Application Description (see clause 10.5) with information sufficient to:
  - signal parameters to the application;
  - indicate the initial class of the application.

## 10.2 Program specific information

A service carrying GEM applications shall contain information sufficient to locate the following:

- the Application Description (see clause 10.4) for each application in the service;
- the source of the application code and data.

## 10.3 Locators within an Application Description

Some fields of the Application Description contain locators, e.g. locators to a directory containing certain kinds of files. These locators can be to any transport defined within a GEM terminal specification, e.g. they can be locators to an object carousel, part of a data carousel, an http URL, etc. GEM does not mandate any particular transport. It does, however, require at least one transport that is capable of carrying the information needed to launch applications. This transport shall be capable of carrying files, or directory hierarchies containing files. The ability to list the contents of a directory is optional.

# 10.4 Application Description

The Application Description provides full information on an application, its parameterization, the required activation state of it etc. Specifications based on GEM shall permit the signalling of multiple applications per service, without any arbitrary upper bound less than 255.

Data in the Application Description allows the service provider to request that the GEM terminal change the activation state of an application.

MHP [1], clause 10.4 defines an Application Information Table that fulfils this requirement.

## 10.4.1 Application Description transmission and monitoring

It shall be possible to arrange for signalling such that the maximum time interval between the moment the Application Description is updated and the moment the new version is detected by the terminal will be no more than 30 seconds.

## 10.4.2 Visibility of Application Description and tuning

If an application tunes away from a transport stream where its signalling is carried without selecting a new service, it shall be permitted to continue running even if the Application Description is no longer available to the GEM terminal.

GEM terminal specifications for packaged media targets may not support a tuner. In such a case, the above paragraph does not apply.

## 10.4.3 Content of the Application Description

The Application Description describes applications and their associated information. It shall contain information sufficient to derive the following.

**Table 11: Application description**

| Function | Type |
|---|---|
| application_type | enumeration |
| organisation_id | 32 bit unsigned integer |
| application_id | 16 bit unsigned integer |
| application_control_code | enumeration |
| application_profiles_count | 4 bit unsigned integer |
| for (i=0; i<N1; i++) { | |
| application_profile | 16 bit unsigned integer |
| version.major | 8 bit unsigned integer |
| version.minor | 8 bit unsigned integer |
| version.micro | 8 bit unsigned integer |
| } | |
| service_bound_flag | boolean |
| visibility | enumeration |
| application_priority | 8 bit unsigned integer |
| application_name | String |
| application_icon_locator_count | unsigned integer |
| for (i=0; i<N2; i++) { | |
| application_icon_locator | Locator |
| application_icon_flags | 16 bit unsigned integer |
| } | |

application_type: Identifies the type of application. Specifications based on GEM shall provide a mechanism for indicating at least two application types: DVB-J and DVB-HTML. For information purposes current registrations for these application types are shown in table 12.

**Table 12: Application types**

| Application_type | Description |
|---|---|
| 0x0000 | Reserved_future_use |
| 0x0001 | DVB-J application |
| 0x0002 | DVB-HTML application |
| 0x0003 to 0x7FFF | Subject to registration with DVB |

These values are formally maintained by the DVB Project [18].

organisation_id: This 32 bit field is a globally unique value identifying the organization that is responsible for the application. These values are registered in TS 101 162 [29]. Values of zero shall not be encoded.

This field is reproduced in the `organisationName` field of the subject name in the "leaf" certificate of an authenticated application (see clause 12.5.6, "subject").

NOTE 1: The inclusion of this field in the leaf certificate provides authentication of the value.

NOTE 2: `organisation_id` values between 0x80000000 and 0xffffffff are known to create problems in some implementations and their use is strongly discouraged.

application_id: This 16 bit field uniquely identifies the application function. This is allocated by the organisation registered with the `organisation_id` who decides the policy for allocation within the organisation. Values of zero shall not be encoded.

The application id values are divided into two ranges: one for unsigned applications and one for signed applications. This is for security reasons (see clause 12.1.1, "Overview of the security framework for applications"). Applications transmitted as unsigned shall use an application id from the unsigned applications range and applications transmitted as signed shall use an application id from the signed applications range.

**Table 13: Value ranges for application_id**

| Application_id values | Use |
|---|---|
| 0x0000 | Shall not be used |
| 0x0001 to 0x3fff | Application_ids for unsigned applications |
| 0x4000 to 0x7fff | Application_ids for signed applications |
| 0x8000 to 0xfffd | Reserved for future use by DVB |
| 0x8000 to 0x9fff | Application_ids for privileged applications |
| 0xa000 to 0xfffd | Reserved for future use by DVB |
| 0xfffe | Special wildcard value for signed applications of an organization |
| 0xffff | Special wildcard value for all applications of an organization |

Application id values 0xffff and 0xfffe are wild cards. They shall not be used to identify an application but are allowed for use in other descriptors. The value 0xffff matches all applications with the same `organisation_id`. The value 0xfffe matches all signed applications with the same `organisation_id`.

The same application identifier may be used in different application types for applications performing essentially the same function.

application_control_code: A DVB-J application control code, as defined in clause 10.4.3.1.

Support for the `REMOTE` application type is not required, but may optionally be present in GEM terminal specifications.

application_profiles_count: The number of application profiles signalled for this application.

application_profile: This 16 bit field is an integer value which represents the application type specific profile. This indicates that a receiver implementing one of the profiles listed in this loop is capable of executing the application.

version.major: This 8 bit field carries the numeric value of the major sub-field of the profile version number.

version.minor: This 8 bit field carries the numeric value of the minor sub-field of the profile version number.

version.micro: This 8 bit field carries the numeric value of the micro sub-field of the profile version number.

The last four fields above indicate the minimum MHP profile on which an application will run. For example, an application that relies on the guarantees of GEM 1.0 would run on an appropriate profile of MHP 1.0.2. The underlying signalling of the application shall indicate the minimum profile that the application requires in a way that can be mapped to MHP profiles and the MHP version number.

service_bound_flag: If this field is set to "1", the application is associated only with the current service and so the process of killing the application shall start at the beginning of the service change regardless of the contents of the destination Application Description.

visibility: This 2 bit field specifies whether the application is suitable to be offered to the end-user for them to decide if the application should be launched. Table 14 lists the allowed values of this field.

**Table 14: Definition of visibility states for applications**

| Visibility | Description |
|---|---|
| 00 | This application shall not be visible either to applications via an application listing API or to users via the navigator with the exception of any error reporting or logging facility, etc. |
| 01 | This application shall not be visible to users but shall be visible to applications via an application listing API. |
| 10 | Reserved_future_use. |
| 11 | This application can be visible to users and shall be visible to applications via the application listing API. |

NOTE 3: For example, in a service offering a number of games to the end-user, these values would be used as follows:

00 - the autostart generic launcher application offering the end user the choice of which game to launch.

11 - the games which the end-user can chose between.

01 - the common server application which all the games use to communicate with a server over the interaction channel.

application_priority: This field identifies a relative priority between the applications signalled in this service. Terminal specifications based on GEM shall support at least 32 priority levels.

- Where there is more than one application with the same Application identification this priority shall be used to determine which application is started.
- Where there are insufficient resources to continue running a set of applications, this priority shall be used to determine which applications to stop or pause.
- A larger integer value indicates higher priority.
- If two applications have the same application identification and the same priority, the GEM terminal may make an implementation-dependent choice on which to start.
- When used in an XAIT, the value of 255 shall identify the initial monitor application. The value of 255 has no special meaning for applications signalled in an AIT.

NOTE 4: Requirements on signalling unbound applications with this priority are found in clause 10.2.2.3.1 "Applications Signaled in the XAIT" of OCAP [5] which is referenced by clause 9.13.3, "Application lifecycle".

application_name: A string that names the application in a way meant to be informative to the user. The signalling shall support strings whose UTF8 encoding is up to 128 bytes, not including any termination character. It is permissible to signal more than one application name, e.g. the application name could be given in several different languages, with a method for determining which one is to be presented to the user, as is done in MHP. In all cases, it shall be possible to associate an ISO 639 [84] language code with each application name. The GEM terminal specification shall define which text encodings are required to be supported for this name.

application_icon_locator_count: The number of application icon locators associated with this application. Signalling to support values of 0 and 1 shall be present. Terminal specifications based on GEM may support any number of application icon locators.

application_icon_locator: Information sufficient to derive a locator to a directory containing application icons. The application icons shall be in files in the directory indicated by this locator, in the format specified in clause 10.4.3.2.

application_icon_flags: Flags describing the icon files in the directory identified by the `application_icon_locator`, in the format specified in clause 10.4.3.2.

NOTE 5: GEM terminal specifications may define application icon signalling that also allows different icon sizes to be indicated.

### 10.4.3.1 DVB-J application control codes

The application control codes for DVB-J applications are listed in table 15.

**Table 15: DVB-J application control code values**

| Code | Identifier | Semantics |
|---|---|---|
| 0x00 | | Reserved_future_use. |
| 0x01 | AUTOSTART | The file system element(s) (e.g. an Object Carousel module) containing the class implementing the Xlet interface is loaded, The class implementing the Xlet is loaded into the VM and an Xlet object is instantiated, and the application is started subject to usual restrictions, etc. |
| 0x02 | PRESENT | Indicates that the application is present in the service, but is not autostarted. |
| 0x03 | DESTROY | When the control code changes from AUTOSTART or PRESENT to DESTROY, the destroy method of the Xlet is called (with the **`unconditional`** parameter set to **false**) by the application manager and the application is allowed to destroy itself gracefully. |
| 0x04 | KILL | When the control code changes from AUTOSTART or PRESENT to KILL, the destroy method of the Xlet is called (with the **`unconditional`** parameter set to **true**) by the application manager. |
| 0x05 | | reserved_future_use. |
| 0x06 | REMOTE | This identifies a remote application that is only launchable after service selection. |
| 0x07 to 0xFF | | Reserved_future_use. |
| 0x07 | DISABLED | Application shall not be started and attempts to start it shall fail. Such applications need not be signalled with a dvb_j_application_descriptor and the initial_class_byte in the dvb_j_application_location_descriptor is never used. |
| 0x08 to 0xFF | | Reserved_future_use. |

See clause 9.2.3, "DVB-J Application Lifecycle".

### 10.4.3.2 Application icons descriptor

Zero or one instance of this descriptor shall be included in the application information of an application. It allows icons to be associated with the application. The content format for these possible icons shall be restricted PNG as specified in clause 15.1, "PNG - restrictions". Table 16 shows the syntax of the application icons descriptor.

**Table 16: Application icons descriptor syntax**

| | No.of Bits | Identifier | Value |
|---|---|---|---|
| application_icons_descriptor() { | | | |
| descriptor_tag | 8 | uimsbf | |
| descriptor_length | 8 | uimsbf | |
| icon_locator_length | 8 | uimsbf | |
| for (i=0; i<N; i++) { | | | |
| icon_locator_byte | 8 | uimsbf | |
| } | | | |
| icon_flags | 16 | bslbf | |
| for (i=0; i<N; i++) { | | | |
| reserved_future_use | 8 | bslbf | |
| } | | | |
| } | | | |

descriptor_tag: This 8 bit integer with value 0x0B identifies this descriptor.

icon_locator_length: This 8 bit integer specifies the number of bytes in the string that prefixes standard icon file name.

icon_locator_byte: This 8 bit value is one byte of the icon locator string.

The icon locator is the first part of the string that specifies the location of the icon files. This is application type dependent. See table 17. The `icon_locator` shall not end with a "/" slash character.

**Table 17: Icon locator semantics**

| Application_type | Description |
|---|---|
| 0x0000 | Reserved_future_use. |
| 0x0001 | For DVB-J this is a path relative to the base directory of the application as defined in MHP [1] clause 10.9.2, "DVB-J application location descriptor". |
| 0x0002 | For DVB-HTML this is a path relative to the physical root of the application as defined in MHP [1] clause 10.10.2, "DVB-HTML application location descriptor". |
| 0x0003 to 0xFFFF | Reserved_future_use. |

icon_flags: This 16 bit field carries a value which is the bitwise OR of the flag bits that identify the icons that are provided for the application. The flag bits are defined in table 18.

**Table 18: Definition of different icon flags**

| Icon flag bits | Description of icon size and pixel aspect ratio |
|---|---|
| 0000 0000 0000 0001 | 32 x 32 for square pixel display |
| 0000 0000 0000 0010 | 32 x 32 for broadcast pixels on 4:3 display (see note) |
| 0000 0000 0000 0100 | 24 x 32 for broadcast pixels on 16:9 display |
| 0000 0000 0000 1000 | 64 x 64 for square pixel display |
| 0000 0000 0001 0000 | 64 x 64 for broadcast pixels on 4:3 display |
| 0000 0000 0010 0000 | 48 x 64 for broadcast pixels on 16:9 display |
| 0000 0000 0100 0000 | 128 x 128 for square pixel display |
| 0000 0000 1000 0000 | 128 x 128 for broadcast pixels on 4:3 display |
| 0000 0001 0000 0000 | 96 x 128 for broadcast pixels on 16:9 display |
| 0000 0010 0000 0000 | 256 x 256 for square pixel display |
| 0000 0100 0000 0000 | 256 x 256 for broadcast pixels on 4:3 display |
| 0000 1000 0000 0000 | 192 x 256 for broadcast pixels on 16:9 display |
| xxxx xxx0 0000 0000 | reserved_future_use |
| xxxx 0000 0000 0000 | reserved_future_use |
| NOTE: Approx. 15/16 pixel aspect ratio on 50 Hz system. | |

The file names for the icon files are encoded in a standard way:

```
filename    = icon_locator "/dvb.icon." hex_string
hex_string  = 4*4hex
hex         = digit | "A" | "B" | "C" | "D" | "E" | "F" | "a" | "b" | "c" | "d" | "e" | "f"
```

```
digit       = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

An icon file shall contain exactly one icon. The icon contained in the icon file shall have the format specified by the 4 hexadecimal digit postscript of its file name.

NOTE: This means that if the icon_flags field of the application_icons_descriptor were to have a value indicating the presence of multiple icons, each of the indicated icons would have its own icon file. For example, if icon_flags has a value of 0x0005, the directory specified by `icon_locator` would contain two files named dvb.icon.0004 (for 24 x 32 square pixel rendering) and dvb.icon.0001 (for 32 x 32 square pixel rendering).

### 10.4.3.3 Graphics constraints descriptor

The graphics constraints descriptor defines the circumstances under which an application can work (or has been tested to work). These circumstances are:

- which full screen graphics resolutions an application supports;
- whether an application can work when its video is controlled (e.g. scaled to less than full screen size) by another application running external to the first application's service context (e.g. EPG or navigator).

**Table 19: Graphics constraints descriptor syntax**

| | No. of Bits | Identifier | Value |
|---|---|---|---|
| graphics_constraints_descriptor() { | | | |
| descriptor_tag | 8 | uimsbf | 0x14 |
| descriptor_length | 8 | uimsbf | |
| reserved_future_use | 5 | bslbf | |
| can_run_without_visible_ui | 1 | bslbf | |
| handles_configuration_changed | 1 | bslbf | |
| handles_externally_controlled_video | 1 | bslbf | |
| For(i=0;i<N;i++) { | | | |
| graphics_configuration_byte | 8 | Uimsbf | |
| } | | | |
| } | | | |

Where the fields have the following meanings:

descriptor_tag: This 8 bit integer with value 0x14 identifies this descriptor.

descriptor_length: This 8 bit field indicates the number of bytes following the descriptor length field.

can_run_without_visible_ui: If this 1 bit field is set to 1 then the application can usefully run with no user interface visible. If this field is set to 0 then the application is not useful with no user interface visible. Applications signalled with this bit set are responsible for registering listeners to detect when it would be reasonable to show their user interface again.

handles_configuration_changed: If this 1 bit field is set to 1 then the application can handle changes in the graphics configuration between those signalled in this descriptor. If set to zero then once the default graphics configuration has been set for an application instance, it will only correctly display under that graphics configuration.

handles_externally_controlled_video: If this 1 bit field is set to 1 then the application can handle being displayed when the presentation of the video in the same service is controlled by an application external to that service. Examples include picture in picture and picture outside picture.

graphics_configuration_byte: This 8 bit field contains a value from the following table. The full screen configurations are sorted from most preferred to least preferred.

**Table 20: graphics configuration byte values**

| Value | Meaning |
|---|---|
| 0 | Reserved |
| 1 | Full screen standard definition |
| 2 | Full screen 960x540 |
| 3 | Full screen 1280x720 |
| 4 | Full screen 1920x1080 |
| 5 to 31 | Reserved for future use by DVB project |
| 32 to 255 | Reserved for future use |

Applications where this descriptor is not signalled shall be assumed to have a descriptor containing one `graphics_configuration_byte` whose value is 1, `can_run_without_visible_ui` being '0', `handles_configuration_changed` being 0 and `handles_externally_controlled_video` being 0.

Applications where the loop of graphics configuration bytes is empty shall be assumed to not care about a default graphics configuration. Either they do not use graphics or they are written to support the full range of graphics configurations defined in the present document and tested accordingly.

## 10.4.4 Applications from previously selected services

If an application with a `service_bound_flag` of 0 is running when a service selection is performed, it shall continue to run in a newly selected service if the same application is signalled in the new service.

NOTE: To efficiently support this feature on services that do not contain the application code, it may be desirable to have signalling equivalent to that described in MHP [1], clause 10.7.5.

## 10.4.5 AIT File

### 10.4.9.1 Syntax

The interaction channel encoding of the Application Description into the AIT File is as follows:

- A single file shall contain all of the data.
- The file shall contain a concatentation of Application Information Sections (specified in MHP [1], clause 10.4.6, "Syntax of the AIT").
- The possibly multiple sections shall be ordered as follows:
  - Ascending order of `application_type`.
  - Within a single value of `application_type` in ascending order of `section_number`.
- All sections shall have `current_next_indicator` set to '1'.

### 10.4.9.2 Syntactic restrictions

#### 10.4.9.2.1 Transport protocols

The only allowed transport protocol type has id value 0x0003. The protocol_id is an identifier of the protocol used for carrying applications. The values of the `protocol_id` are shown in table 21.

**Table 21: Protocol_id**

| Protocol_id | Description |
|---|---|
| 0x0000 | Reserved_future_use. |
| 0x0001 to 0x0002 | Reserved for use by DVB (see note). |
| 0x0003 | Transport via HTTP over the interaction channel as described in clause 10.8.1.3, "Transport via interaction channel". |
| 0x0004 to 0x00FF | Reserved for use by DVB. |
| 0x0100 to 0xFFFF | Subject to registration in TS 101 162 [29]. |
| NOTE: MHP [1] uses the protocol_id values 0x00001 for the MHP Object Carousel and 0x0002 for IP via DVB multiprotocol encapsulation. | |

#### 10.4.9.3 Semantics

The AIT File shall be loaded once during service selection. There is no requirement to monitor or poll the AIT File for any subsequent changes except as part of a subsequent service selection operation referring to the same AIT File. Consequences of this are:

- Only the AUTOSTART and PRESENT application control codes (of those defined in clause 10.4.3.1) are appropriate.
- No standard mechanism for dynamic lifecycle control is provided.

NOTE: If dynamic lifecycle control is required application private mechanisms could be used. For example, a TCP connection over the interaction channel to a controlling server.

- Changes made to the AIT File after service selection shall not be detected or reported.

#### 10.4.9.4 MIME type

The MIME type for an AIT File shall be "application/vnd.dvb.ait". The file extension shall be ".ait".

Earlier versions of GEM and derived specifications defined the MIME type for AIT file as application/dvb.ait without the vnd. prefix. This type name is deprecated and its use is strongly discouraged. If a GEM terminal gets a deprecated MIME type, it shall behave as if the vnd. prefix was present.

## 10.5 DVB-J specific Application Description

### 10.5.1 General

Additional signalling specific to DVB-J applications shall be present in GEM terminal specifications.

## 10.5.2 Content of DVB-J Application Description

For each Application Description that refers to a DVB-J application, it shall be possible to signal information sufficient to derive the following.

**Table 22: DVB-J Application Description**

| Function | Type |
|---|---|
| `for (i=0; i<N; i++) {` | |
| `dvbj_app_parameter` | String |
| `}` | |
| `base_directory` | Locator |
| `for (i=0; i<N; i++) {` | |
| `classpath_element (optional)` | Locator |
| `}` | |
| `initial_class_name` | String |
| `can_run_without_visible_ui` | Boolean |
| `handles_configuration_changed` | Boolean |
| `handles_externally_controlled_video` | Boolean |
| `For(i=0; i<N; i++) {` | |
| `graphics_configuration_byte` | Byte |
| `}` | |

dvbj_app_parameter: A string that is passed to the application as parameters. The signalling shall support parameter strings such that a minimum total length of 240 bytes can be supported. The length is calculated as the sum of (1 + the sum of (1 + length(dvbj_app_parameter)) where the length of a parameter is the length of that parameter string encoded in UTF8, with no termination character. It shall be possible to signal any string that can be represented with UTF8.

initial_class_name: The fully-qualified name of the initial class of this application. This class shall implement the Xlet interface. The signalling shall support UTF8 encoding up to 80 bytes, not including any termination character. It shall be possible to signal any string that can be represented with UTF8.

base_directory: A locator specifying a directory. This directory is used as a base directory for relative path names. This base directory is automatically considered to form the first directory in the class path (after the path to the system's classes).

classpath_element: GEM-based terminal specification may include optional signalling to indicate a list of other locators to be added to an application's class path.

EXAMPLE: MHP [1], clause 10.9.2 defines the `classpath_extension` for this purpose.

If support for this is included in a terminal specification, there may be restrictions placed on these locators, e.g. that they represent sub-directories of the `base_directory`.

can_run_without_visible_ui: If this value is true then the the application can usefully run with no user interface visible. If this value is false then the application is not useful with no user interface visible. Applications signalled with this value set to true are responsible for registering listeners to detect when it would be reasonable to show their user interface again.

handles_configuration_changed: If this value is true then the application can handle changes in the graphics configuration between those signalled in this descriptor. If false then once the default graphics configuration has been set for an application instance, it will only correctly display under that graphics configuration.

handles_externally_controlled_video: If this value is true then the application can handle being displayed when the presentation of the video in the same service is controlled by an application external to that service. Examples include picture in picture and picture outside picture.

graphics_configuration_byte: This 8 bit field contains a value from the following table. The full screen configurations are sorted from most preferred to least preferred.

**Table 23: Graphics configuration byte values**

| Value | Meaning |
|---|---|
| 0 | Reserved |
| 1 | Full screen standard definition |
| 2 | Full screen 960x540 |
| 3 | Full screen 1280x720 |
| 4 | Full screen 1920x1080 |
| 5 to 31 | Reserved for future use by DVB project |
| 32 to 55 | Reserved for future use |

## 10.6 Constant Values

GEM terminal specifications that include the functional equivalent named "Application Signalling" as defined in clause 15.6, "Functional equivalents" and that define additional descriptors in the Application Description shall register the descriptor tag values with the DVB. For information purposes current registrations are shown in table 24.

**Table 24: Registered AIT Descriptor Tags**

| AIT Descriptor Tag Values | Organization |
|---|---|
| 0x60 to 0x65 | ATSC |
| 0x66 to 0x6D | CableLabs |
| 0x6E to 0x7F | reserved |

These values are formally maintained in TS 101 162 [29].

## 10.7 Plug-in signalling

Two signalling scenarios are defined for delegated applications and plug-ins:

- Native signalling scenario.
- GEM signalling scenario.

GEM terminal specifications may substitute functionally equivalent signalling, if required.

### 10.7.1 Native signalling scenario

In this scenario it is sufficient for the plug-in to be signalled as a normal GEM application. Optionally it could also be signalled as a plug-in using the Plug-in descriptor (see clause 10.7.4).

### 10.7.2 GEM signalling scenario

In this scenario GEM plug-in signalling can replace some or all of the native signalling of the delegated application. The GEM signalling allows the GEM terminal to:

- Identify that one (or more) delegated applications are available.
- The plug-in that is required.
- How to start the plug-in.
- How to introduce the delegated application to the plug-in.

The plug-in may use native signalling defined for the delegated application to locate, load and execute it. This is outside the scope of the present document.

Each delegated application is associated with an Application Description (clause 10.4).

The additional GEM signalling provided to support plug-ins is:

- "Delegated application descriptor" (see clause 10.7.3).

  An optional descriptor for delegated applications.

- "Plug-in descriptor" (see clause 10.7.4).

  A mandatory descriptor for plug-ins using GEM signalling.

## 10.7.3 Delegated application descriptor

Zero or one instance of this descriptor can be placed in the Application Description of the delegated application.

This optional descriptor allows one or more specific plug-ins to be identified for this delegated application. This overrides the default identification of the plug-in based on the application type signalled in the Application Description for the delegated application. If this descriptor is absent then no specific plug-in is identified.

Where more than one application is identified they are listed in order from most preferred to least preferred. If none of the preferred applications is available to the GEM terminal then it shall use its default mechanism to identify the plug-in for this delegated application.

**Table 25: Plug-in reference descriptor**

| | No.of Bits | Identifier |
|---|---|---|
| delegated_application_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| for( i=0; i<N; i++ ){ | | |
| application_type | 48 | uimsbf |
| } | | |
| } | | |

descriptor_tag: This 8 bit integer with value 0x0E identifies this descriptor.

descriptor_length: This 8 bit field indicates the number of bytes following the descriptor length field.

application_identifier: This 48 bit field identifies a specific application that can consume this application.

## 10.7.4 Plug-in descriptor

One or more plug-in application descriptors shall be placed in the Application Description of each plug-in.

This descriptor defines, for an application that implements a plug-in, the set of delegated application formats that it supports. The descriptor identifies both the type of delegated application supported and the set of profiles and versions supported. The normal rules for matching an application to a compatible platform apply when matching a delegated application with a suitable plug-in.

DVB-J applications signalled with this descriptor shall implement the API defined in clause 11.7.8, "Plug-in APIs".

The plug-in shall be considered suitable for running content if the `application_type` matches, and the version numbers are compatible.

**Table 26: Plug-in application descriptor**

| | No.of Bits | Identifier |
|---|---|---|
| plugin_application_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| application_type | 16 | uimsbf |
| for( i=0; i<N; i++ ){ | | |
| application_profile | 16 | uimsbf |
| version.major | 8 | uimsbf |
| version.minor | 8 | uimsbf |
| version.micro | 8 | uimsbf |
| } | | |
| } | | |
| } | | |

descriptor_tag: This 8 bit integer with value 0x0F identifies this descriptor.

descriptor_length: This 8 bit field indicates the number of bytes following the descriptor length field.

application_type: This 16 bit field identifies a delegated application type that this application can consume.

application_profile: This 16 bit field is an integer value which represents the application type specific profile.

version.major: This 8 bit field carries the numeric value of the major sub-field of the profile version number.

version.minor: This 8 bit field carries the numeric value of the minor sub-field of the profile version number.

version.micro: This 8 bit field carries the numeric value of the micro sub-field of the profile version number.

## 10.8 Stored Applications

GEM terminal specifications may substitute functionally equivalent signalling, if required.

### 10.8.1 Use of stored application signalling

A GEM terminal that doesn't provide storage or doesn't recognize these extensions will perceive storable applications as viable applications that potentially can be run from the service-bound connection.

#### 10.8.1.1 Stored broadcast service related applications

For stored service-bound applications the service-bound Application Description information shall be used when launching the stored application. So, little information from the Application Description needs to be stored with the application.

#### 10.8.1.2 Stored stand-alone applications

For stored stand-alone applications, the Application Description information used when launching the application shall come from a stored representation of the Application Description. While a stored service is being presented in a service context, the following apply:

- The applications database shall be populated with information from this stored representation of the Application Description for the applications forming part of that stored service.
- The implementation shall report any changes in the stored representation of the Application Description in the same way as it would report changes in a service-bound Application Description.

## 10.8.2 Application storage descriptor

The application storage descriptor advertises that an application can be stored and provides some indications of its properties. The presence of this application storage descriptor indicates that an Application Description File is provided for the application (see clause 10.8.3). For a storable application a single application storage descriptor shall be placed in the Application Description.

This descriptor, and the implied Application Description File, also supports receivers that implement speculative caching.

**Table 27: Syntax of application storage descriptor**

| | No.of Bits | Identifier |
|---|---|---|
| application_storage_descriptor() { | | |
| descriptor_tag | 8 | uimsbf |
| descriptor_length | 8 | uimsbf |
| storage_property | 8 | uimsbf |
| not_launchable_from_broadcast | 1 | bslbf |
| launchable_completely_from_cache | 1 | bslbf |
| is_launchable_with_older_version | 1 | bslbf |
| reserved | 5 | bslbf |
| reserved | 1 | bslbf |
| version | 31 | uimsbf |
| priority | 8 | uimsbf |
| } | | |

descriptor_tag: This 8 bit integer with value 0x10 identifies this descriptor.

descriptor_length: This 8 bit field indicates the number of bytes following the descriptor length field.

storage_property: This 8 bit field indicates the characteristics of the application w.r.t storage.

**Table 28: Semantics of storage property values**

| Storage_property | Property |
|---|---|
| 0 | broadcast related |
| 1 | stand alone |
| 2 to 255 | reserved |

If the storage property is "broadcast related" then the application's life cycle is controlled by the broadcast service. Such applications are suitable for caching but not for running as a stand alone application.

If the storage property is "stand alone" then the application can usefully be launched by the user independently of a broadcast service. Applications with this property may also be launched as if broadcast related if the currently selected service lists them in its Application Description.

launchable_completely_from_cache: This 1 bit field is a flag. When set to "1", this indicates that this application can be run entirely from cache, without connecting to the transport protocol signalled in the application's Application Description, if all the critical files have been cached. If set to "0", a connection to this transport protocol must be made in order to run this application as a broadcast-related application. This flag only applies if running the application as a broadcast-related application, it is ignored when storing an application into a stored service, as all applications signalled as "stand alone" can run without a connection to the transport protocol when run as part of a stored service.

This flag shall only be set to "1" when `not_launchable_from_broadcast` is also set to "1".

NOTE: This flag should be set to "1" only for applications where the object carousel is not present at all.

is_launchable_with_older_version: This 1-bit field is a flag. When set to "1", the GEM terminal shall start the cached application where the version number of the cached application is lower than the version number of the broadcast application. If the version number of the cached application is higher than the version number of the broadcast application, the cached application shall not be started.

If set to "0", the cached application shall not be started. Informative: If this flag is set, the cached application is responsible to handle version conflicts between cached application code and broadcast application data.

not_launchable_from_broadcast: This 1 bit field is a flag. When set to "1" this indicates that the delivery characteristics of this application are such that it is not useful to launch the application unless it has been completely cached/stored. If set to '0' then caching provides some benefit but is not essential.

Applications in stored services and applications where the `not_launchable_from_broadcast` bit is set to "1" shall only be launched where the GEM terminal has stored the complete set of files which are listed in the Application Description File as being critical.

version: This 31 bit field provides the version number of the application. This number starts at zero and increments by one each time any of the files listed in the Application Description File change or the contents of the Application Description File itself changes. Used values shall never be reused, in the event that the number range is exhausted a new application id shall be used.

priority: This field indicates the priority of this application for storage relative to the other applications signalled in this service.

It is only meaningful for applications which have been proactively cached by the GEM terminal implementation and shall be ignored otherwise.

Higher values indicate more important applications to store. The behaviour when applications have the same priority is implementation dependent.

**Table 29: Storage descriptor flag combinations**

| Not_launchable_from_broadcast | Launchable_completely_from_cache | Is_launchable with_older_version | Description |
|---|---|---|---|
| 0 | 0 | 0 | Normal case. |
| 0 | 0 | 1 | Shall not be signalled. |
| 0 | 1 | 0 | Shall not be signalled. |
| 0 | 1 | 1 | Shall not be signalled. |
| 1 | 0 | 0 | Runs if signalled version is stored. |
| 1 | 0 | 1 | Runs if signalled or older version is stored. |
| 1 | 1 | 0 | Runs completely from cache if signalled version is stored. The application cannot be stored due to unavailability of DSM-CC for the current service. |
| 1 | 1 | 1 | Runs if signalled or older version is stored. The application cannot be stored due to unavailability of DSM-CC for the current service. |
| When set, flag indicates that files are present but bitrate is too low. | When set, flag indicates that files are not present in current broadcast at all. | | |

## 10.8.3 Application Description File

### 10.8.3.1 Description

The Application Description File provides the list of files that need to be installed as well as other related necessary information. The notation uses an XML-based syntax.

For those applications that can be stored, an Application Description File file shall be placed in the same carousel as the application.

NOTE: The application description file does not duplicate all the information needed to run the application, the GEM terminal needs to use also the information in the Application Description when installing the application.

Where a file is listed in the Application Description File of more than one application and is stored, the GEM terminal shall ensure that each application sees the correct version of the file for that application. The version of the file visible to one application shall not be changed by any changes in the version of the file visible to any other application which may share that same file.

### 10.8.3.2 Application Description File name and location

The application description file shall be located in the base directory of a DVB-J application (as signalled in the dvb-j application location descriptor). By convention, the name of an ADF is:

```
'dvb.storage.oooooooo.aaaa'
```

where:

```
oooooooo is the organisation id of the application as a 8 character hexadecimal string
aaaa is the application id as a 4 character hexadecimal string
```

The organisation id and application id shall be padded with leading zeros to the specified length.

Lowercase hex digits are used to encode the organisation id and application id.

See also clause 10.8.2, "Application storage descriptor".

### 10.8.3.3 Syntax

The syntax of the Application Description File is defined by the following XML DTD.

The `PublicLiteral` to be used for specifying this DTD in document type declarations of the XML files is:

```
"-//DVB//DTD Application Description File 1.0//EN"
```

and the URL for the SystemLiteral is:

```
"http://www.dvb.org/mhp/dtd/applicationdescriptionfile-1-0.dtd"
```

```
<!ENTITY % object "(dir|file)">

<!-- the main element for the application description -->
<!ELEMENT applicationdescription (%object;)+>

<!ATTLIST applicationdescription version NMTOKEN #REQUIRED>

<!ELEMENT dir (%object;)*>
<!ATTLIST dir
    name CDATA #REQUIRED
    priority NMTOKEN #IMPLIED
>

<!ELEMENT file EMPTY>
<!ATTLIST file
    name CDATA #REQUIRED
    priority NMTOKEN #IMPLIED
    size NMTOKEN #REQUIRED
>
```

### 10.8.3.4 Semantics

version: A decimal integer denoting the version number of this application.

Must not contain leading zeros (unless it is "0"). Must match the version number signalled in the version field of the application storage descriptor in the Application Description entry of this application otherwise the application description file is invalid. (This field allows application authors to ensure that the version number signalled in the Application Description is correct. If it is wrong then this prevents any files being stored.)

name: This attribute provides the name of a file system object (directory or file) that is storable. This is the name of the object within its enclosing directory and hence does not include any directory path information. For the name attribute of a file element only, the last character of the name can be the wild-card character "*". This character will match any string including an empty string.

If name is "." or "..", contains the path separator character "/", or contains the character NUL (U+0000), then receivers shall reject the ADF as invalid.

NOTE 1: No elements are provided for naming object types such as Stream or StreamEvent therefore there is no mechanism to specify that Stream and StreamEvent objects are required to be stored.

NOTE 2: Listing a directory object in the file does not imply anything about those contents of the directory which are not themselves listed in the file.

For requirements on filenames of stored applications, see clause 14.6, "Filename requirements".

priority: This attribute describes how important it is to store this object. The value must be between 0 and 255, inclusive. If it is outside this range then the application description file is invalid. The value zero indicates that it is critical to store the object (i.e. there is no benefit in storing any objects unless this part is stored). Higher values indicate lower storage priority.

The default value for the priority attribute is zero (i.e. critical).

The priority for an object inherits from the immediately enclosing directory.

size: This attribute defines the size in bytes of the file, or files where the name attribute includes a wild-card.

Paths are relative to the directory containing the application description file, i.e. <file> and <dir> elements immediately inside the <applicationdescription> element refer to files and directories in the same directory as the application description file.

## 10.9 Signalling for providers

If a GEM terminal includes support for providers, appropriate signaling for a provider export descriptor and a provider usage descriptor must be defined in the GEM terminal specification.

The provider export descriptor is used in the Application Description for an application which exports (contains) an XletBoundProvider. It declares that the classes of the XletBoundProvider are found in that application.

A provider export descriptor carries information to derive the following information:

provider_name: A string specifying the name of the provider.

provider_class_name: A string specifying the fully qualified name of the class in the application that implements the named provider.

The provider usage descriptor is included in the Application Description for an application which uses an XletBoundProvider and does not include the classes of the provider as part of itself.

A provider export descriptor carries information sufficient to derive the following information:

provider_name: A string specifying the name of the provider.

## 10.10 Signalling for IPTV

### 10.10.1 Service bound application signalling

If the GEM terminal specification adopts SD&S signaling (see TS 102 034 [80]),

service bound applications shall be signalled by including either an ApplicationList or an ExternalapplicationAuthorisedList in either the IPService or the Package elements of SD&S (see TS 102 034 [80]). This is fully specified in annex AR "(normative): XML encoding for AIT" of the present document.

### 10.10.2 XAIT

If the GEM terminal specification adopts SD&S signaling (see TS 102 034 [80]), unbound applications shall be signalled by including an ApplicationList in the SD&S service provider discovery record for the subsidising service provider. This is fully specified in annex AR, "XML encoding for AIT". Any ApplicationList elements in the service provider discovery records for other service providers shall be ignored.

Monitoring for changes in the XAIT shall be performed as defined in clause 5.4.3 of TS 102 034 [80].

NOTE: See org.ocap.application.AppSignalHander for notification of XAIT updates to applications.

# 11 DVB-J platform

## 11.1 The virtual machine

The Java platform is defined in PBP 1.1 [4].

## 11.2 General issues

### 11.2.1 Basic Considerations

Unless otherwise specified, GEM implementations are not required to include any classes or methods marked as deprecated in those API specifications which are either referenced by or included in the present document. Where a class is defined by the present document as implementing a specific interface, and that interface requires the class to provide an otherwise deprecated method, the interface overrides the deprecated mark and the method is required by the present document.

Each DVB-J application instance is considered to logically run in its own virtual machine instance. For this reason, it cannot rely on finalizers that are defined in application classes being run when the application terminates. When the application manager terminates the entity that represents the virtual machine in which the application is run, a conformant implementation is permitted to not run application finalizers, as spelled out in section 2.17.9 of the Java virtual machine specification, second edition [10].

DVB-J applications shall not synchronize on system classes or other exposed system static objects else undefined behaviour may occur.

`SecurityExceptions` shall only be thrown either where they are specified by the defining document or where identified in the present document for selected references which do not include any `SecurityExceptions`.

Unless specified otherwise for a referenced specification, it is an allowable implementation choice to not override methods as long as the defined semantics are respected. The implication of this is that such differences between implementations will be visible when using the `java.lang.reflection` package.

The inclusion of java packages shall not imply the inclusion of other subpackages, e.g. the inclusion of the `javax.tv.media` package does not automatically imply the inclusion of the `javax.tv.media.protocol` subpackage. Inclusion of subpackages shall be explicitly listed in the present document.

DVB-J applications shall not use additional public or protected methods or fields in the `org.dvb` namespace which are not listed in the present document. Applications which scale to run on multiple revisions of the present document shall only use `org.dvb` methods etc. appropriate to the version of the platform on which they are running.

Applications shall not define classes or interfaces in any package namespace defined in the present document. GEM terminals shall enforce this using the `SecurityManager.checkPackageDefinition` mechanism.

NOTE: DVB-J applications that are written to be scalable across multiple GEM profiles and/or versions of profiles and/or optional features may include references to API classes that are only present if the GEM terminal implements the profile or version of a profile or an optional feature that these API classes represent. Such applications have to be written with care using a standard Java idiom for this case. This will ensure that they work on all MHP and GEM terminal implementations, including those that use "eager linking" technology. A simple idiom to achieve this is presented in clause W.3, "Example of testing for optional APIs".

### 11.2.2 Approach to Subsetting

Where a class included in the present document has methods, fields or constructors with signature dependencies on classes not included in the implemented profile, these methods are not required to be present in an implementation. A compliant implementation choice may require these methods and classes to be present.

Where the present document subsets a package, inclusion of the complete package is allowed but clearly not required. The behaviour of the additional features is not specified for broadcast applications.

## 11.2.3 Class Loading

### 11.2.3.1 Fundamental principles

The DVB-J application environment shall be written such that each application appears to run within its own classloader or classloader hierarchy for all classes that are not a part of the platform. As a consequence, two applications will never be able to access the same copy of any application-defined static variable.

In a signed application, all classes or files to be loaded through the classpath shall be signed by at least the set of certificates used to sign the initial xlet class of the application. This applies, for example, to class files comprising the application, and images and other data loaded via the `java.lang.Class.getResource()` mechanism. When authenticating a signed application, a GEM terminal can select any one of these certificates to use to authenticate all subsequent classes or files loaded. The mechanism used to make this selection is implementation dependent.

Where a jar file is included in the search path (e.g. for a `DVBClassLoader`) and is signed according to the GEM security model, all classes or files to be loaded from that jar file shall be considered to be signed by the certificates signing the jar file under the GEM security model.

NOTE 1: GEM imposes no requirements on the use of the signing information within a jar file.

NOTE 2: Where a GEM terminal trusts more than one of the certificates used to sign the initial xlet, it should attempt to select the most trusted of these.

See clause 12, "Security".

Inappropriate, meaningless or illegal locators are silently skipped and searching continues.

### 11.2.3.2 Class loading and providers

For GEM terminal specifications that include the optional provider APIs (see clause 11.7.9), the above requirement for the classes to be signed with a specific certificate is modified as follows: The classes of the provider shall be signed by a certificate where the `organisation_id` in the subject field of the certificate matches the `organisation_id` in the provider name.

## 11.2.4 Unloading

NOTE: Class unloading as required by PBP 1.1 [4] is required.

## 11.2.5 Event listeners

In `org.dvb` and `org.davic` all methods to remove event listeners shall have no effect if the listener is not registered.

NOTE: The number of threads used to send events to event listeners is intentionally implementation dependent. Applications should not block in event listeners as this may prevent other events being delivered.

## 11.2.6 Event model in DAVIC APIs

Events defined in DAVIC APIs DAVIC 1.4.1p9 [17] which currently directly extend `java.lang.Object` shall extend `java.util.EventObject`. Event listeners defined in DAVIC 1.4.1p9 [17] shall extend `java.util.EventListener`.

## 11.2.7 Event model in DAVIC and DVB APIs

Each class in `org.dvb` and `org.davic` inheriting from `java.util.EventObject` is just a container for these fields and no validity checks are done for the parameters by this constructor. Instances of these classes are intended to be constructed by the platform implementation and not by applications. The platform implementation will only construct these events with the appropriate information passed in.

In `org.dvb` and `org.davic`, unless explicitly specified otherwise, all methods to add event listeners add each listener only once if the add method is called with the same parameters multiple times. This means that the same event is delivered only once to each listener even if it has been added twice.

## 11.2.8 Tuning as a side-effect

This clause applies only to GEM terminal specifications which include a tuner.

No GEM API shall cause tuning unless explicitly specified as such. For example, if a locator that requires tuning is received by the `DVBClassLoader` it shall behave as if the specified file is not available.

## 11.2.9 Intra application media resource management

Where an application makes conflicting requests for limited media decoding resources, the media decoding resources that are requested most recently are presumed to be the ones that are most wanted. This applies between MPEG-2 I-Frames, MPEG-2 Video "drips" and streaming video. Similarly, this applies between streaming audio and audio from memory.

When a non-broadcast media presentation (audio or video from memory or still image) is interrupted by a resource loss within the same application, the first presentation is cancelled and will not be restored.

If a broadcast media presentation is interrupted by a resource loss within the same application, the broadcast presentation is restored when the interrupting presentation ends.

## 11.2.10 Application thread priority

The `ThreadGroup` of application threads shall have a `MaxPriority` of `java.lang.Thread.NORM_PRIORITY`.

NOTE: As a consequence applications will not be able to create threads at priorities greater than `java.lang.Thread.NORM_PRIORITY` since they don't have `java.lang.RuntimePermission( "modifyThread" )`. Applications may perform compute intensive tasks within application created threads without being considered unresponsive.

## 11.2.11 Text Encodings

Where the specification of the Java APIs refers to the default character encoding of the platform, the default for GEM shall be "UTF8" as defined in clause 7.1.5, "Monomedia format for text". The encoding "latin1" shall also be supported as defined in ISO 8859-1 [23].

When present in a Java `String`, the mark-up codes defined in table 80, "Codes defined for use in marked-up text files" shall be encoded in Java chars whose most significant byte is zero and whose least significant byte is the value from table 80. The encoding `"DVBMarkupUTF8"` shall be supported and is defined to be the same as UTF8 except as follows:

- In byte to char translations, the mark-up sequences in table 80 shall be translated into chars as defined above.
- In char to byte translations, sequences of characters matching the encodings above shall be translated into the corresponding mark-up code sequences in table 80.

### 11.2.11.1 Text encoding in Service Information

This clause applies only for GEM terminal specifications that include the functional equivalent named "SI" as defined in clause 15.6, "Functional equivalents".

Where methods of the DVB SI API or Java TV APIs access strings encoded in the SI tables and return them to applications as String objects, the following character encodings shall be supported as defined in annex A of the DVB SI specification EN 300 468 [14]:

- ISO 6937 [22] (default).
- ISO 8859 [23] through ISO 8859 [23] (SI string first byte codes 0x01...0x05).

- ISO 8859-1 [23] through ISO 8859-9 [23] (SI string first byte code 0x10).
- 16-bit ISO/IEC 10646-1 [15] UCS-2 (SI string first byte code 0x11).

These encodings shall also be supported by the methods in the `org.dvb.si.SIUtil` class. Support of the other character encodings whose signalling is specified in the DVB SI specification is not required from GEM terminals.

# 11.3 Fundamental DVB-J APIs

The present document does not require a particular text encoding for locators, however terminal specifications are required to define such a text encoding. The entities for which a text encoding is required are specified in clause 14.8.

Where a locator text encoding is required, a locator may be constructed from the text representation using the factory method defined in the class `javax.tv.locator.LocatorFactory`.

NOTE: Portable GEM applications should not contain hard-coded text representations for locators, as it is likely that the locators will vary across networks. If an application needs to be signalled with values for locators, they can be passed in as Xlet arguments, or put in a small text file that is read from the carousel.

## 11.3.1 Java platform APIs

NOTE: The following packages are defined in PBP 1.1 [4].

### 11.3.1.1 java.lang package

The `java.lang` package is supported with the following notes and requirements.

a) The following fields shall not be used by inter-operable applications:

- `System.in`.

b) Applications shall be able to use:

- `System.out`;
- `System.err`;
- `Runtime.traceInstructions()`;
- `Runtime.traceMethodCalls()`;

for debugging without any adverse effects to the application. The output shall not be visible to normal end users and shall not conflict with any other API.

c) The `java.lang.Compiler` class and following methods shall be taken as hints from an application to the system. However, as specified by PBP 1.1 [4] there is no guarantee of what happens:

- `Runtime.gc()`;
- `System.gc()`.

d) In addition to the system properties required by PBP 1.1 [4], the following properties are required to be supported for `System.getProperty()`:

- `dvb.returnchannel.timeout` (see clause 11.5.3, "Support for IP over the Return Channel");
- `dvb.persistent.root` (see clause 11.5.6, "Persistent Storage API").
- `dvb.display.aspect_ratio` (see clause 11.4.1.2, "TV user interface").

With the exception of `dvb.persistent.root` all of these properties are accessible to all applications.

`dvb.persistent.root` is accessible only as defined in clause 11.10.2.1, "java.util.PropertyPermission".

NOTE 1: All GEM terminal specifications require support for the system property "dvb.persistent.root".

Property names beginning "dvb." are reserved for future use.

e) The methods java.lang.SecurityManager.checkPackageDefinition and java.lang.SecurityManager.checkPackageAccess are documented to check a package name against a list of restricted packages.

GEM terminal implementors are recommended to use this mechanism to prevent application classes from having access to implementation classes where this could compromise the security of the GEM terminal.

GEM terminals shall not define classes in the empty ("") package name-space.

NOTE 2: GEM terminal implementors who are also implementing GEM applications need to ensure that the namespaces used for applications do not collide with those used for their GEM implementation.

The call System.currentTimeMillis() shall feature a granularity of the returned time value of not more than 10 ms. The terminal should attempt to keep an accurate clock, e.g. by initialising its clock from a TDT. The default TimeZone for java.util.Calendar shall either be as defined by the end-user or as defined in a TOT if the end-user has not defined any TimeZone. With the call Object.wait(long timeout), the timeout value is specified as a maximum time to wait. The present document further guarantees that if not notified the object will wait for at least timeout 10 ms.

#### 11.3.1.2 java.void

As specified in PBP 1.1 [4], the constants in the java.util.Locale class do not imply support (or otherwise) for these Locales. Locales supported in GEM are specified in profiles (see clause 15.4, "Locale support").

Inter-operable applications shall not call the java.util.TimeZone.setDefault method. The behaviour if this method is called is implementation dependent.

#### 11.3.1.3 Void

#### 11.3.1.4 java.io, javax.microedition.io

The deprecated method java.io.ObjectInputStream.readLine() shall not be called by inter-operable applications.

The classes and interfaces in this package relating to files and file systems have additional semantics defined for GEM specific file systems as follows.

- For broadcast carousels in clause 11.5.1.1, "Constraints on the java.io.File methods for broadcast carousels".
- For persistent storage in clause 11.5.6, "Persistent Storage API".
- For applications running as part of stored services in clause 11.12.2.2, "Modified behaviour of GEM 1.0 APIs".

NOTE: PBP 1.1 [4] requires that javax.microedition.io support a number of protocols, such as "socket:", "http:" and "file:".

#### 11.3.1.5 java.net

Consistent with PBP 1.1 [4], support is required for at least one implementation dependent java.net.URL protocol. Platform methods that return a URL to access resources, such as Class.getResource, ClassLoader.getSystemResource and ClassLoader.getResource shall return instances of java.net.URL using such a protocol where no appropriate protocol is defined in the present document.

In a signed application, a URL to this resource shall be returned as for an unsigned application, regardless of whether the underlying file is signed. When that file is accessed (e.g. via an input stream obtained from java.net.URL.openStream()), then if the file fails authentication as described in clause 11.2.3, "Class Loading", the system shall behave as if the file contained no data (i.e. as if it were a zero-length file).

NOTE: Due to the overhead of processing the signature verification, asset files which are not critical to be authenticated should be loaded using `java.io.File` or `org.dvb.dsmcc.DSMCCObject` and sent as unsigned.

Support is required for the "`file:`" protocol. Applications shall be able to construct instances of `java.net.URL` using strings containing "`file:`" URLs as defined in RFC 1738 [61] where the `<host>` element is the empty string and the <path> element is an absolute filename.

The return types of `URL.getContent()` are defined by the mappings from data type to java class name listed in table 30.

**Table 30: Return types of URL.getContent()**

| Data type | Return type |
|---|---|
| Unknown or unsupported data types | java.io.InputStream |
| Text/plain | java.io.InputStream |
| Text/vnd.dvb.utf8 | java.io.InputStream |
| Text/Generic | java.io.InputStream |
| Image/png | java.awt.image.ImageProducer |
| Image/jpeg | java.awt.image.ImageProducer |
| Image/mpeg | org.havi.ui.HBackgroundImage |

The behaviour of `URL.getContent()` responds to data type using information with the following priority:

a) Content type signalling such as:

- The content type descriptor in the OC (see clause B.2.3, "Content type descriptor").
- The HTTP header (if supported in the profile) the Content-Type header field (if present).

b) The filename extension (if known) (see table 7, "File type identification").

c) Open the file and study.

For GEM terminal specifications that do not include a functional equivalent named "Carousel" as defined in clause 15.6, "Functional equivalents" this requirement regarding the behaviour of `URL.getContent()` does not apply; however, if the functionally equivalent signalling contains data type information, it is recommended that it be given the same priority as the content type descriptor is given.

`getContentType` returns the value contained in the optional content type descriptor in the OC (see clause B.2.3, "Content type descriptor") if present.

`getFileNameMap` returns information derived from table 7 "File type identification".

See the Object Carousel signalling described in clause B.2.3, "Content type descriptor".

## 11.3.2 GEM platform APIs

### 11.3.2.1 org.dvb.lang

The `org.dvb.lang` package is supported as defined in annex I, "DVB-J fundamental classes".

Where no parent is specified at creation, the delegation parent classloader of `DVBClassLoader` shall be the original classloader of the calling application.

Classloader delegation is defined in the specification for `java.lang.ClassLoader` in PBP 1.1 [4].

### 11.3.2.2 org.dvb.event

#### 11.3.2.2.1 Generic description

The `org.dvb.event` package is supported as defined in annex J, "DVB-J event API".

When sending events from a `org.dvb.event.EventManager` to listeners, the implementation shall ensure that any single listener shall only be sent one platform generated event instance at one time. If a new event is generated before the listener method has returned from processing a previous event, the GEM terminal shall not call that listener method until the call for processing the previous event returns.

The behaviour where applications have multiple such event listeners registered is implementation dependent. Implementations may use multiple threads to send the same event instance to any such multiple listeners. Implementations may ensure that at most one event listener in any one application for all these events is executing at any one time.

The return value of `UserEvent.getWhen()` shall be the difference, measured in milliseconds, between the time when the event occurred and midnight, January 1, 1970 UTC, i.e. like `System.getCurrentTimeMillis()`.

#### 11.3.2.2.2 Additional semantics for org.dvb.event

Privileged applications (and no others) shall be able to exclusively reserve the user input events defined in table 87 "Additional input events required by privileged applications". Attempts by applications without this permission to reserve these user input events shall fail. These events shall not be delivered via the conventional focus mechanism.

### 11.3.3 Java TV

The Java TV APIs are defined in Java TV [16].

NOTE: This includes the packages `javax.tv`, `javax.media` and their subpackages.

## 11.4 Presentation APIs

### 11.4.1 Graphical User Interface API

#### 11.4.1.1 The Core GUI API

Property names for use with the `getProperty` method on `java.awt.Image` and its sub classes beginning "`dvb.`" are reserved for future use.

Applications shall be able to use Toolkit.beep without any adverse effects to the application. The output is not required to be audible to normal end users and shall not conflict with any other API.

The methods `getScreenResolution` and `getScreenSize` shall be supported with the additional semantics described in HAVi [50].

The encoding of image content types for use by `java.awt.image` are defined in clause 7.1.1, "Bitmap image formats". The set of formats supported is profile dependent.

When using the `java.awt.FontMetrics` class, the width of a set of characters or string returned by the `charsWidth` or `stringWidth` method shall be correct taking into account any kerning and sub-pixel positioning applied by the font renderer. Calculating the same number by adding the widths of the individual characters is not required or expected to return the same number since it will not take into account any kerning or sub-pixel positioning applied by the font renderer.

The downloading of fonts from the network (see clause D.2.2, "Downloaded fonts") shall be supported using the methods concerned on `org.dvb.ui.FontFactory`. Failure to download a font shall be reported by these methods. The constructor for `java.awt.Font` shall only be aware of resident fonts, as described in clause 7.3, "Resident fonts". These fonts are available e.g. to the AWT APIs without using the downloadable font API specified in annex U, "Extended graphics APIs".

For GEM terminals, the time at which an input event occurs as reported by `java.awt.event.InputEvent.getWhen` does not mean the time at which the event is reported to a listener of a GEM application. In particular, if a GEM application has a backlog in processing events and events are being queued, the time at which the event occurred shall be before the event enters the queue and hence also before when it leaves the queue to be reported to the GEM

application's listener. The return value of `getWhen()` shall be the difference, measured in milliseconds, between the time when the event occurred and midnight, January 1, 1970 UTC, i.e. like `System.getCurrentTimeMillis()`.

In implementations of GEM, the "single instance of an application-created Frame" that "is permitted per GraphicsDevice" will be created by the GEM implementation for the default `HGraphicsDevice`. Attempts by GEM applications to construct instances of Frame for the default `HGraphicsDevice` will cause the constructor to throw `java.lang.UnsupportedOperationException`.

#### 11.4.1.2 TV user interface

The packages `org.havi.ui` and `org.havi.ui.event` defined in HAVi [50] shall be supported with the following modifications. The following HAVi UI widgets and supporting classes and interfaces are not required by GEM:

- `org.havi.ui.HAnimateEffect`
- `org.havi.ui.HAnimateLook`
- `org.havi.ui.HAnimation`
- `org.havi.ui.HDefaultTextLayoutManager`
- `org.havi.ui.HEventMulticaster`
- `org.havi.ui.HFlatEffectMatte`
- `org.havi.ui.HFlatMatte`
- `org.havi.ui.HGraphicButton`
- `org.havi.ui.HGraphicLook`
- `org.havi.ui.HIcon`
- `org.havi.ui.HImageEffectMatte`
- `org.havi.ui.HImageMatte`
- `org.havi.ui.HListElement`
- `org.havi.ui.HListGroup`
- `org.havi.ui.HListGroupLook`
- `org.havi.ui.HMultilineEntry`
- `org.havi.ui.HMultilineEntryLook`
- `org.havi.ui.HRange`
- `org.havi.ui.HRangeLook`
- `org.havi.ui.HRangeValue`
- `org.havi.ui.HScreenDimension`
- `org.havi.ui.HSinglelineEntry`
- `org.havi.ui.HSinglelineEntryLook`
- `org.havi.ui.HStaticAnimation`
- `org.havi.ui.HStaticIcon`
- `org.havi.ui.HStaticRange`
- `org.havi.ui.HStaticText`

- org.havi.ui.HSwitchable
- org.havi.ui.HText
- org.havi.ui.HTextButton
- org.havi.ui.HTextLook
- org.havi.ui.HToggleButton
- org.havi.ui.HtoggleGroup

If the optional support for MPEG-2 I-Frames is not included, all attempts to request an instance of org.havi.ui.HStillImageBackgroundConfiguration shall fail. The class itself shall still be present.

The following text shall be considered to be present on the end of the method description of org.havi.ui.HScene.setVisible:

- If this HScene is already visible, then this method brings it to the front. The semantics of show() and setVisible(true) are identical.

  This behaviour is optional in GEM terminals.

Instances of org.havi.ui.event.HRcEvent are reported through the normal java.awt event mechanism due to the inheritance from java.awt.event.KeyEvent.

With the exception of the HSound.load method, no methods specified in HAVi [50] shall throw a security exception in the GEM context. The permissions for HSound.load are those defined for java.io.

The following semantics shall be used for the getVideoController method on HVideoDevice.

- It shall only return JMF players (see javax.media in Java TV [16]) which are in the Prefetched or Started states and which are using that HVideoDevice as one of their scarce resources. Otherwise null will be returned. It shall not return JMF players from other applications if those are using the video device underlying the HVideoDevice.
- Except as specified below, it shall only return JMF players which have been already created in response to the application calling javax.media.Manager.createPlayer or which have been returned by javax.tv.service.selection.ServiceContext.getServiceContentHandlers (see Java TV [16]).
- The only exception to the above is the situation where video is being played in the background as part of the context of an application but where javax.tv.service.selection.ServiceContext.getServiceContentHandlers has not yet been called. In this case, getVideoController shall return the same JMF player as would be returned by getServiceContentHandlers if it was to be called subsequently.

The signatures of the classes HComponent and HContainer shall be extended with "implements org.dvb.ui.TestOpacity".

The methods HGraphicsConfiguration.getPunchThroughToBackgroundColor shall not be used by inter-operable applications.

The methods fontAvailable() and downloadFont() in the class org.havi.ui.HFontCapabilities shall not be used by inter-operable applications.

NOTE 1: The package javax.tv.graphics is required, as per clause 11.3.3, "Java TV".

Applications shall only pass in calls to the method javax.tv.graphics.TVContainer.getRootContainer the exact same XletContext instance as was passed to their initXlet method when it was called by the GEM terminal. The behaviour of getRootContainer if passed any other XletContext is implementation dependent and returning null is one valid option. Except for this case, this method shall never return null in a GEM implementation. The methods javax.microedition.xlet.XletContext.getContainer and javax.tv.graphics.TVContainer.getRootContainer shall return the same as would be returned by a call to org.havi.ui.HSceneFactory.getDefaultHScene() under the same circumstances.

The method HComponent.isDoubleBuffered shall not be used by inter-operable applications.

Any changes made to the org.havi.ui.HGraphicsConfiguration of an org.havi.ui.HGraphicsDevice shall be reported via the java.awt.GraphicsDevice and java.awt.GraphicsConfiguration APIs, as appropriate. For each HGraphicsConfiguration, there shall be a GraphicsConfiguration where getBounds() returns the same values as HGraphicsConfiguration.getPixelResolution().

NOTE 2: Application developers should not rely on GEM terminals supporting full-screen exclusive mode, thus, as permitted by Personal Basis Profile, the method java.awt.GraphicsDevice.isFullScreenSupported may always return false.

The system property dvb.display.aspect_ratio shall indicate the shape of the physical display that the GEM terminal believes is used by its default HScreen.This may not be the same as the aspect ratio of the receiver's current output signal. This shall be encoded as the ratio of width to height, both expressed as positive decimal integers and separated by a ":", e.g. "4:3" or "16:9". This shall be the same ratio as would be returned by VideoFormatControl.getDisplayAspectRatio for a JMF player decoding on that HScreen.

#### 11.4.1.3 Extended graphics

See annex U, "Extended graphics APIs".

The class org.dvb.ui.DVBTextLayoutManager is not required by the present document.

#### 11.4.1.4 Television viewing mode

GEM includes two ways for applications to receive input events: the normal AWT method in java.awt.Component and the org.dvb.event package (see clause 11.3.2.2, "org.dvb.event"). Via the normal AWT method, the application normally receives all input events when the component has the focus. The minimum set of input events that are supported is defined in clause G.5, "Input events".

Often the resident navigator software of the GEM terminal uses many of the keys on the remote control for its own navigation purposes. The navigator shall not act on input events which are delivered to GEM applications. In particular, the navigator shall not respond to number key input if that input is delivered to one or more GEM applications. Conversely, if the GEM application with input focus has indicated disinterest in number key input (using HScene.setKeyEvents with an HEventGroup not including the number keys) then navigators which respond to number key input are free to do so. Rules on when GEM applications should request interest in particular input events and when they should not are found below.

To ensure consistent user experience, the following rules are defined:

- An application creating an HScene and placing components into it shall not by default get the input focus for these components.
- The application may request to get the input focus by calling Component.requestFocus(). If this is granted and the focus moved to the requested component, this component shall receive input events as defined in clause J.1.
- The application may request to receive a subset of input events via the org.dvb.event API even when not having the AWT focus.

For applications delivered within normal television services, it is recommended that the following items are taken into account:

- When the display is primarily showing the video of the television service and the user perception is that he is not actively interacting with an application but is just watching the television service (called 'television viewing mode'), the applications should not request the AWT focus but let most of the input events go to the resident navigator (e.g. number keys, directional arrow keys and Enter may cause the normal actions the user expects from the navigator).
- In the 'television viewing mode' the applications should request only the minimum required input events, e.g. only input events that cause some further action to happen and that may transition the user from the 'television viewing mode' into a state where the user more actively interacts with the applications.

- If the applications require some input events for the user to be able to activate the application when in the 'television viewing mode', it is recommended that the VK_COLORED_KEY_(0 to 3) and VK_TELETEXT (see note below) input events are used for this purpose and the applications request them via the org.dvb.event API. In practice this means that the navigator can not map any essential function directly to the VK_COLORED_KEY(0 to 3) events in television viewing mode.

- Subtitles should be available when end-users perceive themselves as watching television. In order to achieve this, GEM applications shall not have a visible full screen HScene where more than 60% of the pixels are wholly transparent to video when in "television viewing mode".

- GEM applications wishing to leave "television viewing mode" and receive events other than the VK_COLORED_KEY_(0 to 3) and VK_TELETEXT are obliged to display a non-transparent graphics object on the screen that covers at least 3 % of the visible area on the screen (equivalent to 9 852 pixels on a 634 x 518 visible screen in a 720 x 576 or 720 x 480 graphics mode) which clearly indicates to the user the "non-navigator" mode of the receiver with respect to input event handling. (Obviously, the pixel values given only apply to systems with the indicated resolution; on other systems, the 3 % requirement applies, but results in different pixel values.)

NOTE 1: In television services where DVB Teletext is transmitted within the service, a typical navigator action in the 'television viewing mode' can be to activate a teletext decoder, if supported by the terminal. The application requesting this input event may cause the DVB Teletext decoder not to be conveniently accessible to the end user. Therefore, within services that carry DVB Teletext it is recommended that the VK_TELETEXT input event is requested in the 'television viewing mode' only by such applications that provide a simulcast GEM application version of the teletext service, thus providing a replacement for the DVB Teletext information.

In environments where WST Teletext services are prevalent when there is no DVB Teletext service associated with the video service, the GEM application should display an information service of a similar nature to teletext. When receiving this event for the second time the information service should terminate and the user will return to normal viewing mode.

The VK_TELETEXT event should have no other purposes in GEM applications.

NOTE 2: The items above apply only to the 'television viewing mode' as explained above. In states where the user experience is that the user is clearly primarily interacting actively with an application the applications can naturally use the normal AWT input event mechanism and request the focus as well as use any of the supported input events via the org.dvb.event API.

NOTE 3: For data services, the first interaction state of an application that is automatically started after service selection should behave as television viewing mode. This is in order to allow users who are navigating with the navigator to seamlessly navigate through these services without getting confused.

NOTE 4: As a consequence of clause 4.1.4, "Addition of non-GEM interfaces" the requirements of this clause are required of all GEM-based terminal specifications; as a further consequence, GEM terminal specifications are not permitted to define extensions that have to be invoked by applications in order to obtain the behaviour mandated by the present document.

NOTE 5: The method java.awt.event.KeyEvent.getKeyChar() may return java.awt.event.KeyEvent.CHAR_UNDEFINED only for KEY_PRESSED and KEY_RELEASED events, never for KEY_TYPED events. In no case should an interoperable application rely on the return value of getKeyChar() for a KEY_PRESSED or KEY_RELEASED event; further, the value returned for an undefined key is not defined interoperably and may vary across platform implementations.

### 11.4.1.5 Font bindings

#### 11.4.1.5.1 PFR0

For fonts in the PFR0 format clause 7.4.1, "PFR", the bindings between the Java APIs relating to font metrics and the font format itself shall be as follows;

The return values of the methods FontMetrics.getMaxAscent and FontMetrics.getMaxDescent shall be yOffsetTop and yOffsetBottom respectively as defined in clause D.3.5.3, "Conversion of units".

For PFR fonts with the auxiliary data record:

- `FontMetrics.getAscent` shall be obtained from the `ascent` field.
- `FontMetrics.getDescent` shall be obtained from the `descent` field.
- `FontMetrics.getLeading` shall be obtained from `externalLeading`.

  All values in the auxiliary data record shall be converted from font metrics units to pixels as described in clause D.3.4, "Converting font metrics to display pixels" before being returned. This conversion shall round up.

For PFR fonts without the auxiliary data record:

- `FontMetrics.getAscent` shall be the same as `getMaxAscent`.
- `FontMetrics.getDescent` shall be the same as `getMaxDescent`.
- `FontMetrics.getLeading` shall be zero.

The "advance width" of a character, in metrics units, is defined as follows:

- For PFR fonts with the "`proportionalEscapement`" flag set, the advance width of a character is given by the "`charSetWidth`" field in the character record.
- For PFR fonts without the "`proportionalEscapement`" flag set, the advance width of a character is given by the "`standardSetWidth`" field in the physical font record.

`java.awt.FontMetrics.charWidth()` shall return the advance width of the character converted to pixels. This conversion shall round up.

`java.awt.FontMetrics.stringWidth()`, `java.awt.FontMetrics.bytesWidth()` and `java.awt.FontMetrics.charsWidth()` are calculated by summing the advance widths of all the characters in the string, and adjusting for any kerning in the font. Adjustments for kerning are performed in metrics units, then the result is converted to pixel units. This conversion shall round up.

`java.awt.FontMetrics.getMaxAdvance()` returns the maximum value that `java.awt.FontMetrics.charWidth()` can return.

NOTE: For proportional fonts the implementation calculates this value from the advance width of every character in the font; there is no field in the PFR file which contains it.

#### 11.4.1.5.2 OpenType

For fonts in the OpenType format as described in clause 7.4.2, "OpenType", the binding between the Java API relating to font metrics and the font format itself shall be as follows.

The return values of the methods `FontMetrics.getMaxAscent`, `FontMetrics.getMaxDescent`, `FontMetrics.getAscent`, `FontMetrics.getDescent and FontMetrics.getLeading` shall be as follows:

- `FontMetrics.getMaxAscent` shall be obtained from the `yMax` field of the `Font Header ('head')` table;
- `FontMetrics.getMaxDescent` shall be obtained from the `yMin` field of the `Font Header ('head')` table, and shall return `-yMin`;
- `FontMetrics.getAscent` shall be obtained from the `sTypoAscender` field in the platform-independent OS/2 tables.
- `FontMetrics.getDescent` shall be obtained from the `sTypoLineGap` field in the platform-independent OS/2 tables.
- `FontMetrics.getLeading` shall be obtained from the `LineGap` field in the `Horizontal Header ('hhea')` table.

The results of `java.awt.FontMetrics.stringWidth()`, `java.awt.FontMetrics.bytesWidth()` and `java.awt.FontMetrics.charsWidth()` return the total advance width for rendering the specified object and may account for advanced typesetting features.

NOTE: The total advance width returned from these methods is affected by the layout engine, which might use advanced typesetting features like kerning or ligatures. The advance of a sequence of characters is not necessarily the sum of the advances of its characters.

## 11.4.2 Streamed Media API

### 11.4.2.1 Framework of solution

The `javax.media` and `javax.media.protocol` packages from Java TV [16] shall be implemented with the clarifications, extensions and restrictions as defined in the corresponding clauses below.

### 11.4.2.2 Clarifications

The JMF "time base time" when playing MPEG content delivered in MPEG transport streams is used as a constantly progressing time whose rate may be synthesized from the Program Clock References / the System Time Clock in MPEG or by some other appropriate method. The value does not have any direct relation to the value of the MPEG System Time Clock in the receiver.

The media time of JMF when playing MPEG content delivered in MPEG transport streams is just a time value that progresses as the media stream is played, but whose actual value is implementation dependent.

NOTE: There is no implied or expected connection between the JMF media time and any time base(s) provided by the MPEG-2 / DSM-CC Normal Play Time.

When creating a JMF player, Media locators and URLs which reference a service, event or elementary stream will create a player which plays the content concerned direct from the network. Media locators and URLs which reference files will create a player which will download the content concerned from the network during the Prefetching state of the player concerned. If the content cannot be downloaded then such players will never enter the Prefetched state.

Interoperable applications shall not call `javax.media.MediaHandler.setSource()`.

In the `javax.media.PackageManager`:

a) The effect of the `set<xxx>PrefixList` methods is limited to the application calling the method.

b) The `SecurityException` of the `commit<xxx>` methods shall always be thrown in GEM.

The events `javax.media.ControllerEvent` and `javax.media.GainChangeEvent` shall inherit from `java.util.EventObject`.

The `interfaces javax.media.ControllerListener` and `javax.media.GainChangeListener` shall inherit from `java.util.EventListener`.

`SubtitlingLanguageControl.isSubtitlingOn()` returns the current state of subtitle presentation and NOT the last value set with `setSubtitling`. In particular `setSubtitling` may be set to true but if there are no subtitles in the network or no subtitle service component selected then `isSubtitlingOn` shall return false.

Any elements in `DVBLocators` after and including the `event_id` element are ignored by JMF players.

NOTE: For GEM terminal specifications that do not include the functional equivalent named "Content Referencing" as defined in clause 15.6, "Functional equivalents", this paragraph does not apply. Instead, the present document requires that any information in a locator beyond that identifying a service (e.g. the time of a specific program event) is to be ignored by JMF players. See also clause 14.8, "Locators and content referencing".

#### 11.4.2.3 Default media player behaviour

For a JMF player which is presenting a service, the following rules will be followed in the order given to decide which service components will be presented when multiple audio, video or subtitle components are present in a service:

a) For audio and subtitle streams in different languages, user preferences will be used to determine which streams are selected.

b) The streams which are first in the network signalling information (i.e. in the PMT) will be presented. GEM terminal specifications may define additional mechanisms that override this default.

If any of the media components comprising the service change then the implementation will as far as possible replace the changed components with suitable replacements in a user preference and implementation dependent manner.

#### 11.4.2.4 Required controls for video drips

The following controls are supported for GEM terminal specifications that support video drips (see clause 7.1.3, "MPEG-2 Video "drips""):

- `javax.tv.media.AWTVideoSizeControl`
- `org.dvb.media.BackgroundVideoPresentationControl`

#### 11.4.2.5 Extensions to the Framework

##### 11.4.2.5.1 DVB specified extensions

The classes and interfaces defined in annex N, "Streamed media API extensions" of the present document are included, with the exception of the following:

- `org.dvb.media.SubtitlingEventControl;`
- `org.dvb.media.SubtitleAvailableEvent;`
- `org.dvb.media.SubtitleListener;`
- `org.dvb.media.SubtitleNotAvailableEvent;`
- `org.dvb.media.SubtitleNotSelectedEvent;`
- `org.dvb.media.SubtitleSelectedEvent;`
- `org.dvb.media.CAStopEvent;`
- `org.dvb.media.CAException.`

If a `Player` bound to a `DripFeedDataSource` receives a `ResourceWithdrawnEvent` and later a `ResourceReturnedEvent`, the video output from the video decoder to which the player is bound is implementation dependent. The GEM application using the `Player` shall call the `DripFeedDataSource.feed` method with an I-frame as soon as possible after `ResourceReturnedEvent` is received.

NOTE: Application developers should take this into account when constructing video drip sequences. A large number of P-frames between I-frames could lead to a significant delay before the visible display can be refreshed due to the limitations on the frequency with which new data can be fed to the decoder.

##### 11.4.2.5.2 Extensions in org.davic

The following classes and interfaces will be included from the `org.davic.media` package, as defined in annex L of DAVIC 1.4.1p9 [17].

- MediaPresentedEvent
- MediaLocator
- MediaTimePositionControl
- ResourceWithdrawnEvent
- FreezeControl
- ResourceReturnedEvent
- MediaTimePositionChangedEvent
- LanguageNotAvailableException
- NotAuthorizedException
- MediaFreezeException
- LanguageControl
- AudioLanguageControl

For the packaged media target, the following classes are not required by the present document:

- org.davic.media.FreezeControl
- org.davic.media.MediaFreezeException

The following class is required for terminal specifications that include the conditional access API:

- org.davic.media.NotAuthorizedMediaException

The following classes will be included from the org.davic.media package as defined in annex L of DAVIC 1.4.1p9 [17] with the following semantic modification:

- The completion of the action started by calling setMediaTimePosition() on the MediaTimePositionControl will be signalled by a org.davic.media.MediaTimePositionChangedEvent.
- The org.davic.media.MediaPresentedEvent shall only be thrown following changes caused using controls in the org.davic package and as a consequence of the transition of a JMF player into the Started state. Instances of this event shall be generated as close to the presentation of the content as the device can detect, e.g. when the new content leaves the output of a hardware decoder. In this second case, it augments the JMF state transition events and does not replace them.

NOTE 1: Changes effected using javax.tv.media.MediaSelectControl do not lead to the org.davic.media.MediaPresentedEvent being thrown.

NOTE 2: The present document defines more specific conditions under which ResourceWithdrawnEvent and ResourceReturnedEvent than the original DAVIC specification. This can be found in clause 11.6.2, "Service selection API".

#### 11.4.2.5.3 Extensions in javax.tv

In javax.tv.media.MediaSelectControl the InsufficientResourcesException is thrown by the select() method if selecting any service component fails due to a lack of system resources.

In javax.tv.media.AWTVideoSize.equals(), the "data members" shall be considered to be the "source" and "dest" parameters used to create the object.

The components returned by javax.tv.media.MediaSelectControl.getCurrentSelection are the currently selected components which may be different from those previously selected by this control. Mechanisms which may modify the current selection include:

- Control of subtitles and audio language.
- The CA system including the common interface e.g Host Control replace / clear_replace.
- User intervention via the navigator.

The following clarifications shall apply to the method getDefaultSize in javax.tv.media.AWTVideoSizeControl:

a) This method shall return the current default AWTVideoSize. The default shall be that which would be implemented by the GEM terminal when the video scaling and positioning of a JMF player is under control of the GEM platform (see org.dvb.media.VideoFormatControl.DFC_PLATFORM). When the video scaling

and positioning is in the DFC_PLATFORM mode, the return value of this method shall change to track changes made by the policy underlying DFC_PLATFORM.

b) Calling AWTVideoSizeControl.setSize() with the AWTVideoSize object returned by this method as the parameter shall set the video scaling to the current default at the time getDefaultSize() was called. Hence it shall be equivalent to calling BackgroundVideoPresentationControl.setVideoTransformation with the video transformation returned by the method VideoFormatControl.getVideoTransformation(getDecoderFormatConversion()) and not with the video transformation returned by getVideoTransformation(DFC_PLAFORM).

NOTE: When the GEM terminal is in pan and scan mode, (see org.dvb.media.VideoFormatControl.DFC_PROCESSING_PAN_SCAN), the return value of AWTVideoSizeControl.getSize() will be out of date almost as soon as the method has returned.

#### 11.4.2.5.3.1 Enumerated Types

Java TV defines the following enumerated types:

- javax.tv.service.navigation.DeliverySystemType;
- javax.tv.service.guide.ProgramScheduleChangeType;
- javax.tv.service.ServiceInformationType;
- javax.tv.service.ServiceType;
- javax.tv.service.SIChangeType;
- javax.tv.service.SIRequestFailureType;
- javax.tv.service.navigation.StreamType.

Platforms including Java TV may define additional values for these types through subclassing. Applications written to Java TV or to any specification that includes Java TV should not rely on methods using values of these types being constrained to the set of values defined as constants in Java TV; values unknown to the application author may be returned.

Specification authors are advised not to define new values that semantically overlap with the values defined by Java TV or elsewhere.

### 11.4.2.5.4 Required controls for broadcast profiles and packaged media profiles

The following controls are supported for the broadcast streaming formats specified in clause 7.2, "Media streaming formats":

- org.davic.media.LanguageControl
- org.davic.media.AudioLanguageControl
- org.davic.media.SubtitlingLanguageControl
- org.davic.media.FreezeControl
- javax.tv.media.MediaSelectControl
- javax.tv.media.AWTVideoSizeControl
- org.dvb.media.VideoPresentationControl
- org.dvb.media.BackgroundVideoPresentationControl
- org.dvb.media.SubtitlingEventControl
- org.dvb.media.VideoFormatControl

- `org.dvb.media.DVBMediaSelectControl`

The following controls are supported for media decoded from clause 7.1.4, "Monomedia format for audio clips":

- `org.davic.media.MediaTimePositionControl`

Some of the classes referenced in clauses 11.4.2.5.4 and 11.4.2.5.5 are not required as specified by other clauses in the present document.

#### 11.4.2.5.5 Clarifications

In `SubtitlingLanguageControl`, subtitling being presented shall be defined as the subtitle decoder running. It does not require subtitles to be visible on screen at that time. See clause 13.5, "Subtitles".

When a `PresentationChangedEvent` is generated, the Player in question should re-evaluate the list of controls returned by the `getControls()` method for that player. Similarly, Players supporting the `MediaSelectControl` should re-evaluate the list of Controls returned by `getControls()` after calls to the `MediaSelectControl.select()` method is called.

Any controls referenced by an application after a `PresentationChangedEvent` or a `MediaSelectSucceededEvent` is generated may fail silently if they are no longer valid. Controls which are valid for the new content remain unaffected. Applications should re-acquire any controls they require after either of these events has been generated.

If the content being presented by a JMF player becomes unavailable due to tuning, `org.davic.media.ResourceWithdrawnEvent` shall be sent to listeners registered to receive that events. A `ResourceReturnedEvent` shall be sent to all listeners registered at that time if the content becomes available again.

The following controls shall work on a JMF Player in any state (including `Unrealized`, `Realizing`, `Realized`, `Prefetching`, `Prefetched` and `Started` Players) and changes made via these controls shall be preserved across JMF state transitions:

- `javax.tv.media.AWTVideoSizeControl`
- `org.dvb.media.VideoPresentationControl`
- `org.dvb.media.BackgroundVideoPresentationControl`

It is implementation-specific whether or not the effect of other JMF controls included in the present document survives JMF state transitions.

When decoding of the media stream is frozen by `org.davic.media.FreezeControl`, scaling the video (e.g. by `BackgroundVideoPresentationControl`) it is recommended that this have immediate effect on the existing frozen video still. GEM terminals that cannot do this shall temporarily resume decoding of the media stream to acquire a new video still that shall be scaled as specified, and shall automatically re-freeze the media after this video still has been acquired.

Where streaming service components of the same service (e.g. MPEG-2 elementary streams) are being decoded by different JMF players, there shall be no synchronisation between those components. Specifically, if the video and audio from a service are being presented by separate JMF players then they shall be presented in "free-running" mode ignoring the PCR.

#### 11.4.2.5.6 Component-based JMF players

JMF Players start as background players. Players which do not provide a `org.dvb.media.ComponentBasedPlayerControl` can only be used to present video in the background and their `getVisualComponent` method shall always return null. Players that provide an implementation of the `org.dvb.media.ComponentBasedPlayerControl` are capable of acting both as background and component-based players, and their `getVisualComponent` method must follow the semantics defined by that control and this clause.

If an application calls `Player.getVisualComponent()` and receives a non-null Component, that application then owns the visual component for that Player. It continues to own the visual component for that Player until the application is destroyed or calls `ComponentBasedPlayerControl.releaseVisualComponent()`. An application that owns

the visual component of a Player will get the same component returned by calls to `Player.getVisualComponent()` on that Player until it releases the component. Calls to `Player.getVisualComponent()` will return null if the visual component for that Player is owned by another application.

If `getVisualComponent` returns non-null, video shall continue running in the background until the first call to the paint method of the component returned by `getVisualComponent`. When this transition happens the video is resized and repositioned as required by the visual component. Possible areas outside of the component are updated to follow the rules in clause 13.3, "Graphics".

After this transition, applications shall re-acquire the list of JMF controls for the player. It is implementation dependent whether any previous JMF controls are re-used. Applications shall not use any previous JMF controls which are not in the new list of JMF controls. When a JMF player is performing as a component based player, the following JMF controls shall not be supported and video scaling and positioning shall be done by controlling the size and position of the component returned by the `getVisualComponent` method.

- `org.dvb.media.BackgroundVideoPresentationControl.`
- `javax.tv.media.AWTVideoSizeControl.`

Players which return a non-null `java.awt.Component` from the `getVisualComponent` method must fully support the semantics for `java.awt.Component` concerning positioning and scaling.

### 11.4.2.5.7 Streaming Monitoring API

The Streaming Monitoring API is required in the OTT target. The classes and interfaces are defined in annex “N.3 Streaming Monitoring API” of the present document.

The Streaming monitoring API is used to expose to applications information about streamed CoD played by JMF Player. This API is applicable with Adaptive Streaming as well as to non-adaptive streams and allows an application to be notified about changes in buffer status and quality of played stream.

The API is composed from the following classes and interfaces:

- `MonitoringControl`
- `MonitoringEvent`
- `StreamQualityChangeEvent`
- `StreamQualityInfo`
- `VideoStreamQualityInfo`
- `BufferingStatusEvent`
- `MonitoringListener`

The `MonitoringControl` is a new `Control` and its implementation shall be returned by JMF `Player` (`javax.media.Controller.getControl(),javax.media.Controller.getControls()`) used to play OTT media streams independently from whether the system supports adaptive streaming. By using this `Control` an application has access to information about played stream. Application is able to check audio and video format, current quality of a stream being presented to the user as well as all available qualities describes in a manifest file passed as a Locator to the Player, current download bitrate of a stream and line rate. It is also able to set a quality of the associated CoD.

Manifest file tells the receiver that several alternative representations are available on the server for the same content, e.g. more files corresponding to encodings at different bit rates of the same content which the receiver can seamlessly switch to based on varying network conditions.

`MonitoringEvent` is a base class for any event related to changes in the properties of playing media. At the current state `StreamQualityChangeEvent and BufferingStatusEvent` can occur.

BufferingStatusEvent shall be used when some changes in buffering status appear, e.g. buffer is underrun or network connection has been lost and buffer is not filling anymore. To prevent implementations from being overloaded buffering status events should not be sent faster than one per seconds.

`StreamQualityChangeEvent` shall be used when JMF Player is switching to another stream defined in media's manifest file due to changes in network conditions (Adaptive Streaming).

`StreamQualityInfo` is a class used to encapsulate necessary information about each stream described in media's manifest file.

`VideoStreamQualityInfo` is a subclass of `StreamQualityInfo` class. It is dedicated to encapsulate information specific for video streams.

To receive `MonitoringEvent` events, an object has to implement the `MonitoringListener` interface and use `MonitoringControl.addMonitoringListener()` method to register its interest in media property's events. All events are posted to each registered listener.

#### 11.4.2.5.8 Media Stream Synchronization API

The media stream synchronization API is an optional extension of the GEM core APIs, which enables to establish a synchronization link between multiple JMF players. It enhances the JMF player model as defined in JavaTV [16], which already defines a limited master/slave concept (See the section "managing controllers" in class javax.media.Player in JavaTV for more details). For this purpose a set of new controls and events are defined, which provides a flexible master/slave extension which is in-line with the JavaTV architecture.

The media stream synchronization API is defined in "N.3 Media Stream Synchronization API", an informative usage example is in Annex W.5.

The media stream synchronization API can be deployed on any existing and future GEM targets. It may be used to synchronize streams on terminals deployed only on a single network (e.g. dual tuner STBs) as well as on hybrid terminals, on which the GEM targets are combined into a hybrid profile.

The media stream synchronization API can be used to couple JMF players into a master and slave relationship and enables synchronization of the media presentation by synchronizing the clocks of these players. The media presentation can be controlled by controlling the master player, the slave player is synchronized accordingly. A master/slave relationship between JMF players lasts until the slave is explicitly removed from the master. Any JMF player becomes a master by adding a slave to it. Any JMF player becomes a slave, when it is added as a slave to another player.

Changes to the presentation state of the master player change the state of the slave player accordingly, i.e. starting a master player automatically starts the slave. Additionally all methods and controls, which change the media time or rate of the master player also have the same effect on the slave. Note that the slave can still be controlled directly, i.e. it can be moved to a different state (e.g. stopped) or set to a different media time. Any state change on a slave player doesn't affect the state of the master player. It is not permitted to set the media time and media rate of a slave player to a value different from the master player.

##### 11.4.2.5.8.1 API behavior in border cases

If an application calls a method with a null parameter instead of an object instance, the method throws a NullPointerException. Methods which return an array of elements return an empty array if no element is available.

##### 11.4.2.5.8.2 Establishing a Master/Slave Relationship

A master/slave relationship is a temporary synchronization relationship between JMF players. A player obtains the master role, i.e. it contains the reference clock to which other slave players can be synchronized. The master/slave relationship is established dynamically by adding a player as a slave to another player. It lasts until the slave is removed. The API permits to add multiple slaves to a master player. When a player has a slave role, it cannot act as a master for other players, i.e. there's only a one-level master/slave hierarchy.

An application, which wants to establish a master/slave relationship among players, can obtain an instance of `MasterSlaveSyncLinkageControl` of the master player. On platforms which don't allow synchronization, a call to `getControl("org.dvb.media.MasterSlaveSyncLinkageControl")` on any player will return null.

A master/slave relationship is established when another player is added to the master via the `addSlave` method of the `MasterSlaveSyncLinkageControl`, which becomes a slave player when the `addSlave` method was

successfully executed. A SyncControl can be obtained for any player, which can be used as a slave player but the methods of `SyncControl` are only effective, while a Master/Slave relationship is established.

To remove a slave from a master/slave relationship, the method `removeSlave` is used. If a slave had been removed, it can be controlled independently from the master. After all slaves are removed from a master, the master is again a normal player and could serve as a slave to another master.

Adding or removing a slave player can happen in any state of the master and slave player. These operations don't impact the player states of the master and the slave, i.e. the return value of `Player.getState()` does not change for the master and slaves.

Not all media streams and their corresponding JMF players can be successfully synchronized - If synchronization can never be established during the `addSlave` method, an `IncompatibleSynchronizableMediaTypeException` is thrown.

#### 11.4.2.5.8.3 Adding a Slave

When a slave has been added, the master `Player`:

- Invokes `setTimeBase` on the `slave` with the master `Player's TimeBase`. If this fails, `addSlave` throws an `IncompatibleTimeBaseException`.
- Synchronizes the slave with the `Player` using `setMediaTime`, `setStopTime`, and `setRate`.
- Takes the added slaves latency into account when computing the master `Player's` start latency. When `getStartLatency` is called, the master `Player` returns the greater of: its latency before the `Controller` was added and the latency of the added `Controller`.
- Takes the added slave's duration into account when computing the master `Player's` duration. When `getDuration` is called, the `Player` returns the greater of: its duration before the `slave` was added and the duration of the added `slave`. If either of these values is DURATION_UNKNOWN, `getDuration` returns DURATION_UNKNOWN. If either of these values is DURATION_UNBOUNDED `getDuration` returns DURATION_UNBOUNDED.
- Adds itself as a `ControllerListener` for the added slave so that it can manage the events that the slave generates. (See the Events section below for more information.)
- Invokes control methods on the added slave in response to methods invoked on the master `Player`. The methods that affect slave players are discussed below.

#### 11.4.2.6.8.4 Removing a Slave

When a `slave` is removed from a master `Player's` control, the master `Player`:

- Resets the slave's `TimeBase` to its default.
- Recalculates its duration and posts a `DurationUpdateEvent` if the master `Player's` duration is changed.
- Recalculates its start latency.

#### 11.4.2.5.8.5 Starting a Master Player

In accordance with the behavior for managed players as described in JavaTV, the start behavior is as follows:

When you call `start` on a master `Player`, all of the `slaves` managed by the `Player` are transitioned to the *Prefetched* state. When the `slaves` are *Prefetched*, the master `Player` calls `syncStart` with a time consistent with the latencies of each of the slaves.

**Calling realize, prefetch, stop, or deallocate on a Master Player**
When you call `realize`, `prefetch`, `stop`, or `deallocate` on a master `Player`, the `Player` calls that method on all of the `slaves` that it is managing. The `Player` moves from one state to the next when all of its `slaves` have reached that state. For example, a `Player` in the *Prefetching* state does not transition into the *Prefetched* state until all of its `slaves` are *Prefetched*. The `Player` posts `TransitionEvents` normally as it changes state.

**Calling syncStart or setStopTime on a Master Player**

When you call `syncStart` or `setStopTime` on a master `Player`, the `Player` calls that method on all of the `slaves` that it is managing. (The `Player` must be in the correct state or an error is thrown. For example, the `Player` must be *Prefetched* before you can call `syncStart`.)

**Setting the Time Base of a Master Player**
When `setTimeBase` is called on a master `Player`, the `Player` calls `setTimeBase` on all of the slaves it's managing. If `setTimeBase` fails on any of the `Controllers`, an `IncompatibleTimeBaseException` is thrown and the `TimeBase` last used is restored for all of the `Controllers`.

**Getting the Duration of a Master Player**
Calling `getDuration` on a master `Player` returns the maximum duration of all of the added `Controllers` and the master `Player`. If the `Player` or any slave has not resolved its duration, `getDuration` returns `Duration.DURATION_UNKNOWN`.

**Closing a Master Player**
When `close` is called on a master `Player` all slaves are closed as well.

#### 11.4.2.5.8.6 Setting the Media Time and Rate of a Master Player

In accordance with the behavior for managed players as described in JavaTV, the event handling is as follows:

When you call `setMediaTime` on a master `Player`, its actions differ depending on whether or not the `Player` is *Started*. If the master `Player` is not *Started*, it simply invokes `setMediaTime` on all of the slaves it's managing. If the `Player` is *Started*, it posts a `RestartingEvent` and performs the following tasks for each slave:

- Invokes `stop` on the slave.
- Invokes `setMediaTime` on the slave.
- Invokes `prefetch` on the slave.
- Waits for a `PrefetchCompleteEvent` from the slave.
- Invokes `syncStart` on the slave

The same is true when `setRate` is called on a master `Player`. The `Player` attempts to set the specified rate on all slaves, stopping and restarting the `Controllers` if necessary. If some of the `Controllers` do not support the requested rate, the `Player` returns the rate that was actually set. All `Controllers` are guaranteed to have been successfully set to the rate returned.

#### 11.4.2.5.8.7 Loss of Synchronization

The API permits to set a synchronization tolerance time value, within which the different Media streams may deviate from the master. If the master/slave player cannot keep the players synchronized within the tolerance value, this condition is reported with a `SyncStateChangedEvent`. After the synchronization is re-established between the master and the slave, this is also reported with a `SyncStateChangedEvent`.

A slave player always attempts to resynchronize to the master, however when it determines that synchronization will never be established, it set its status to unsynchronizable and reports this with an SyncStateChangedEvent with unsynchronzable.

#### 11.4.2.5.8.8 Event Handling

In accordance with the event behavior for managed players as described in JavaTV, the event handling is as follows:

Most events posted by a slave are filtered by the master `Player`. Certain events are sent directly from the slave through the master and to the listeners registered with the master.
To handle the events that a slave can generate, the master registers a listener with the slave when the master/slave relationship is established. Other listeners that are registered with the slave are still notified, while it is being managed by the master. When a slave is removed from the master `Player`'s list of slaves, the slave removes itself from the masters listener list.

**Transition Events**

A master `Player` posts `TransitionEvents` normally as it moves between states, but the slaves affect when the master changes state. In general, a master `Player` does not post a transition event until all of its slaves have posted the event.

**Status Change Events**

The master `Player` collects the `RateChangeEvents`, `StopTimeChangeEvents`, and `MediaTimeSetEvents` posted by its slaves and posts a single event for the group.

**DurationUpdateEvent**
A JMF `Player` posts a `DurationUpdateEvent` when it determines its duration or its duration changes. A master `Player's` duration might change if a slave updates or discovers its duration. In general, if a slave posts a `DurationUpdateEvent` and the new duration changes the master's duration, the master Player posts a `DurationUpdateEvent`

**CachingControlEvent**
A master `Player` reposts `CachingControlEvents` received from a `Players` that it manages, but otherwise ignores the events.

**ControllerErrorEvents**
A master `Player` immediately reposts any `ControllerErrorEvent` received from a `slave` that it is managing to its `ControllerListener`. After a `ControllerErrorEvent` has been received from a slave, a master `Player` no longer invokes any methods on the slave; the slave is ignored from that point on and the slave is removed from the master.

### 11.4.2.6 Restrictions on the Framework for Broadcast

Controls that are supported by GEM need not have an associated GUI component. Any calls to `Control.getControlComponent()` may return null.

Developers of GEM applications should not rely on the presence of the following classes or interfaces:

- `javax.media.CachingControl` (unless returned by a call to a JMF method).
- `javax.media.CachingControlEvent.`
- `javax.media.GainControl` (unless returned by a call to a JMF method).

A GEM implementation need not return a `CachingControl` or `GainControl` from a call to `Player.getControls()` however this is not prohibited. If a GEM implementation does return a `GainControl`, then the volume that is set using this control may not be greater than the system volume level. I.e. the gain control may only change the volume between mute and the volume level at the current system volume level. This system volume level shall be represented as 1.0.

If the visual component returned by `Player.getVisualComponent()` is hidden (as returned by `Component.isHidden()`), the video is hidden. It is legal to remove a visual component from an AWT hierarchy and add it back to an AWT hierarchy (perhaps in a different place). The video will be hidden when the component is removed and shown again when the component is painted again.

Applications should not expect `PushDataSource` and `PullDataSource` to exist for broadcast MPEG content, and it is not required that functional implementations be provided for implementations of JMF in DVB. The `DataSource` used may be implementation-specific and non-specified apart from its inheritance from `javax.media.protocol.DataSource`.

The constructor for `URLDataSource` may always throw `IOException` for broadcast MPEG content.

For the purposes of integration with `javax.tv.media.MediaSelectControl` "resynchronization" is not considered to occur provided that the same PCR_PID is referenced between the new and existing service components.

Implementations of JMF shall not tune but shall allow selection of media components from transport streams which can be reached without tuning even where they are not part of a currently selected service.

The failure modes if a locator is used which cannot be reached without tuning are defined as follows:

- For a JMF player which is a `ServiceMediaHandler`, the failure modes are fully specified in the description of `MediaSelectControl`.
- If a JMF player which is not a `ServiceMediaHandler` is created with such a locator then it shall not be able to enter the `Realized` state.

  The `Realizing` state shall be exited with the posting of a `ResourceUnavailableEvent`.
- For JMF players which are not `ServiceMediaHandlers`, `MediaSelectControl` shall be restricted to operating only on service components belonging to transport streams currently available without tuning.

NOTE: In this version of the present document, no media types are defined for which `javax.media.Player.addController` can have any other behaviour apart from throwing a `IncompatibleTimeBaseException`.

### 11.4.2.7 Intersection Between MediaSelectControl and SubtitlingLanguageControl/ AudioLanguageControl

The method `org.davic.media.LanguageControl.listAvailableLanguages()` shall list all available languages in the service concerned. This includes languages carried in service components which are not currently selected, either because the service component concerned has been de-selected using `MediaSelectControl` or because it was never selected initially because it was not carried in one of the components listed in the component tags of a locator which included component tags.

For the purposes of the present document, "component tags of a locator" is to be interpreted as meaning the description of the required components in a locator. This clause also places semantic requirements on subtitling-related APIs. As these APIs are optional in the present document, these requirements apply only if these APIs are required by the GEM terminal specification.

The method `org.davic.media.LanguageControl.selectLanguage()` shall over-ride any selection of service components made either by `javax.tv.media.MediaSelectControl` or by using a locator for a service which included component tags. If `selectLanguage()` has changed the set of service components selected, the methods on `javax.tv.media.MediaSelectControl` shall behave as if that change had been made through `MediaSelectControl` and return the correct set of service components taking into account the change made by `selectLanguage()`.

If the methods on `javax.tv.media.MediaSelectControl` are used to replace or remove the service component carrying the currently active audio or subtitles then the presentation of this service component shall stop. If the methods on `javax.media.MediaSelectControl` are used to add a service component carrying audio or subtitles when a service component of this media type is not already being presented then presentation of this service component shall be started. The language of the newly selected service component shall be returned by calls from the application to the method `getCurrentLanguage` on `AudioLanguageControl` or `SubtitlingLanguageControl` respectively. Attempting to select multiple service components of the same media type to be presented at the same time using methods on `MediaSelectControl` shall fail unless the GEM terminal concerned has sufficient resources to present them simultaneously. The failure mode shall be that as specified in `MediaSelectControl` for when insufficient resources are available.

Control of the availability of subtitles using `MediaSelectControl` or the methods `selectLanguage` and `selectDefaultLanguage` (which `SubtitlingLanguageControl` inherits from `LanguageControl`) shall not impact the value of "Application Control" as mentioned in figure 34, "Determining subtitling language and presentation setting" and vice versa.

### 11.4.2.8 Intersection between Streamed Media API and TV User Interface API

#### 11.4.2.8.1 Basic Principles

The receiver's output picture shape and/or WSS signalling is normally derived from the configuration of the `HVideoDevice` and the decoder format conversion being applied, as in "TV behaviour control". When an application

has the HGraphicsDevice reserved, the picture shape and/or WSS signalling shall be derived solely from the output aspect ratio of the HGraphicsDevice.

Some of the decoder format controls pre-defined as part of org.dvb.media.VideoFormatControl may require control over the pixel aspect ratio of the final video output signal after video, graphics and backgrounds have been combined as shown in figure 17. In order to enable this, a JMF player shall attempt to reserve the HVideoDevice instance on which its output is being displayed. Failure to reserve or appropriately configure the HVideoDevice shall not be considered fatal to the JMF player but may result in an inferior TV video presentation.

NOTE: The actual configurations of the HVideoDevice used and available are dependent on the precise configuration of the GEM terminal. Differences will exist between set-top boxes, integrated digital TVs and PCs. The precise nature of any "inferior TV video presentation" is dependent on the configuration of the GEM terminal, on the input video and the characteristics of the final output device. Typically, the result will be distorted, cropped or scaled video.

Component-based Players shall not have an associated HVideoDevice. Therefore the HVideoComponent.getVideoDevice() method shall always return null.

#### 11.4.2.8.2 TV Behaviour Control

The "TV Behaviour Control" in figure 31, "Format control in the presence of a JMF player", is the default video transformation behaviour for JMF players which have none other set. Setting the VideoTransformation to DFC_PLATFORM shall set the video format control to TV behaviour control mode.

NOTE 1: In this mode, any changes in the video input signal where "television behaviour control" requires changing the pixel aspect ratio of the final output signal (i.e. after the combination of video with graphics, subtitles and backgrounds) will also change the pixel aspect ratio of any graphics, subtitles and backgrounds combined into that one final output signal, assuming the JMF player has successfully reserved the HVideoDevice and can configure it appropriately.

NOTE 2: For an example of "TV behaviour control" in set-top boxes and integrated digital TVs, see E-Book.

NOTE 3: Where the JMF player is unable to change the configuration of the HVideoDevice, "TV behaviour control" takes into account the active HVideoConfiguration when determining the decoder format conversion to apply. This situation may occur if another application has reserved the HVideoDevice or if an application has set a conflicting configuration on another HScreenDevice of the HScreen.

#### 11.4.2.8.3 Application Behaviour Control

When a GEM application is controlling video presentation through BackgroundVideoPresentationControl (for video in the background) or VideoPresentationControl and methods on java.awt.Component (for video in a component), the application is responsible for monitoring the incoming video format and changing the video presentation if desired.

Instances of VideoTransformation constructed using the constructor of that class shall not impact the configuration of the HVideoDevice. Instances of VideoTransformation returned by VideoFormatControl.getVideoTransformation shall have the same impact on the HVideoDevice as the equivalent conversion applied in "TV behaviour control mode". These may not be fully achievable if the JMF player does not have the HVideoDevice reserved or if an application has set a conflicting configuration on another HScreenDevice of the HScreen.

#### 11.4.2.8.4 Dynamic Behaviour

A JMF player shall attempt to reserve the HVideoDevice instance on entry to the Prefetched state from any state except the Started state. A JMF player in the prefetched or started states which does not have the HVideoDevice reserved shall attempt to reserve it only when the application which created the JMF player calls the setVideoTransformation method of the control org.dvb.media.BackgroundVideoPresentationControl. A JMF player leaving the Started or Prefetched state for the Realized state shall release the HVideoDevice if it has it reserved.

A JMF player which has the HVideoDevice reserved and then loses the right to control that device shall post an instance of the event org.davic.media.ResourceWithdrawnEvent to all registered listeners for

ControllerEvents on that Player instance. If the right to control that device is subsequently recovered when a attempt to reserve the device (as defined above) succeeds, then an org.davic.media.ResourceReturnedEvent shall be posted.

#### 11.4.2.8.5 Resource Management Details

Inter-operable applications shall not call any of the methods on the instance of ResourceClient returned by HVideoDevice.getClient when a JMF player has reserved that HVideoDevice instance.

Inter-operable applications are allowed to call HVideoDevice.releaseDevice or HVideoDevice.setConfiguration directly. GEM terminals shall continue to present video on a best effort basis within the constraints imposed by the new video configuration.

### 11.4.2.9 Integration with providers

The following shall be supported to enable the streamed media API to be used in combination with the provider API (see clause 11.7.9, "Provider API"). Since service providers (clause 11.7.9) are optional in GEM, this clause is not required by GEM terminal specifications.

- An implementation specific DataSource (referred to as ProviderDataSource) that is aware of which SelectionProvider objects are in scope for a particular application. This DataSource shall check if the protocol of the MediaLocator is one for which a service provider in scope of that application has registered as a handler. The getContentType method of this DataSource is assumed to have a hard-coded return value of "multipart.dvb.service" (according to syntax specified by ContentDescriptor.mimeTypeToPackageName).

NOTE: The preceding requirement is that a class with that behaviour be present. It need not have the name ProviderDataSource in implementations.

- An implementation specific Player that can use instances of ProviderDataSource as input, i.e. which will not throw a IncompatibleSourceException when a ProviderDataSource is passed to its setSource method. For the purposes of the "algorithm for creating a Player from a MediaLocator", this is assumed to be listed in the content package-prefix-list as supporting the media-content-type-name "multipart.dvb.service", i.e. the content type supported by ProviderDataSource.

### 11.4.2.10 Additional and modified semantics for IPTV

For services and content items available through RTSP, calls to the method Player.setRate shall result in a request to change the scale value as defined in clause 12.34 of RFC 2326 [83]. The return value of the method shall be the actual scale value chosen by the server.

NOTE: There is no standardized mechanism for obtaining the set of available rates from a server. GEM applications wanting that information need to obtain it from the server themselves.

### 11.4.2.11 Time-setting operations for OTT

In an OTT target, when the methods involved in changing the current media time, i.e. javax.media.Player.setMediaTime() and org.davic.media.MediaTimePositionControl.setMediaTimePosition() are invoked, implementations will achieve the result expected by applications, transparently to the applications themselves. The way the result is actually achieved is implementation-dependant and may also depend upon the server (e.g. lack of HTTP Range header support) and/or the content (fixed vs adaptive rate).

After the player has successfully retrieved the part of the file containing the indicated media-time and the player has re-entered the started state a javax.media.MediaTimeSetEvent will be generated.

Implementations will adjust the time values passed to the above mentioned methods to the most appropriate time point available in the container.

# 11.5 Data access APIs

## 11.5.1 Broadcast Transport Protocol Access API

The broadcast transport protocol access API is defined in P "(normative): Broadcast Transport Protocol Access".

Relative file names used to access objects in the application filesystem shall be taken as being relative to the base directory indicated in the Application Description defined in clause 10.5, "DVB-J specific Application Description".

NOTE 1: Absolute and relative paths may be used in accordance with the APIs concerned, e.g. with `java.net.URL` (when used with a "`file://`" string prefix) and with `java.io.File`.

The caching rules specified in clause B.5, "Caching" shall be evaluated at the time when the information is loaded using `DSMCCObject.synchronousLoad()` or `DSMCCObject.asyncronousLoad()` or is loaded implicitly in response to other actions as defined in clause 11.5.1.1, "Constraints on the java.io.File methods for broadcast carousels". After the DVB-J object has been loaded, any possible changes in the object carousel to the content the object represents are not visible to the application when it accesses the content using this DVB-J object instance.

### 11.5.1.1 Constraints on the java.io.File methods for broadcast carousels

The application shall be able to use the standard `java.io.File` class for access to broadcast carousels (e.g. a carousel unaware application). In this case, the following definitions shall apply:

- The constructor of `File` only creates an instance of the abstract pathname and shall not cause synchronous access to the broadcast carousel that would block the thread.
- After the constructor of the DVB-J `File` object has been run, the directory entry information relating to this object may be in an unloaded state. The information relating to this object (e.g. its length) comes from the parent directory which is not required to have been loaded at this point by the constructor. However, this information may be available if the implementation has the information in cache.
- If the directory entry information for a DVB-J `File` object is in the unloaded state, then this information shall be synchronously loaded when any of the following methods are called on the object: `can_Read()`, `exists()`, `isDirectory()`, `isFile()`. If the loading fails, all these methods shall return false.
- If the content is in unloaded state, it shall be synchronously loaded when the `list()` method is called for a directory. If this implicit load should result in a service transfer, it shall not be done implicitly and the list() shall return an empty list.
- The method `lastModified()` returns a value as specified in the GEM terminal specification. GEM terminal specifications shall define signalling for this value to indicate at least 256 distinct values.

NOTE 1: Even if the "carousel" functional equivalent is adopted by a GEM terminal specification, the above applies. Such specifications have to define the binding for `lastModified()`, even if this binding is the same as MHP's.

NOTE 2: GEM terminal specifications may, of course, define a binding that provides for a larger set of values than the minimum required for the method `lastModified()`.

- Any version changes in a file after the constructor for an `FileInputStream`, `FileReader` or `RandomAccessFile` is called or an `InputStream` referring to a file is obtained through other means (e.g. via `javax.microedition.io.InputConnection`) will not be visible in the data read from that instance.
- The creation of a `FileInputStream`, `FileReader` or `RandomAccessFile` shall throw `FileNotFoundException` if the referenced object is not a file.
- Failure of a signed file to be authenticated shall be reported as defined in clause 12.6.1, "General principles".
- Failure of a signed directory to be authenticated shall be reported by the list methods returning null when called on a file object representing the directory whose authentication failed.

- The value returned by `java.io.File.length()` may not be accurate as it returns the value from the directory information rather than the actual size of the file.

There are no guarantees that the most recent version of a file will be returned unless the network signalling specifies that the file concerned requires transparent access.

NOTE 3: Although jar files may be accessed, using them extensively is not recommended within object carousels. Accessing files contained within a jar file that is contained within an object carousel is likely to be slower than accessing the same file when the jar file is unpacked into an object carousel.

#### 11.5.1.2 Methods dealing with write access

The `java.io.File` class also contains methods that assume write access to the file system. Due to its broadcast nature, the receiver naturally does not have write access to the carousel.

NOTE: A broadcast carousel is not a read-only file system (which has the property of not changing). The carousel content can certainly be written and modified, but only by the service provider - not the receiver. Therefore, the situation is equivalent to a Unix file system where the user has only read permissions, but not write permissions or ownership of the files.

The following `java.io.File` methods deal with write access to directories: `canWrite()`, `mkdir()`, `mkdirs()`, `renameTo()`.

For abstract pathname entries in the broadcast carousel, the following behaviour shall apply:

- `canWrite()` returns `false` to indicate that the file can not be written to.
- `mkdir()`, `mkdirs()` and `renameTo()` return `false` to indicate that the request failed.

#### 11.5.1.3 Behaviour following loss of a broadcast carousel

When a broadcast carousel is lost to an application as described in clause 6.2.5.3, "Loss of carousel behaviour", the following failure modes shall be followed for data which is unavailable.

- Attempting to load data from a file using a content format specific API shall fail as if the file itself never existed. This shall be done according to the specific API concerned.
- The classloader created by the platform when the application was first launched never attempts to automatically recover following loss of its initial broadcast file system.
- Attempting to load a class which is unavailable shall fail as if the class was never present.
- After the constructor for an `InputStream` has been successfully called, the data for that `InputStream` shall be maintained by the platform until the `InputStream` is closed. This is the same behaviour as specified for `InputStream` and file version changes in clause 11.5.1.1, "Constraints on the java.io.File methods for broadcast carousels".
- Attempting to create a `FileInputStream` for a file whose data is unavailable shall fail with a `FileNotFoundException`.
- Attempting to create a `RandomAccessFile` for a file whose data is unavailable shall fail with an `IOException`.
- Attempting to use a `FileDescriptor` of a `FileInputStream` whose data is unavailable shall result in a FileInputStream or InputStreamReader where all methods shall throw an `IOException`.
- Attempting to call methods on a `File` object where the filesystem information needed for the method call (see clause 11.5.1.1, "Constraints on the java.io.File methods for broadcast carousels") is unavailable shall fail in the same way as if the `File` itself never existed.
- Operations ongoing at the time of loss of broadcast carousel shall be terminated. Blocked Java I/O operations shall throw `InterruptedIOException`.

### 11.5.2 Support for Multicast IP over the Broadcast Channel

Where support for carriage of multicast IP over the broadcast channel (see clause 6.2.6) is included, the following shall apply:

a) In `java.net.MulticastSocket`, the `send` method shall throw an `IOException` when called on sockets bound to unidirectional sources of IP multicast data.

b) In `java.net.DatagramSocket`, the `send` method shall throw an `IOException` when called on sockets bound to unidirectional sources of IP multicast data.

c) The class `org.dvb.net.DatagramSocketBufferControl` shall be supported see annex Q, "Datagram socket buffer control".

d) Behaviour if unsupported:

   When the application tries to `joinGroup()` on a Multicast address, ProtocolException shall be thrown to indicate failure on joining the group.

### 11.5.3 Support for IP over the Return Channel

NOTE 1: Where support for IP over the return channel is not included, APIs in the `java.net` package will fail as defined in the specification of this API. Where support for IP over the return channel is supported, platforms not implementing specific optional return channel protocols will fail as defined in the specification of this API.

On devices whose return channel can be connected or disconnected, connecting a `java.net.Socket` or a `java.net.URLConnection` to a host addressed via the return channel shall automatically setup a connection to the default connection target subject to the application having return channel permission for the default ISP and the return channel either being not connected or being connected to a different target but with the return channel resource reserved by another application with a lower priority. Such connections shall be automatically disconnected after a period of inactivity on that connection which it should be possible to define using the navigator. This period shall be returned to applications in the `"dvb.returnchannel.timeout"` system property encoded in seconds as a decimal number.

See clause 11.10, "Java permissions".

The class `org.dvb.net.DatagramSocketBufferControl` shall be supported see annex Q, "Datagram socket buffer control".

NOTE 2: Support multicast of IP over the return channel is not required by clause 6.3, "Interaction channel protocols". On GEM terminals not supporting this, the methods concerned will fail using one of their defined failure modes.

### 11.5.4 MPEG-2 Section Filter API

The MPEG-2 section filter API is defined in annex E of DAVIC 1.4.1p9 [17].

### 11.5.5 Mid-Level Communications API

See annex R, "DVB-J return channel connection management API".

On devices whose return channel can be connected or disconnected, when use of `java.net.Socket` or a `java.net.URLConnection` automatically sets up a connection, this shall be reported through this API to any applications listening for it.

Implementations shall automatically drop connections established using this API after the same period of inactivity as defined for automatically setup connections in clause 11.5.3, "Support for IP over the Return Channel".

If an application which has reserved a `ConnectionRCInterface` and established a connection then releases the resource without disconnecting, the behaviour of the GEM terminal is implementation dependent.

If there are open sockets using that connection, the GEM terminal should keep the connection established until the time-out period defined in clause 11.5.3, "Support for IP over the Return Channel" for automatically setup connections.

NOTE: On GEM terminals not including a return channel, this API is present but will report that no `RCInterface` instances are visible to any application.

## 11.5.6 Persistent Storage API

- The API to persistent storage shall be the `javax.microedition.io` package, the `java.io` package and the extensions to it found in the `org.dvb.io.persistent` defined in annex K, "DVB-J persistent storage API".
- The `"dvb.persistent.root"` property which can be obtained from `java.lang.System.getProperty()` identifies the directory at the root of the file name space used for persistent storage. This value shall be the same for all applications.
- Accessing any files or directories in the parent directory of this root or directories above that shall throw a SecurityException as defined in the specification for the `java.io` package.
- Signed applications which are authenticated and which are granted the right to access persistent storage have the following privileges:
  - Read only access to the root directory (defined above).
  - Automatic read and write access to an "organization" sub-directory named `<organisation_id>`.
  - Automatic read and write access to an "application" sub-directory named `<organisation_id> + <file.separator> + <application_id>`.
- When the permission request file for an application requests access to persistent storage and this is granted, the GEM terminal shall be responsible for creating the necessary directories in which the application is allowed to read/write where these do not already exist. This shall be done shall be done no later than immediately prior to the first access (including existence checks such as `File.isDirectory()`) by that application instance to the directories concerned.

  For applications which are granted file access (see clause 12.6.2.7, "File Access"), the necessary directories are only its organization and application sub-directories. For applications which are granted a credential, (see clause 12.6.2.6, "Credentials"), the necessary directories are those defined by any filename elements in that credential up to but not including the first directory containing a wildcard.

EXAMPLE: A filename element such as:

`org0_id/appA_id/external/-`

creates

`org0_id/appA_id/external`

When the GEM terminal automatically creates an "application" sub-directory as part of the necessary directories defined above, it shall set the owner of the created sub-directory to the application whose name corresponds to that sub-directory and set read and write access to the application whose name corresponds to that sub-directory and no access for other applications. The GEM terminal shall set the owner of any additional automatically created necessary directories that reside below an "application" sub-directory to the application whose name corresponds to that "application" sub-directory and set read and write access to the application whose name corresponds to that "application" sub-directory and no access for other applications.

If at the time when a GEM terminal would create the "application" sub-directory (if it did not exist), that directory already exists but is owned by an application other than the one whose `application_id` is used as its name, the GEM terminal shall remove this directory and any contents of it and create an empty "application" sub-directory with the owner and access rights set as defined above.

- The owner of the "organization" directory shall be the platform and the directory itself shall always have organization read / write and world read access.

- All files in persistent storage shall store the 48 bit application identifier of their creator as the "owner" of the file. Only the owner of the file is entitled to change the file attributes and file access rights. The owner of a file cannot be changed once a file is created.
- Applications may only create files or directories in directories to which they have write access.
- The existing semantics for the java.io package are respected.
- See clause 14.5, "Text encoding of application identifiers". The fields `<organisation_id>` and `<application_id>` are hexadecimal text encodings (without leading zeros) of the fields in the 48 bit application identifier of the application concerned as defined in clause 10.4.3, "Content of the Application Description".
- All access to persistent storage shall be consistent with the security mechanism defined in clause 12, "Security".

NOTE 1: As defined in clause 11.5.1, "Broadcast Transport Protocol Access API", relative paths cannot be used to access persistent storage in an inter-operable way.

NOTE 2: Unsigned applications have no access to persistent storage. See clause 12.6.2.7.1, "Unsigned applications".

The contents of a file placed in persistent storage shall persist at least until one of the following conditions occur:

a) There is not enough free persistent storage available to satisfy the persistent storage needs of a running application.

b) The file stored in persistent storage expires.

c) The file is cleared (i.e. deleted) by the creating application or an application with appropriate permissions.

d) The file is cleared as a result of an explicit request by the end user.

e) The persistent storage used by GEM applications exceeds 75 % of the value in table 88, "Minimum requirements for other resources for conformance purposes".

NOTE 3: The behaviour of the GEM terminal when any of the above conditions occur is implementation specific except as follows:

Each organisation that has a GEM `organisation_id` may have one special file in persistent storage. This file is identified by having the name "prefs.bin" and being located in the sub-directory immediately below the root of the persistent file namespace whose name is the `organisation_id` of the organisation concerned. As long as the special file has a size less than or equal to 2 K bytes, the following behaviour applies:

a) GEM terminals do not automatically delete these special files except when persistent storage is full and all files in persistent storage are these special files.

b) GEM terminals that offer the end-user a user interface to manage the contents of persistent storage should present these special files such that they are less likely to be deleted by the end-user than any files that are not special.

The terminal shall give priority to the retention of higher priority files over lower priority files within the scope of a specific(organization ID, application ID) pair. See `org.dvb.io.persistent.FileAttributes` for priorities.

The details regarding the release of cleared (i.e.deleted) or expired files are implementation dependent.

The GEM terminal shall have a means to clear at least the quantity of persistent storage defined in table 88, "Minimum requirements for other resources for conformance purposes". This mechanism shall be usable in conformance tests, and is not subject to the rules above.

Some GEM terminals may allow a file to be opened for writing by only one `FileOutputStream` (or other file-writing object) at a time. In such situations the constructors in `FileOutputStream`, `RandomAccessFile` and `FileWriter` will fail if the file involved is already open. Concurrently running applications writing to the same file(s) in persistent storage cannot rely on this mechanism and will need to co-ordinate access to persistent storage themselves.

### 11.5.7 File Storage Device Access

#### 11.5.7.1 Basic Specification

The API for access to file storage devices is the `org.ocap.storage` package defined in clause 13.3.7.5 and annex V of OCAP [5].

#### 11.5.7.2 DVB specific modifications

The following modifications apply to the `org.ocap.storage` package in the context of the present document:

- The requirement that "Persistent storage device contained within or attached to a Host device shall be represented by a StorageProxy" does not apply in the present document. Only detachable and removable devices need be represented in this way.
- The "storage" target for the class `MonitorAppPermission` is not defined in the context of the present document. Exceptions stated as being thrown for applications that do not have `MonitorAppPermission("storage")` shall always be thrown. Hence, the methods `StorageProxy.deleteVolume` and `StorageProxy.initalize` are not required to be implemented in GEM terminals. It is expected that the initialize feature would be provided by the resident navigator in GEM terminals supporting access to memory cards.
- GEM persistent file systems are not required to support the extended attributes defined in the class `org.ocap.storage.ExtendedFileAccessPermissions`. The method `org.dvb.io.persistent.FileAttributes.setFileAttributes` shall throw an IOException when called for a file whose file system does not support these extended attributes.

## 11.6 Service information and selection APIs

### 11.6.1 Signalling-specific service information API

There is no signalling specific SI API defined in GEM. GEM terminal specifications shall provide any needed service information API specific to their signalling.

### 11.6.2 Service selection API

The service selection API is defined by the `javax.tv.service.selection` package from Java TV [16] and the `org.dvb.service.selection` package defined in annex AK.

All instances of `ServiceContext` returned by the GEM terminal shall be instances of `DvbServiceContext`.

On the first occasion when a the JMF player is obtained for a specific service (either by ServiceContext.getServiceContentHandlers or HVideoDevice.getVideoController), any JMF players returned shall always be in the started state If they are presenting video, that video shall always be presenting on the background video device unless all of the following apply:

- the previous service presented in that service context contained video;
- that video was presented in a Component (i.e. the ServiceMediaHandler presenting it was a component based player);
- the component continues to be displayed after the completion of the selection of the current service in the service context.

If all of these conditions are met then the video of the new service shall be displayed in the same Component as the video of the previous service and the JMF player used shall be a component based player.

Applications shall re-acquire the list of service content handlers after a service selection completes. It is implementation dependent whether any previous handlers are re-used. JMF players which are not re-used are stopped and may be disposed by the platform. Where a JMF player is re-used, it shall be reset to a default condition as if it was not being reused. In particular the implementation shall cancel all listeners previously defined and reset previously defined characteristics such as audio language. If the video is presented in a background video device and no default video transformation is set for the service context then video scaling and positioning shall be reset.

The exception `javax.tv.services.selection.ServiceContextException` shall never be thrown by GEM terminals.

The default media player behaviour following service selection is defined under clause 11.4.2.3, "Default media player behaviour".

The protected constructor of `ServiceContextFactory` is for implementation use. GEM applications shall not subclass `ServiceContextFactory`. Implementations are not required to behave correctly if they should do this.

Implementations of this API are required to perform tuning as part of implementing the `ServiceContext.select` methods where the new service to be selected is part of a transport stream known to the GEM terminal but not currently tuned to.

All the GEM applications running within the same service context shall have the ability to obtain and modify the state of the service component handlers for all service components being presented in that `ServiceContext`. For service component handlers which are JMF players, modifying the state includes changes to the settings of JMF controls in addition to the state machine of the JMF player itself.

Applications not running in that service context (e.g. the GEM navigator) shall not be able to modify the state of service component handlers unless they first inform the applications in the service context of this. For service component handlers which are JMF players, this informing shall be done by sending a `org.davic.media.ResourceWithdrawnEvent` to all applications which have currently registered listeners for ControllerEvents on the JMF player whose state will be modified.

Once a `ResourceWithdrawnEvent` has been sent, any changes made by applications inside the service context to the state of service component handlers will be ignored by the GEM terminal. Methods to query state shall return the actual state and methods to change the state shall silently fail. It is implementation dependent whether events related to JMF continue to be sent to applications.

If the application outside the service context ceases to need to make changes to the state of service component handlers, the applications inside the service context shall return to having that right exclusively. The applications inside the service context shall be informed of the return of this right. For service component handlers which are JMF players, this informing shall be done by sending a `org.davic.media.ResourceReturnedEvent` to all applications which have currently registered listeners for `ControllerEvents` on the JMF player which is returned to exclusive control.

Following receipt of a `ResourceReturnedEvent`, the GEM applications running in the service context are responsible for returning the configuration of all service component handlers to what they require. The GEM terminal is not required to automatically restore any configuration of any service component handler to any former value. In the case of a service component handler which is a JMF Player, the configuration is the set of writeable properties accessed through the `Controls` of that `Player`.

NOTE 1: Example properties include video scaling, video positioning and audio language choice.

NOTE 2: One situation where this may occur is where the end-user of the GEM terminal brings up the UI of the navigator, does some operation and then exits the navigator, returning control back to the GEM applications.

NOTE 3: There is no relationship required between the generation of these events and the Xlet state model. GEM terminals may chose to put all GEM applications which are in the Active state into the Paused state when the bring up the UI of the navigator however there is no requirement for this.

A running DVB-J application shall be considered as a component of the service being presented in the `ServiceContext` in which the application is running. It shall have a `ServiceContentHandler` which is intentionally not defined by the present document except that it shall not be a `ServiceMediaHandler`. The results of calling the `getServiceContentLocators` method on this `ServiceContentHandler` are implementation dependent, and there is no requirement for these Locators to be accepted as input by any method in the present document. If the `ServiceContext` in which a DVB-J application is running is stopped, the DVB-J application shall be stopped like all other service components being presented in that service context. `ServiceContentHandler`s for DVB-J applications shall not be shared between DVB-J applications if more than one application is running.

NOTE 4: An Xlet is not a `javax.tv.service.navigation.ServiceComponent`, the elementary stream(s) carrying the object carousel carrying the Xlet are represented by at least one of these.

NOTE 5: Services with only DVB-J applications and no media components will have no `ServiceMediaHandlers` but only `ServiceContentHandlers` representing the running DVB-J applications.

NOTE 6: The present document intentionally does not define any circumstances under which `SelectionFailedEvent(MISSING_HANDLER)` is required to be generated.

NOTE 7: Stopping all the components of a service (both media and Xlets) results in a service with no components being presented, the same as if a service with no components had been initially selected.

The following stream types (from `javax.tv.service.navigation.StreamType`) shall be selectable: AUDIO, VIDEO, SUBTITLES. Service components of these types shall have a `ServiceContentHandler` which may be shared between multiple service components of the same or different stream types. It is implementation-dependent whether other stream types are selectable, but failure when selecting a stream type that is not selectable is handled as defined in the present document. The effect of successfully selecting a stream type that is not in the preceding list is implementation-dependent.

If the selected service is a DVB NVOD reference service, GEM terminals shall make an implementation dependent choice from those NVOD time shifted service associated with that reference service and try to select the chosen time shifted service. The algorithm for the implementation dependent choice shall be automatic and not involve the end-user. Any failure shall be reported as specified through this API.

NOTE 8: Applications which wish to chose a specific time shifted service need to explicitly specify that service and not rely on the choice of the receiver.

This API shall support the following features on those GEM terminals which support the feature concerned.

- Internet clients (see clauses 9.7, "Lifecycle of internet access applications" and 11.14.1, "Internet client control APIs").
- Stored services (see clauses 9.6.2, "Stored services" and 11.12.2.2, "Modified behaviour of GEM 1.0 APIs").
- Services signalled over the interaction channel (see clauses 9.6.1, "Applications loaded from the interaction channel" and 11.11.12, "Support for the HTTP Protocol in DVB-J").

This API shall support the following features:

- Services whose locator is an instance of the `org.dvb.locator.FrequencyLocator` class.
- Service instances representing PSI-only services.

Service contexts shall not have any hard coded restrictions on the types of service that can be selected in them. For example, a service context which is at one point presenting a broadcast service may not fail to have a stored service selected in it for no other reason than the service to be selected being a stored service.

In the method `ServiceContext.select(Locator[])`:

- The language applying to Xlets shall apply to all GEM applications regardless of type.
- Xlets can be listed using the application locator, see MHP [1], clause 14.1.7, "Application locator".
- Any listed locators for applications not signalled as AUTOSTART or PRESENT shall be ignored.

Subclasses of `javax.tv.service.selection.PresentationChangedEvent` shall be associated in time with the generation of a `MediaPresentedEvent` and not the JMF state change event which may have happened some seconds previously.

Re-selecting the currently selected service in a service context shall be as defined in the `javax.tv.service.selection` package with the additional clarification that running applications whose `service_bound_flag` is set to 1 shall not be killed and re-started if the Xlet is supposed to be running after the selection operation completes.

Completion of a service selection operation (whether reported via `NormalContentEvent`, `AlternativeContentEvent` or `SelectionFailedEvent`) shall be reported once all `ServiceMediaHandler` instances are in the started state, without waiting for any xlets to enter the active state.

## 11.6.3 Tuning API

GEM terminal specifications for packaged media targets are not required to support a tuner. The definitions and requirements of this clause apply only to GEM terminal specifications that support a tuner.

### 11.6.3.1 Generic description

The tuning API is defined in annex H of DAVIC 1.4.1p9 [17] apart from section H.4, (the `Locator` and `DvbLocator` classes) which are found in clause 11.7.6, "Content referencing". The reference to the `DvbLocator` class does not apply to GEM terminal specifications that do not include the functional equivalent named "Content Referencing" as defined in clause 15.6, "Functional equivalents".

The following methods in this package shall throw `java.lang.SecurityException` where and only where the application does not have a `org.dvb.net.tuning.TunerPermission`:

- `NetworkInterfaceController.reserve`
- `NetworkInterfaceController.reserveFor`

The class `org.davic.net.tuning.dvb.DvbNetworkInterfaceSIUtil` is not required.

See also clause 11.8.3, "Additional permissions classes".

Tuning automatically performed by other APIs in GEM shall be reported through the tuning API to any applications listening for it.

### 11.6.3.2 Tuning in IPTV

In IPTV, the tuning API shall represent control over the input to the MPEG-2 demultiplexer. Hence tuning to a transport stream means connecting that transport stream to this input.

### 11.6.3.3 Tuning in OTT

In OTT, the tuning API shall represent control over the input to the stream decoder / demultiplexer. Hence tuning to a transport stream means connecting that transport stream to this input.

## 11.6.4 Conditional access API

The present document does not require a conditional access subsystem, nor does it place requirements on one if it is present.

GEM terminal specifications may define any needed conditional access API.

## 11.6.5 Protocol independent SI API

### 11.6.5.1 Generic description

The protocol independent SI API is defined by the following packages from Java TV [16]:

- `javax.tv.service`
- `javax.tv.service.guide`
- `javax.tv.service.navigation`
- `javax.tv.service.transport`

The mapping of this onto the DVB-SI protocol is specified in annex O, "Integration of the JavaTV SI API". The mapping of this onto services whose service information is obtained through a provider is defined in clause 11.7.9, "Provider API".

The mapping of the protocol independent SI API onto the underlying SI protocol is not defined in GEM. Thus, the reference to annex O does not apply for GEM terminal specifications that do not include the functional equivalent named "SI" as defined in clause 15.6, "Functional equivalents". However, GEM terminal specifications shall provide a mapping of the protocol independent SI API onto their SI signalling, as specified in annex O.

Cancellation shall fail if the request is no longer pending. It is implementation dependent if cancellation fails under any other circumstances. Implementations are not required to support cancellation of requests between when the requested SI data has arrived in the device and when execution of the `notifySuccess` method starts.

The interface `javax.tv.service.ServiceMinorNumber` is not required to be implemented by any object defined in GEM.

This API shall support the following features on those GEM terminals which support the feature concerned.

- Internet clients (see clauses 9.7, "Lifecycle of internet access applications" and 11.14.1, "Internet client control APIs").
- Stored services (see clauses 9.6.2, "Stored services" and 11.12.2.2, "Modified behaviour of GEM 1.0 APIs").

The method javax.tv.service.SIManager.getService(Locator) shall, if passed an instance of org.dvb.locator.FrequencyLocator or a "dvb:" locator referencing a PSI-only service, return an instance of javax.tv.service.Service representing the service concerned. The service returned shall have the following characteristics:

- the `getName` method shall return a string of length zero;
- the `getServiceType` method shall return UNKNOWN;
- the `retrieveDetails` method shall report `notifyFailure` with `SIRequestFailureType(DATA_UNAVAILABLE)`.

Such services shall never be returned to a GEM application instance from a call to `SIManager.filterServices`.

A service of these characteristics shall be returned even if the service referenced by the `FrequencyLocator` is one in the service list of the receiver for which the name, type and details are known.

For the purposes of `SIManager.getService(Locator)` throwing an `InvalidLocatorException`, the `FrequencyLocator` shall be considered valid if and only if the GEM terminal has a `NetworkInterface` of the same type (cable/satellite/terrestrial) as the descriptor used to construct the Locator.

### 11.6.5.2 Transport independent and dependent services

The following clarifications shall apply to Java TV as used in GEM.

- All instances of Service that are not instances of a standardised sub-type of Service (e.g. AbstractService, StoredApplicationService or InternetClientService) shall be instances of either TransportIndependentService or TransportDependentService.
- Instances of TransportDependentService shall not appear in the list of service returned by SIManager.filterServices, even when null is passed to that method.

NOTE: As a consequence, DeliverySystemTypeFilter cannot return instances of TransportDependentService.

- SIManager.getService shall return an instance of TransportDependentService when passed a transport dependent locator.
- ServiceDetails.getService shall not return a TransportDependentService
- References to "broadcast service information" shall be interpreted to include service information in the IP network as defined in clause 6.5.2, "Service information and metadata protocols" of the present document.

#### 11.6.5.3 Modified Semantics of Existing APIs

Clause T.2.2.16 of OCAP [5] shall be supported for instances of javax.tv.service.navigation.ServiceDetails corresponding to abstract services.

#### 11.6.5.4 Extensions

This is the org.dvb.service and org.dvb.service.navigation packages.

### 11.6.6 Service discovery and selection for IPTV

This is the org.dvb.service.sds package.

The org.dvb.service.DvbSIManager class shall implement org.dvb.service.sds.SDSRecordAccess.

The `org.dvb.service.sds` package is defined in annex AW.

### 11.6.7 Integration between protocol independent SI API and TV-Anytime

This is the org.dvb.tvanytime.javatv package.

The org.dvb.service.DvbSIManager class shall implement the org.dvb.tvanytime.javatv.CRIDAccess interface.

Instances of javax.tv.service.guide.ProgramEvent whose data was obtained via the TV-Anytime protocols used in TS 102 539 [81] shall implement org.dvb.tvanytime.javatv.TVAProgramEvent.

The `org.dvb.tvanytime.javatv` package is defined in annex AY.

## 11.7 Common infrastructure APIs

### 11.7.1 APIs to support DVB-J application lifecycle

This API is formed of the Java classes and interfaces found in the `javax.tv.xlet package` specified in Java TV [16]. In addition, the package `javax.microedition.xlet` contains similar classes. In either case, the `XletContext` passed to the `Xlet` shall implement both `javax.microedition.xlet.XletContext` and `javax.tv.xlet.XletContext`.

GEM terminals shall determine which Xlet interface is implemented by the initial class of an application, and use the corresponding application management API. In the case where the initial class of an GEM application implements both `javax.tv.xlet.Xlet` and `javax.microedition.xlet.Xlet`, the GEM terminal shall use the `javax.tv.xlet` application management API.

#### 11.7.1.1 Xlet properties

The following named Xlet properties shall be supported:

- `dvb.org.id.`
- `dvb.app.id.`
- `dvb.caller.parameters.`

They can be retrieved from an Xlet's XletContext by calling `getXletProperty` with the string name defined above.

All keys for `XletContext.getXletProperty` beginning `'dvb.'` are reserved for use in DVB project specifications.

The array of strings returned by `XletContext.getXletProperty(XletContext.ARGS)` shall be the array of strings defined in the DVB-J application descriptor (see clause 10.5.2) in the same order as specified in the signalling. Each entry in the loop of that descriptor shall be presented as one string in the array returned by this method. Zero-length strings in the signalling shall be represented as the empty string.

The "dvb.caller.parameters" XletProperty contains the array of Strings that can be passed to applications started via AppProxy. If this application was started by the system or by another application without passing parameters via DVBJProxy (using either the start or init methods), an empty array is returned as the value of this XletProperty.

The "dvb.org.id" and "dvb.app.id" Xlet properties shall be initialised from the `application_identifier` of the application.

**Table 31: Property name / type / encoding mapping**

| Property Name | getXletProperty return type |
|---|---|
| dvb.app.id | String encoded as in clause 14.5, "Text encoding of application identifiers". |
| dvb.org.id | String encoded as in clause 14.5, "Text encoding of application identifiers". |
| dvb.caller.parameters | String[], no information to be indicated with an array of length zero. |
| dvb.installer.parameters | getXletProperty return type String[] (Not required in GEM). |

#### 11.7.1.2 Actions for DVB-J applications to perform in their destroy method

Xlets shall perform at least the following in their `Xlet.destroyXlet` method and before calling `XletContext.notifyDestroyed`:

- Cause any threads that they have created to exit voluntarily.
- Stop, deallocate and close any JMF players that they have created.
- Stop and destroy any Java TV service selection ServiceContext objects that they created.
- Release any other scarce resources that they created, e.g. `NetworkInterfaceControllers` if they do any tuning.
- Flush any images using the `Image.flush()` method.
- Xlets shall not cause any unnecessary delay in their `Xlet.destroyXlet` method.
- De-register any event listeners.

If applications do not release these resources then the platform may do it for them.

Regardless of whether an application releases resources or whether the platform does it, the appropriate notifications shall be sent to all other GEM applications which are registered to receive these. For resources which are managed by an API using the `org.davic.resources` package, these notifications are specified by that API. For resources which are related to JMF players which are `ServiceMediaHanders`, if the GEM terminal implementation changes the configuration of these resources (e.g. to tidy-up), this shall be notified as defined in clause 11.6.2, "Service selection API" for "Applications not running in that service context".

NOTE: Applications have to allow for tuning happening in parallel with the execution of their `destroyXlet` method. Hence implementations of that method should not rely on being able to load classes while executing as the carousel which the application was using may not be available.

## 11.7.2 Application discovery and launching APIs

This API is formed of the `org.dvb.application` package defined in annex S, "Application listing and launching".

In GEM terminal specifications that include the functional equivalent named "Carousel"defined in clause 15.6, "Functional equivalents", full support of all properties defined for use with the method `AppAttributes.getProperty` in table 32 is required. In GEM terminal specifications that do not include this functional equivalent, support of these properties is optional.

The following properties are defined for use with the method `AppAttributes.getProperty`.

**Table 32: Application attribute properties**

| Property name (see note) | Return |
|---|---|
| dvb.j.location.base | Returns String containing `base_directory_bytes` from DVB-J application location descriptor. |
| dvb.j.location.cpath.extension | Returns String[] derived from `classpath_extension_bytes` of DVB-J application location descriptor with each array entry corresponding to a pathname entry as defined for `classpath_extension_bytes`. |
| dvb.transport.oc.component.tag | Returns Integer containing the `component_tag` from the selector bytes of the transport protocol descriptor.<br>If more than one transport protocol descriptor is in the AIT for a remote application then it is implementation dependent which of them is returned. |
| dvb.transport.oc.component.tag | Returns Integer containing the `component_tag` from the selector bytes of a transport protocol descriptor with `protocol_id` 0x0001.<br>If more than one transport protocol descriptor with `protocol_id` 0x0001 is in the AIT for a remote application, it is implementation dependent which of them is returned. |
| dvb.ait.descriptors | See clause 11.7.8, "Plug-in APIs". |
| dvb.transport.oc.locator | An `org.davic.net.Locator` that identifies the object carousel.<br>This shall be a Locator that is suitable for the `ServiceDomain.attach(Locator)` method. This shall be the Locator for the component specified by the "`dvb.transport.oc.component.tag`" property, in the service identified by `getServiceLocator()`. |
| NOTE: Property names beginning "`dvb.`" are reserved for future use. | |

Where a property is specified as either containing a value from a particular descriptor or being derived from a particular descriptor, the property shall not be returned for applications for which that descriptor is not signalled. Where a property is specified as containing a value from a transport protocol descriptor with a particular `protocol_id`, the property shall not be returned for applications for which a transport protocol descriptor with that `protocol_id` is not signalled.

NOTE: The three properties `dvb.j.location.base`, `dvb.j.location.cpath.extension` and `dvb.transport.oc.component.tag`, defined above, are specific to transport via an MHP object carousel. They may not be available if other transport is used in other specifications, e.g. in a GEM terminal specification or a future version of the present document.

Table 33 defines the source of the information which shall be used for methods returning information from entries in the application database for an application signalled in an Application Description.

**Table 33: Information source for methods on AppAttributes**

| Method | Information source |
|---|---|
| getName() | One of the names that can be found in the application_name of the Application description. |
| getName(String ISO639code) | A name of the application_name of the Application description corresponding to the specified language, if available. |
| getNames() | All of the names for the application which can be found in the application_name of the Application description and their ISO 639 [84] language code. |
| getProfiles() | The set of profiles indicated in the application_profile of the Application description. |
| getPriority() | The value indicated for the application_priority of the Application description. |
| getVersions(String profile) | The values version.major, version.minor and version.micro for the specified profile from the Application description. |
| getIsServiceBound() | True if the `service_bound_flag` of the Application description indicates true. Otherwise false. |
| isStartable() | There is no information source for this method, the return value is derived as specified in the method description. For the purpose of the method description, remote applications are as specified in the GEM terminal specification, if they are supported. |
| getIdentifier() | The `organisation_id` and application_id of the Application description. |
| getServiceLocator() | If remote applications are supported, the locator for a remote application shall encapsulate the values found in the appropriate signalling in the terminal specification. |
| getLocator() | The `application_icon_locator` of the Application description. |
| getIconFlags() | The `application_icon_flags` of the Application description. |

## 11.7.3 Inter-application communication API

This API is formed of the packages `java.rmi` and `javax.microedition.xlet.ixc` as specified in PBP 1.1 [4] plus the package `org.dvb.io.ixc`.

Two named xlet properties are introduced: dvb.org.id and dvb.app.id. They can be retrieved from an xlet's XletContext by calling `getXletProperty("dvb.org.id")` and `getXletProperty("dvb.app.id")`, respectively.

The package `org.dvb.io.ixc` provides an access mechanism for the registry for registering and looking up remote objects for inter-xlet communication. This mechanism implicitly provides the organisation ID and application ID of an object when exported, but requires the caller to provide these values when importing an object. The `org.dvb.io.ixc` registry shall be mapped to the `javax.microedition.xlet.ixc` registry as follows.

**Table 34: ixc registry name mapping**

| Type of access | GEM Registry Name | J2ME Registry Name |
|---|---|---|
| GLOBAL scope | name | dvb:/signed/organisation_id/application_id/name |
| GLOBAL scope | name | dvb:/unsigned/organisation_id/application_id/name |
| GLOBAL scope | /organisation_id/application_id/name | dvb:/signed/organisation_id/application_id/name |
| GLOBAL scope | /organisation_id/application_id/name | dvb:/unsigned/organisation_id/application_id/name |
| SERVICE scope | name | dvb:/service/service_context_id/signed/organisation_id/application_id/name |
| SERVICE scope | name | dvb:/service/service_context_id/unsigned/organisation_id/application_id/name |
| SERVICE scope | /organisation_id/application_id/name | dvb:/service/service_context_id/signed/organisation_id/application_id/name |
| SERVICE scope | /organisation_id/application_id/name | dvb:/service/service_context_id/unsigned/organisation_id/application_id/name |

`organisation_id` and `application_id` refer to the `organisation_id` and `application_id` of the calling Xlet (see clause 10.4.3), and shall be formatted as specified in `org.dvb.io.ixc.IxcRegsitry.lookup()`. In the above table, service_context_id is a unique platform generated opaque name that identifies the service context for the purpose of SERVICE scope inter-xlet communication. The same name shall be used by all applications running in that service context.

The javadoc for the `org.dvb.io.ixc` package is provided in annex Y, "Inter-application and Inter-Xlet communication API". An example is provided in MHP [1], clause W.2 "Example of exporting an object for inter-application communication".

NOTE: Due to the security policy in clauses 12.6.2.14.1 and 12.6.2.14.2, all xlets can also import and export objects in the PBP registry under the "dvb:/ixc/*" namespace. See MHP [1], clause W.6 for an example of how two xlets can securely verify each others' identities.

### 11.7.4 Basic MPEG concepts

NOTE: The API from DAVIC defined in this clause is made optional for packaged media targets in clause 15, "Detailed platform profile definitions" because it relates to transport streams.

This API is formed of the Java classes defined in annex G of DAVIC 1.4.1p9 [17]:

- `ApplicationOrigin`
- `ElementaryStream`
- `Service`
- `TransportStream`

Methods returning instances of elementary stream. service or transport stream shall return instances of the DVB specific subclass (`DvbElementaryStream` etc.), if the GEM terminal specification includes these classes.

### 11.7.5 Resource notification

The resource notification API is defined in annex F of DAVIC 1.4.1p9 [17].

- The `notifyRelease()` method shall be called for all `ResourceClients` where the corresponding resource has been removed without the application releasing it. The `release()` method shall be called for specific resources where time to cleanup is of practical use considering the specific resource in question. The only one of these in the present document is connection oriented return channels.
- The `requestData` parameter of the `requestRelease` method shall implement the `java.rmi.Remote` interface. In APIs using the `ResourceProxy` interface, where the method to reserve the resource has a parameter 'requestData', this parameter shall also implement `the java.rmi.Remote` interface or be `null`. Use of other values shall result in `null` being used when `requestRelease()` is called. Implementing the Remote interface does not force java.rmi to be used when the current owner of the resource is in the same application as code requesting ownership.
- The `resourceProxy.getClient()` method shall always return the `ResourceClient` provided to the platform by the application when that instance of a `ResourceProxy` was created or reserved. The `ResourceClient` for the current owner of a resource is only returned when this method is called on the `ResourceProxy` which currently holds the resource concerned.

This text clarifies the DAVIC javadoc with respect to multiple simultaneous applications. Interpretations of the DAVIC javadoc which are inconsistent with this are excluded.

### 11.7.6 Content referencing

This API is formed of the DAVIC `Locator` class and the `javax.tv.locator` package.

The DAVIC package `org.davic.net.dvb` is not required by the present document. GEM terminal specifications may define one or more subclasses of the DAVIC `Locator` class.

The signature of the `org.davic.net.Locator` class shall be extended with:

`"implements javax.tv.locator.Locator".`

The `createFactory()` method of `javax.tv.locator.LocatorFactory` shall always return `org.davic.net.Locator(s)` which implement the `javax.tv.locator.Locator` interface when provided with a locator syntax that is valid in the terminal specification. See also clause 14.8, "Locators and content referencing".

In the present document, methods whose signature has a return type of `org.davic.net.Locator` or `javax.tv.locator.Locator` shall return an instance of `org.davic.net.Locator` (or a subclass of that) where the locator returned can be represented by the locator syntax described by the terminal specification. In this case, the locator returned shall contain an identification of a service.

Any optional extensions of locators (e.g. for specifying components, events, etc.) are considered in a comparison and if they are not equally present in both locators then the comparison shall fail.

For the above locators "best effort" comparison shall be exact.

The protected constructor of `LocatorFactory` is for implementation use. GEM applications shall not subclass `LocatorFactory`. Implementations are not required to behave correctly if they should do this.

### 11.7.7 Common error reporting

This API is formed of the interface and exceptions defined in annex G of DAVIC 1.4.1p9 [17]:

- `NotAuthorizedInterface`
- `NotAuthorizedException`
- `ObjectUnavailableException`
- `ResourceException`
- `TuningException`

`NotAuthorizedInterface` and `NotAuthorizedException` are required to be present for GEM terminal specifications that include the section filter API. However, for GEM terminal specifications that do not include the conditional access API, `NotAuthorizedException` shall not be generated by the platform.

NOTE: In such terminal specifications, there is consequently no mechanism for obtaining a platform object that implements `NotAuthorizedInterface`.

Support for the `ObjectUnavailableException` and `ResourceException` classes is not required for GEM terminal specifications that do not include the optional support for 11.7.4, "Basic MPEG concepts" in the entry for "common infrastructure" in table Error! Reference source not found. of clause 15, "Detailed platform profile definitions".

Support for the exception `TuningException` is not required for GEM terminal specifications that do not include the MPEG-2 Section Filter API.

### 11.7.8 Plug-in APIs

The plug-in API is defined in annexes AF, "Plug-in APIs" and AE, "Inner Applications" of the present document.

NOTE: This is the package org.dvb.application.plugins.

GEM inter-operable plug-ins signalled with the delegated application descriptor (see clause 10.7.3) shall implement the `org.dvb.application.plugins.Plugin` interface. Failure to implement this interface shall result in the plug-in being ignored. Plug-ins which only execute "delegated" applications signalled in the AIT do not need to implement the `javax.tv.xlet.Xlet` interface as part of the initial entry point from the application manager to the plug-in. Plug-ins which can execute both AIT and natively signalled "delegated" applications shall implement both the Plug-in and the Xlet interfaces.

The semantics of the method `org.dvb.application.AppAttributes.getProperty()` are extended such that when it is called with the string "`dvb.ait.descriptors`", an array of bytes containing the `application_descriptors_loop` shall be returned.

Each running instance of a plug-in shall execute in a single logical VM instance as defined in clause 11.2.1, "Basic Considerations". Hence all delegated application Xlets returned by the initApplication method are all executed in the same logical VM instance. It is the responsibility of the plug-in to ensure the presence or absence of shared context between instances of delegated application Xlets in the same logical VM instance.

## 11.7.9 Provider API

The provider APIs are defined in annex AN.

NOTE: This is the package `org.dvb.spi`.

### 11.7.9.1 API framework

This is the `org.dvb.spi` and `org.dvb.spi.util` packages.

The following tables define the mapping that shall be used between the provider APIs and the corresponding API as called by GEM applications.

### 11.7.9.2 SelectionProvider

This is the `org.dvb.spi.selection` package.

**Table 35: Mapping from javax.tv.service.Service to provider APIs**

| Method | Mapping |
|---|---|
| getLocator | ServiceReference.getLocator. |
| getName | ServiceDescription.getLongName. |
| getServiceType | ServiceDescription.getServiceType. |
| hasMultipleInstances | Calculated by the receiver, returns true if and only if there are multiple services with the same transport independent locator. |
| retrieveDetails | SelectionProvider.getServiceDescriptions, either when the Java TV method is called or previously if the results have been cached and are still valid. |

**Table 36: javax.tv.navigation.ServiceDetails**

| Service | Mapping |
|---|---|
| addServiceComponentChangeListener | None needed. |
| removeServiceComponentChangeListener | |
| retrieveComponents | Not defined in the present document. |
| getDeliverySystemType | ServiceDescription.getDeliverySystemType. |
| getLongName | ServiceDescription.getLongName. |
| getProgramSchedule | No mapping required if schedule descriptions are obtained via standard SI. If schedule descriptions are obtained from a Provider then the mapping is to SimpleSIProvider.getScheduleDescriptions, either when the Java TV method is called or previously if the results have been cached and are still valid. |
| getService | Indirect - implementation must maintain enough information to lookup the service for a ServiceDetails. |
| getServiceType | ServiceDescription.getServiceType. |
| retrieveServiceDescription | Same as for DVB-SI in GEM 1.0 - always fails. |

### 11.7.9.3 SI providers

This is the `org.dvb.spi.si.simple` and `org.dvb.spi.si.full` packages.

**Table 37: javax.tv.guide.ProgramSchedule**

| Service | Mapping |
|---|---|
| addListener | None needed. Events are generated after calls to programDescriptionAvailable and scheduleDescriptionAvailable. |
| removeListener | |
| getServiceLocator | Indirect - implementation must maintain enough information to lookup the service for a ProgramSchedule. |
| retrieveCurrentProgramEvent | SimpleSIProvider.getProgramDescriptions, either when the Java TV method is called or previously if the results have been cached and are still valid. |
| retrieveFutureProgramEvent | |
| retrieveFutureProgramEvents | |
| retrieveNextProgramEvent | |
| retrieveProgramEvent | |

**Table 38: javax.tv.guide.ProgramEvent**

| Service | Mapping |
|---|---|
| getDuration | Derived from ProgramDescription.getStartTime and ProgramDescription.getEndTime. |
| getStartTime | ProgramDescription.getStartTime. |
| getEndTime | ProgramDescription.getEndTime. |
| getName | ProgramDescription.getName. |
| getRating | Not defined in the present document. |
| getService | Indirect - implementation must maintain enough information to lookup the service for a ProgramEvent. |
| retrieveComponents | Not supported. |
| retrieveDescription | ProgramDescription.getDescription. |

#### 11.7.9.4 InteractionChannelTransportProvider

The following table describes the interaction of DSMCCObject with the ICT service provider implementation, including differences in behaviour from classical broadcast carousel objects.

**Table 39**

| DSMCCObject Method | Service Provider Method | Comments |
|---|---|---|
| DSMCCObject(String)<br>DSMCCObject(String, String)<br>DSMCCObject(DSMCCObject, String) | InteractionChannelTransportProvider.createResourceTransportObject(String) | |
| isLoaded() | ResourceTransportObject.isLoaded() | |
| isStream() | - | Not applicable; always return false. |
| isStreamEvent() | - | Not applicable; always return false. |
| isObjectKindKnown() | - | Not applicable; always return false. |
| synchronousLoad() | ResourceTransportObject.synchronousLoad() | |
| asynchronousLoad() | ResourceTransportObject.asynchronousLoad() | |
| abort() | ResourceTransportObject.abort() | |
| prefetch(String, byte)<br>prefetch(DSMCCObject, String, byte) | InteractionChannelTransportProvider.prefetch(String, byte) | |
| unload() | ResourceTransportObject.unload(); | |
| getURL() | - | Return the file: URL describing the abstract pathname with respect to the relevant logical file system root. |
| addObjectChangeEventListener(ObjectChangeEventListener) | ResourceTransportObject.addUpdateListener(ResourceUpdateListener) | |
| removeObjectChangeEventListener(ObjectChangeEventListener) | ResourceTransportObject.removeUpdateListener(ResourceUpdateListener) | |
| loadDirectoryEntry(AsynchronousLoadingEventListener) | ResourceTransportObject.loadMetadata(AsyncResourceLoadListener) | |
| setRetrievalMode(int) | - | Not applicable; has no effect. |
| getSigners()<br>getSigners(boolean) | - | It is expected that the service provider implementation will implement whatever signature verification scheme is appropriate, so these methods will have no effect. An array of length 0 will be returned. |

## 11.7.10 Content referencing for IPTV

This is the `org.dvb.locator.ip` package.

The IPTV content referencing API are defined in annex AU.

## 11.7.11 TV-Anytime content referencing and metadata

These are the `org.dvb.tvanytime.metadata`, and `org.dvb.tvanytime.resolution` packages from TS 102 816 [82] together with the `org.dvb.tvanytime.metadata.ip` package as defined in the present document.

The following shall apply to instances of `javax.tv.locator.Locator` that reference ProgramEvents whose metadata is only available via the TV-Anytime protocols used in TS 102 539 [81].

- If the Locator instance was obtained by explicitly or implicitly resolving a CRID then the implementation shall remember the CRID concerned. Attempts to retrieve the ProgramEvent instance using such a Locator shall result in an attempt to retrieve the metadata associated with the remembered CRID.
- If the external form of the Locator matches the optional ProgramURL element in a TV-Anytime <ScheduleEvent> or <BroadcastEvent> fragment available to the GEM terminal, the GEM terminal shall obtain the CRID from that fragment and then attempt to retrieve the metadata for that CRID.
- If the external form of the Locator includes time and duration and the GEM terminal has schedule information corresponding to that entire duration then it should retrieve and return the metadata for the CRID with the greatest overlap with the time and duration of the locator.
- Otherwise attempts to retrieve the ProgramEvent instance for the Locator instance shall fail with a SIRequestFailureType of DATA_UNAVAILABLE.

For avoidance of doubt, there is no requirement for implementation to attempt reverse resolution of Locators to CRIDS.

The `org.dvb.tvanytime.metadata.ip` package is defined in annex AX.

### 11.7.12 Content referencing for OTT

This is part of the `org.dvb.locator.ott` package.

The OTT content referencing API is defined in annex AU.

## 11.8 Security

### 11.8.1 Basic Security

The GEM platform shall always install a `SecurityManager` before starting any DVB-J applications.

In GEM terminals where the `mhp.smartcard.reader` property is known and has the value "SUPPORTED", GEM applications shall be able to use a Provider installed by a GEM application as defined in annex AJ, "Cryptographics service provider installation," in those methods in the `java.security` package which have a Provider as an input parameter.

### 11.8.2 APIs for return channel security

This API is defined in the J2ME Security (JSSE) Optional Package Specification v1.0 which is found in FP 1.1 [78]. It includes the package `javax.microedition.io`.

NOTE 1: JSSE is optional in PBP 1.1 [4].

Cipher suites listed in table 56 "Profile of cipher suites that implementations are required to support" shall have the name found in the "cipher suite" column of that table with "TLS_" replaced by "SSL_". If other cipher suites from TLS RFC 2246 [58] are supported, it is implementation dependent whether their names start with "TLS_" or "SSL_".

NOTE 2: This language might be interpreted as disallowing these cipher suites to also be reported with a name beginning with "`TLS_`". GEM terminal implementations are not subject to such a restriction; the method `getSupportedCipherSuites()` on the concerned classes may return cipher suite names beginning with "`TLS_`" in addition to the names starting with "`SSL_`" that are required to be returned.

For GEM terminal specifications, this is to be read as requiring that these methods return at least the following constants:

```
SSL_RSA_WITH_NULL_MD5
SSL_RSA_WITH_NULL_SHA
SSL_RSA_EXPORT_WITH_DES40_CBC_SHA
```

```
SSL_RSA_WITH_3DES_EDE_CBC_SHA
SSL_RSA_WITH_DES_CBC_SHA
```

NOTE 3: `SSL_NULL_WITH_NULL_NULL` is not required to be returned. This cipher suite is used internally in the TLS/SSL protocol negotiation. Support of this protocol does not require that it be made available to applications via the getSupportedCipherSuites methods.

NOTE 4: If GEM terminal specifications add cipher suites in addition to the ones above, it may be appropriate for them to require the more current names beginning with `TLS_`, rather than names beginning with `SSL_`.

Also see clause 12.10, "Security on the return channel".

In GEM terminals where the `mhp.smartcard.reader` property is known and has the value "SUPPORTED", MHP applications shall be able to use a Provider installed by a GEM application as defined in annex AJ, "Cryptographics service provider installation" in both KeyManagerFactory and TrustManagerFactory.

## 11.8.3 Additional permissions classes

See annex T, "Permissions".

For GEM terminal specifications that do not support the functional equivalent named "conditional access", the package `org.dvb.net.ca` is not required.

For GEM terminal specifications that do not support the functional equivalent named "SI", the class `org.dvb.net.tuning.DvbNetworkInterfaceSIUtil` is not required.

For GEM terminal specifications that do not support the "Tuning API", in clause 11.6.3, listed in table Error! Reference source not found., the class `org.dvb.net.tuning.TunerPermission` is not required.

## 11.8.4 General Security Issues

Unless otherwise specified, sub-classes of `java.security.Permission` are not required to check the input parameters to their constructors. When a GEM platform constructs these, it is responsible for creating them with legal values. The behaviour of a GEM platform if an application creates such an instance with illegal values is intentionally not specified.

## 11.8.5 Cryptographic API

This API is defined in the J2ME Security (JCE) Optional Package Specification v1.0 part of FP 1.1 [78].

NOTE: The JCE optional package is optional in PBP 1.1 [4].

In GEM terminals where the mhp.smartcard.reader property is known and has the value "SUPPORTED", GEM applications shall be able to use a Provider installed by a GEM application as defined in annex AJ, "Cryptographics service provider installation".

GEM terminals shall include a default provider for the `javax.crypto.Cipher` class as follows:

- In all GEM terminals, the RSA algorithm shall be supported.
- In GEM terminals implementing clause 12.10.2, "TLS cipher suites", the algorithms listed in that clause shall additionally be supported.

## 11.8.6 DVB Extensions for Cryptography

### 11.8.6.1 Introduction (informative)

"Java2 Standard Edition" includes APIs providing the ability to use PKCS11 tokens (such as smart cards) in Java applications. These APIs simplify the way in which Java applications can handle removable smart cards. They are also needed to login into PKCS11 token for non-key related operations such as encryption and decryption.

GEM is based on J2ME Personal Basis Profile (PBP 1.1 [4]) which does not provide these new APIs. For this reason, the following new packages are defined in the present document, see annex AI, "DVB Extensions for cryptography".

- `org.dvb.security`
- `org.dvb.auth.callback`
- `org.dvb.net.ssl`
- `org.dvb.security.pkcs11`

#### 11.8.6.1.1 The org.dvb.security package

This package provides the AuthProvider class which is needed to log into a PKCS11 token for non key related operations. In that case, there is no other way for a java application to send the PIN code.

It also provides the `KeyStoreBuilder` class to install a callback handler which is called when a PIN code is required to create a `KeyStore`.

This class is a convenience feature to simplify applications and there is also a method to give a password when a `KeyStore` is instantiated. Thus it is possible for an application to log into the token for key operations without using a `KeyStoreBuilder`.

#### 11.8.6.1.2 The org.dvb.auth.callback package

This package is required to support the `KeyStoreBuilder` mechanism. Applications are responsible for asking the end-user for the PIN code. Applications will request the PIN code by implementing the interface `CallbackHandler` from this package. This handler will receive a `PasswordCallback` object when the implementation requires the PIN code.

NOTE: In order to prevent other applications listening in, it is recommended that applications use `org.dvb.event` to obtain exclusive access to the number keys when asking for the PIN code.

#### 11.8.6.1.3 The org.dvb.net.ssl package

During the SSL handshake, the SSL engine uses `TrustManagers` to make sure the certificate chain received from the server is trusted. If client authentication is enabled, the SSL engine will use `KeyManagers` to find a certificate chain and a private key according to the `CertificateRequest` message from the server. The `CertificateRequest` message gives a list of acceptable CA for the server.

In J2SE 5.0 the classes `TrustManagerFactory` and `KeyManagerFactory` can be initialized with a set of `KeyStoreBuilder` where keys and certificate material can be found. To have the same functionality in GEM, the classes `DVBTrustManagerFactory` and `DVBKeyManagerFactory` have been defined. The method init in these classes is used by applications to provide a set of `KeyStoreBuilders` to these factory classes.

Without these two classes, applications can only provide the PIN code at the time the `KeyManagerFactory` is created. Hence if the token is replaced, the existing `KeyManagerFactory` cannot be used and the application would need to create a new instance to use with the new token.

#### 11.8.6.1.4 The org.dvb.security.pkcs11 package

In J2SE 5.0, the Sun PKCS11 provider is configured via a text configuration file. This file is used, to indicate which PKCS11 slot is associated with the provider. For GEM, it may be useful for an application to select a particular slot (if there are several slots and tokens). The methods `getSlotList` and `getTokenInfo` of the class `DVBPKCS11Provider` can be used to get the list of slots and tokens available. The default slot used will be defined by the java property "dvb.security.pkcs11.defaultSlotId". If an application needs to use another slot, it should call the method DVBPKCS11Provider.setSlotId(int slotId) to change the slot used by the provider. The slot can only be changed when the provider is not logged into the token.

### 11.8.6.2 Specification

This API is formed of the following package and is defined in annex AI, "DVB Extensions for cryptography".

- `org.dvb.security`
- `org.dvb.security.pkcs11`
- `org.dvb.auth.callback`
- `org.dvb.net.ssl`

Installation of cryptographic service providers shall be supported as defined in annex AJ, "Cryptographics service provider installation".

# 11.9 Other APIs

## 11.9.1 Timer support

This API is formed of the Timer API defined in Java TV [16] in the `javax.tv.util` package.

Implementations are required to meet the following specifications:

- Minimum repeat interval less than or equal to 40 ms.
- Granularity less than or equal to 10 ms.

The only condition defined in GEM where `TVTimer.scheduleTimerSpec` may throw `TVTimerScheduleFailedException` is if the GEM terminal is unable to provide any more timers. See table 88, "Minimum requirements for other resources".

NOTE: The minimum repeat interval of 40ms was motivated by a standard definition frame rate of 25 Hz, however this was not meant to imply that the timer could be used for frame-accurate animation.

## 11.9.2 User settings and preferences API

This API is defined in annex L, "User settings and preferences API".

The preferences listed below shall be accessible to all applications.

- User Language.
- Parental Rating.
- Country Code.
- Default Font Size.

Other preferences shall only be accessible where a `UserPreferencePermission` for "read" has been granted (see clause 11.10.2.8, "org.dvb.user.UserPreferencePermission").

## 11.9.3 Profile and version properties

All of the system properties defined in this clause are required. For GEM-based terminal specifications, the properties indicating the profile ("mhp.profile.enhanced_broadcast", "mhp.profile.interactive_broadcast" and "mhp.profile.internet_access") are to be interpreted as referring to the profile descriptions in clause 15. The properties referring to version numbers are to be interpreted as referring to the corresponding MHP version number.

NOTE: This means that a receiver implementing a terminal specification based on the interactive broadcast profile of GEM would return property values consistent with the MHP [1] interactive broadcast profile.

Applications can discover the supported profile and the version of the profile (and thus, what functionality is supported) by retrieving profile and version properties. If a particular profile is not supported then the related version properties shall return null.

More specifically, the properties listed in table 40 shall be included in the property set of the `java.lang.System` class. Thus these properties can be retrieved using `java.lang.System.getProperty()`. Since this API returns a string, numeric return values shall be encoded as defined by `java.lang.Integer.toString(int)`.

**Table 40: System properties for profile and version interrogation**

| Property | Semantics | Possible values | Example |
|---|---|---|---|
| mhp.profile.enhanced_broadcast | Indicates whether the enhanced broadcast profile is supported | "YES" | "YES" |
| mhp.profile.interactive_broadcast | Indicates whether the interactive broadcast profile is supported | "YES", null | null |
| mhp.profile.internet_access | Indicates whether the internet access profile is supported | "YES", null | null |
| mhp.eb.version.major | Major version number of the supported enhanced broadcast profile | Non-negative integer value | "1" |
| mhp.eb.version.minor | Minor version number of the supported enhanced broadcast profile | Non-negative integer value | "0" |
| mhp.eb.version.micro | Micro version number of the supported enhanced broadcast profile | Non-negative integer value | "0" |
| mhp.ib.version.major | Major version number of the supported interactive broadcast profile | Non-negative integer value | "1" |
| mhp.ib.version.minor | Minor version number of the supported interactive broadcast profile | Non-negative integer value | "0" |
| mhp.ib.version.micro | Micro version number of the supported interactive broadcast profile | Non-negative integer value | "0" |
| mhp.ia.version.major | Major version number of the supported internet access profile | Non-negative integer value | "1" |
| mhp.ia.version.minor | Minor version number of the supported internet access profile | Non-negative integer value | "0" |
| mhp.ia.version.micro | Micro version number of the supported internet access profile | Non-negative integer value | "0" |
| NOTE: In previous versions of the present document, "NO" might be returned instead of null for profiles which are not supported. | | | |

The properties for querying the version of a profile which is not supported by the GEM terminal shall not be supported. Hence, calling `System.getProperty()` for these properties shall return null.

### 11.9.3.1 Information on options

Clause 15, "Detailed platform profile definitions" defines what is mandatory and optional for each profile. In order to give an application information about which options a particular GEM implementation supports, a property string is defined for each option (with the same granularity as in clause 15).

A GEM implementation supports an option if and only if the corresponding property is known and its value is "SUPPORTED". The properties are part of the property set of the java.lang.System class.

The general syntax of the properties that indicate whether a certain feature is supported or not is:

```
mhp.option.<the optional feature>.
```

The table 41 lists the currently defined options.

**Table 41: System properties for optional feature interrogation**

| Option | Property |
|---|---|
| IP Multicast over Broadcast Channel | mhp.option.ip.multicast |
| HTTP 1.1 over the interaction channel | mhp.option.http.1.1 |
| DSMCC user-to-user over the interaction channel | mhp.option.dsmcc.uu |
| DVB-HTML | mhp.option.dvb.html |
| Does this GEM receiver have a smart card reader accessible to applications through the smart card reader API? | mhp.smartcard.reader |
| Memory for stored services (see clause 9.6.2, "Stored services") is greater than zero (see note) | mhp.stored.services |
| Memory card access (see clause 11.5.7) | mhp.option.memory.card |
| OpenType (see clauses 7.4.2 and 11.4.1.5.2) | mhp.option.opentype |
| High Definition (see clauses 7.2.2, "Video" and G.1.1.3, "High definition") | mhp.option.highdef |
| Buffered animation (see org.dvb.ui.BufferedAnimation) | mhp.option.buffered.animation |
| Broadband content guide | mhp.option.broadband.content.guide |
| Privileged applications | mhp.option.privileged.applications |
| NOTE: The available memory may be insufficient to store a particular application and hence requests to add applications to stored services may fail even though this property is both known and "SUPPORTED". | |

## 11.9.4 Non-CA smart card API

The API for access to non-conditional smart cards shall be comprised of the following:

- the "SATSA-APDU" optional package defined by SATSA [70];
- the class`javax.microedition.apdu.APDUPermission` defined in clause B.1.2.1 of that document;
- the org.dvb.smartcard package defined in annex AM, "Smart card reader API".

NOTE 1: For GEM terminals without a non-CA smart card reader, the failure mode is defined by SATSA [70].

GEM does not require support for U(SAT).

NOTE 2: SATSA's Generic Connection Framework calls for opening a Connection to a Java Application, identified by an ID (the AID). At the time of this writing, many of the smart cards of interest are standards based but are not Java cards. Although an Application ID could be set on these cards, it is not mandatory and rarely available.

NOTE 3: Any method call of the form Connector.open("apdu...;target=a0.00...") issued against such cards returns a ConnectionNotFound exception. This is because the card has neither an AID set nor is a Java Card with such application listening.

NOTE 4: Any method call of the form Connector.open("apdu...;target=SAT") may also fail as this is normally used on (U)SIM as defined per SIM Application Toolkit mode.

In order to open the APDUConnection with this kind of smartcards, the method openDefaultConnection() in org.dvb.smartcard.SmartCardReader must be used.

In these APDUConnection instances, the Application Selection (SELECT FILE by DF NAME) and MANAGE CHANNEL commands are allowed.

NOTE 5: To open the APDUConnection when the smart card contains AID or it is a (U)SIM, the GCF method can still be used. In this case, Application Selection (SELECT FILE by DF NAME) and MANAGE CHANNEL commands are not allowed.

### 11.9.5 XML parsing API

#### 11.9.5.1 SAX

The XML parsing API is defined in clause 2, "JAXP Subset", of JSR 172 [76].

#### 11.9.5.2 JDOM

This is the org.dvb.xml.jdom package from TS 102 816 [82].

### 11.9.6 GEM terminal hardware API

This is the org.dvb.hardware package defined in annex AZ, "MHP terminal hardware API".

### 11.9.7 Content Download API

The content download API is described in Annex BA.

The content download API is based on the shared DVR API for scheduling and managing recording requests [85].

## 11.10 Java permissions

Under the security model for DVB-J applications, a permissions file is associated with authenticated applications; unauthenticated applications operate within a sandbox environment, which means certain forms of locators will not operate, and certain APIs will not be available. Authenticated applications may be granted permission to use the restricted locators and APIs.

One mechanism by which an application may be trusted, and thus request additional permissions, is for that application to be signed. As described in clause 12.1.3, GEM terminal specifications may add other mechanisms for establishing that an application is trusted. Thus, in this clause the term "signed application" is to be interpreted as meaning an application that is eligible for being granted permissions beyond the DVB-J sandbox. "Unsigned application" is to be interpreted as meaning an application that has not been packaged in such a way.

This clause explains how the permissions that are defined to be included in the sandbox that is available to unsigned applications and the permissions that can be requested in the permission request file are mapped to the Permission objects in the Java platform.

### 11.10.1 Permissions for unsigned applications

The GEM security policy includes a set of resources that are always guaranteed to be granted to applications, if the application is executed. Unsigned applications have access to only these resources.

This clause defines the mapping of those resources to the Java Permission objects.

#### 11.10.1.1 java.awt.AWTPermission

This control access to sensitive parts of AWT which is not needed for GEM applications. This shall be denied for both unsigned and signed applications.

#### 11.10.1.2 java.net.SocketPermission

Because access to return channel is not within the sandbox, this is not required for unsigned applications.

#### 11.10.1.3 java.util.PropertyPermission

For unsigned applications, a read permission shall be granted for all properties defined in the present document except for those explicitly identified as only available for signed applications. The permission shall be denied for the action string "write".

A read permission may also be granted for other properties except for those explicitly identified as only available for signed applications.

#### 11.10.1.4 java.lang.RuntimePermission

This permission shall be denied for both unsigned and signed applications.

#### 11.10.1.5 java.io.SerializablePermission

This permission shall be denied for both unsigned and signed applications.

#### 11.10.1.6 java.io.FilePermission

A read permission shall be granted for the subtree under which the implementation mounts the object carousels.

#### 11.10.1.7 javax.tv.media.MediaSelectPermission

The Media API (i.e. JMF) is within the sandbox, so all applications shall be granted a javax.tv.media.MediaSelectPermission with a locator string "*" that indicates access to all media streams.

#### 11.10.1.8 javax.tv.service.ReadPermission

Access to Service Information is within the sandbox, so all applications shall be granted a javax.tv.service.ReadPermission with a locator string "*".

#### 11.10.1.9 javax.tv.service.selection.ServiceContextPermission

All applications shall be granted a `javax.tv.service.selection.ServiceContextPermission` with a name string `"getServiceContentHandlers"` and action string `"own"`.

`ServiceContextPermission( "access", "own" )` shall be granted to all MHP applications.
`ServiceContextPermission( "access", "*" )` shall not be granted.

#### 11.10.1.10 java.util.Locale.setDefault

This method shall throw a security exception.

#### 11.10.1.11 Applications signalled in AIT File

Applications signalled in an AIT File (clause 10.4.5, "AIT File") shall have an instance of `SocketPermission` for each combination of host and port number listed in all transport protocol descriptors with the protocol ID value 0x0003 in both their AIT entry and in the "common" descriptor loop. Each permission shall have the "connect" action only.

#### 11.10.1.12 javax.microedition.xlet.ixc.IxcPermission

Unsigned applications shall be granted:

- IxcPermission("dvb:/unsigned/*", "lookup").
- IxcPermission("dvb:/unsigned/organisation_id/application_id/*", "bind").
- IxcPermission("dvb:/ixc/organisation_id/application_id/*", "bind").
- IxcPermission("dvb:/ixc/*", "lookup").
- IxcPermission("dvb:/service/service_context_id/unsigned/*", "lookup").
- IxcPermission("dvb:/service/service_context_id/unsigned/organisation_id/application_id/*", "bind").

`organisation_id` and application_id refer to the `organisation_id` and application_id of the application being granted the permission (see clause 10.4.3 "Content of the Application Description"), and shall be formatted as specified in org.dvb.io.ixc.IxcRegistry.lookup() and service_context_id is the platform generated opaque name of the service context (see clause 11.7.3, "Inter-application communication API").

#### 11.10.1.13 MonitorAppPermission and ServiceTypePermission

Unsigned applications shall not be granted MonitorAppPermission or ServiceTypePermission.

### 11.10.2 Additional Permissions for signed applications

Signed applications can additionally request to get more permissions. These permissions are requested using the permission request file (clause 12.6, "Security policy for applications"). This clause defines the mapping from the items in the permission request file to the Java Permissions that may be granted by the GEM terminal in response to the request.

#### 11.10.2.1 java.util.PropertyPermission

For signed applications, a read permission shall be granted for all properties for unsigned applications and `dvb.persistent.root`.

#### 11.10.2.2 java.io.FilePermission

A read permission shall be granted for the subtree under which the implementation mounts the object carousels or any other filesystem that may be mounted using the DSMCC APIs.

The following occurs when the permission request file requests the permission to access persistent storage and this is granted:

1) A `FilePermission` that permits access to the persistent storage directory subtree is created.

NOTE 1: This `FilePermission` created complies with the access rights defined in clause 12.6.2.7.2, "Policy for signed applications".

2) Where a GEM terminal supports access to memory cards (see clause 11.5.7, "File Storage Device Access"), the implementation shall create instances of `java.io.FilePermission` allowing access to the locations where the file systems for a memory card will appear in its file system namespace.

When there is a persistent file credential in the permission request file and this is granted, `FilePermissions` are created as follows:

- file path = value of `dvb.persistent.root` property + filename from the credential.
- action = string containing "read" if "read" is indicated in the credential; or string containing "write, delete" if "write" is indicated in the credential".

NOTE 2: A single instance of `FilePermission` is not sufficient to express the limitations on accessing files and directories through credentials required by clause 12.6.2.6, "Credentials". Implementations are responsible for enforcing those requirements and cannot rely on a single instance of `FilePermission` to achieve this.

#### 11.10.2.3 org.dvb.net.ca.CAPermission

When the permission request file requests the permission to communicate with a CA system and this is granted, a `CAPermission` is created as follows:

The CA system ID string from the permission request file is directly used as the first part of the string used for the `CApermission` constructor, this is concatenated with a colon character and the list of the attribute strings based on the attributes listed as true in the permission request file.

This clause does not apply for GEM terminal specifications that do not include the functional equivalent named "Conditional Access" as defined in clause 15.6, "Functional equivalents".

#### 11.10.2.4 org.dvb.application.AppsControlPermission

When the permission request file requests the permission to have additional permissions to control the lifecycle of applications and this is granted, an `AppsControlPermission` is created.

#### 11.10.2.5 org.dvb.net.rc.RCPermission

When the permission request file requests the permission to communicate through the return channel and this is granted, an `RCPermission` is created as follows:

- For the default ISP item, the `RCPermission` is created with "target:default" string.
- For items with phone numbers in them, the string is "target:" + the phone number prefix in the permission request file + "*".

On `org.dvb.net.rc.ConnectionRCInterface`, the method `getCurrentTarget` shall always throw a `SecurityException`. The method `setTarget` shall throw a security exception where the application doesn't have the permission to use the target specified. The method `setTargetToDefault` shall throw a security exception where the application doesn't have either "target:default" or "target:*" permissions.

#### 11.10.2.6 org.dvb.net.tuning.TunerPermission

When the permission request file requests the permission to access the Tuning API and this is granted, an `TunerPermission` is created.

If a GEM terminal specification does not support the "Tuning API", in clause 11.6.3, listed in table Error! Reference source not found., the class `org.dvb.net.tuning.TunerPermission` is not required by the present document and this clause does not apply.

#### 11.10.2.7 javax.tv.service.selection.SelectPermission

By default a `SelectPermission` is created with a locator "*" and action string "own" and a `ServiceContextPermission` is created with a name "*" and action string "own", unless denied by an entry in the permission request file.

GEM terminal specifications may define types of service other than broadcast. The present document allows replacement of this clause to only apply to broadcast services. Hence, applications will have the permissions needed to select all broadcast services in their own service context(s) unless denied by an entry in the permission request file.

#### 11.10.2.8 org.dvb.user.UserPreferencePermission

When the permission request file requests the permission to read and/or write user preferences and this is granted, a `UserPreferencePermission` is created as follows:

- When the permission request file includes "true" for the "read" attribute and this is granted, a `UserPreferencePermission` is created with the string "read".
- When the permission request file includes "true" for the "write" attribute and this is granted, a `UserPreferencePermission` is created with the string "write".

#### 11.10.2.9 java.net.SocketPermission

When the permission request file requests the permission to communicate with remote hosts and this is granted, `SocketPermissions` are created with the host and action as indicated in the permission request file.

These permissions shall not be granted in MHP terminals where all return channels are represented by instances of `org.dvb.net.rc.ConnectionRCInterface` (i.e. where all return channels are connection oriented) unless an instance of `org.dvb.net.rc.RCPermission` is also granted.

#### 11.10.2.10 org.dvb.media.DripFeedPermission

When the permission request file requests the permission to use the drip feed feature and this is granted, a `DripFeedPermission` shall be created.

#### 11.10.2.11 org.dvb.application.storage.ApplicationStoragePermission

When the permission request file requests the permission to store applications and this is granted, a ApplicationStoragePermission is created. This permission class is defined in annex AG, "Stored application APIs".

#### 11.10.2.12 javax.microedition.apdu.APDUPermission

When the permission request file requests the permission to access the smart card API and this is granted, an instance of `javax.microedition.apdu.APDUPermission` is created with the name "aid".

#### 11.10.2.13 ServiceContextPermission

On MHP terminals supporting the internet access profile, all applications which are granted a SelectPermission (see clause 11.10.2.7, "javax.tv.service.selection.SelectPermission") shall be granted instances of `javax.tv.service.Selection.ServiceContextPermission` with names:

- "create";
- "destroy";
- "stop";

all of these to have the actions string "own".

#### 11.10.2.14 javax.microedition.xlet.ixc.IxcPermission

Signed applications shall be granted:

- IxcPermission("dvb:/signed/*", "lookup").
- IxcPermission("dvb:/signed/organisation_id/application_id/*", "bind").
- IxcPermission("dvb:/ixc/organisation_id/application_id/*", "bind").
- IxcPermission("dvb:/ixc/*", "lookup").
- IxcPermission("dvb:/service/service_context_id/signed/*", "lookup").
- IxcPermission("dvb:/service/service_context_id/signed/organisation_id/application_id/*", "bind").

`organisation_id` and application_id refer to the `organisation_id` and application_id of the application being granted the permission (see clause 10.4.3 "Content of the Application Description"), and shall be formatted as specified in org.dvb.io.ixc.IxcRegistry.lookup() and service_context_id is the platform generated opaque name of the service context (see clause 11.7.3, "Inter-application communication API").

#### 11.10.2.15 org.dvb.spi.ProviderPermission

Where the permission request file requests the right to install providers and this is granted, an instance of ProviderPermission shall be created for the classname listed. If the scope attribute of the request is "application" then the action of the permission shall be "xlet". If the scope attribute of the request is "system" then the action of the permission shall be "system". Permissions with the action "system" shall only be granted to privileged applications authenticated according to clause 12.18, "Authentication of privileged applications".

#### 11.10.2.16 Permissions for Unbound and Privileged Applications

For signed applications:

- When the permission request file requests servicetypepermission and this is granted, an instance of ServiceTypePermission shall be created with the type and action requested in the permission request file.
- When the permission request file requests "monitorapplication" capabilities and this is granted, instances of MonitorAppPermission shall be created for each capability granted.

NOTE 1: See clause 12.18, "Authentication of privileged applications" for requirements on the granting of "monitorapplication" capabilities.

NOTE 2: Clause 10.2.2.2.3 of OCAP [5] (included in the present document by clause 9.13.2, "Service model") defines a number of requirements relating to the granting of Permissions to applications including privileged applications.

## 11.11 Content referencing

The following mapping shall be used between the types of locator defined in table 59, "Addressable entities, locators and their text representation" and the DVB-J methods defined in this clause. It lists the Java methods and constructors that accept or return (as defined by their method signature) instances of `org.davic.net.Locator`, `javax.tv.locator.Locator`, `javax.media.MediaLocator` or their subclasses. The external form of the locators shall as described in table 59, "Addressable entities, locators and their text representation" for the corresponding entity being referenced. Where the same method is listed as accepting multiple forms of locator, then it is required to accept all forms listed in this clause.

Where a method listed below is defined (in its specification) to check its input then it shall only accept the forms of locator listed below as being valid for that method from among those defined in the present document and in the GEM terminal specification. Other forms of locator from among those defined in the present document shall be rejected as specified for the method concerned. If a method does not specify a means of rejecting inappropriate locators then it shall fail silently apart from Exceptions and Events which do not check their input and where it is the responsibility of the platform to use correct locators when constructing them. The present document does not prevent methods accepting other forms of locator that are not defined in the present document.

### 11.11.1 Transport stream

For packaged media targets, support of locators to reference a transport stream are optional.

The term "DVB locators" is considered to refer to all valid locators as described in table 59.

The following methods used in the present document shall accept or return instances of Objects which describe an MPEG transport stream. Methods which accept a locator for a transport stream as an input parameter shall also accept all other DVB locators which include the information to identify a transport stream. If present, service_id, component_tag, event_id and path_segments shall be retained but not used except where there is both a method that accepts such a locator and a query method specified to return that input. In this case the specified semantics of the query method shall be respected.

- `javax.tv.service.transport.TransportStream.getLocator()`.
- `javax.tv.service.transport.TransportStreamCollection.retrieveTransportStream()`.

NOTE 1: This requirement holds only if the terminal supports `TransportStreamCollection` objects.

- `org.davic.net.tuning.StreamTable.getTransportStreams()`.
- `org.davic.net.tuning.NetworkInterfaceController.tune()`.
- `org.davic.net.tuning.NetworkInterface.getLocator()`.
- `org.davic.net.tuning.NetworkInterfaceController.reserveFor()`.
- `org.davic.net.tuning.StreamTable.listTransportStreams()`.
- `org.dvb.si.SITransportStream.getDvbLocator()`.

NOTE 2: For GEM terminal specifications that include the functional equivalent named "Content Referencing" as defined in clause 15.6, "Functional equivalents", the numeric identifiers in the locator are matched in the DVB-SI tables as follows: The numeric identifiers original_network_id and transport_stream_id are matched against the corresponding fields in the NIT. If present, the network_id is matched against the corresponding field in the NIT.

Additionally, `org.davic.net.tuning.NetworkInterface.getLocator()` shall return a DvbNetworkBoundLocator for the network to which the NetworkInterface is connected.

## 11.11.2 Network

The term "DVB network" is to be interpreted as referring to a valid network, as described in table 59.

For the packaged media target, `NetworkCollection` and `Network` objects are never returned, as described in clause 14.8, "Locators and content referencing". For this reason, this clause does not apply to the packaged media target.

The following methods used in the present document shall accept or return instances of Objects which describe a DVB network.

- `javax.tv.service.transport.NetworkCollection.retrieveNetwork()`.
- `javax.tv.service.transport.Network.getLocator()`.

## 11.11.3 Void

## 11.11.4 Service

### 11.11.4.1 MPEG/GEM specific service

The following methods used in the present document shall accept or return instances of Objects which describe GEM services and PSI-only services. Methods which accept a locator for a MPEG/GEM service as an input parameter shall also accept all other GEM locators which include the information to identify a service. If present, component_tag, event_id and path_segments shall be retained but not used except where there is both a method that accepts such a locator and a query method specified to return that input. In this case the specified semantics of the query method shall be respected.

- `org.dvb.dsmcc.DSMCCStream.getStreamLocator` where the method `isMPEGProgram` returns true.
- `org.dvb.dsmcc.ServiceDomain.attach()`.
- `org.dvb.dsmcc.ServiceDomain.getLocator()`.
- `org.dvb.application.AppAttributes.getServiceLocator()` where the service is an MPEG / DVB service.
- `org.dvb.dsmcc.ServiceXFRReference` - constructor where the service is a MPEG/DVB service.
- `org.dvb.dsmcc.ServiceXFRReference.getLocator()` - where the service is a MPEG/DVB service.

NOTE: The numeric identifiers in the locator are matched in the DVB-SI tables as follows: the numeric identifiers original_network_id, transport_stream_id (if present) and service_id are matched against the corresponding fields in the SDT. If present, the network_id are matched against the corresponding field in the NIT.

The following methods are not required by the present document:

- `org.davic.net.ca.CAModule.buyEntitlement();`
- `org.davic.net.ca.CAModule.queryEntitlement();`
- `org.dvb.si.SIDatabase.retrieveSIService();`

- `org.dvb.si.SIDatabase.retrievePMTService();`
- `org.dvb.si.PMTService.getDvbLocator();`
- `org.dvb.si.SIBouquet.getSIServiceLocators();`
- `org.dvb.si.SIService.getDvbLocator();`
- `org.davic.net.ca.TuneRequestEvent - constructor;`
- `org.davic.net.ca.TuneRequestEvent.getLocator().`

The semantics for these methods apply only if the GEM terminal specification requires a given method.

#### 11.11.4.2 Generic service

The following methods used in the present document shall accept or return instances of Objects which reference generic services. These methods are also required to accept or return the same locators as in the previous clause.

- `javax.tv.service.navigation.LocatorFilter` - constructor.
- `javax.tv.service.navigation.LocatorFilter.getFilterValue().`
- `javax.tv.service.SIManager.getService().`
- `javax.tv.service.navigation.ServiceDetails.getLocator().`
- `javax.tv.service.Service.getLocator().`
- `javax.tv.service.SIManager.retrieveServiceDetails().`
- `javax.tv.service.navigation.ServiceList.findService().`
- `org.dvb.application.AppAttributes.getServiceLocator()` - where the service is not a MPEG/GEM specific one.
- `org.dvb.dsmcc.ServiceXFRReference` - constructor where the service is not a MPEG/GEM specific one.
- `org.dvb.dsmcc.ServiceXFRReference.getLocator()` where the service is not a MPEG/GEM specific one.
- `org.dvb.dsmcc.ServiceXFRException` - constructor.
- `org.davic.media.MediaLocator` -constructor.
- `javax.media.MediaLocator` - constructor.
- `org.dvb.locator.NetworkInterfaceBoundMediaLocator` - constructor.
- `javax.media.Manager.createPlayer(MediaLocator).`
- `javax.media.Manager.createDataSource(MediaLocator).`
- Constructor of various JMF events consumes JMF `MediaLocators`:
  - `org.dvb.media.CAStopEvent, org.dvb.media.PresentationChangedEvent, org.dvb.media.ServiceRemovedEvent, org.dvb.media.NoComponentSelectedEvent.`
- Various JMF events return JMF `MediaLocators` which were passed into their constructors:
  - `org.dvb.media.CAStopEvent, org.dvb.media.PresentationChangedEvent, org.dvb.media.ServiceRemovedEvent,org.dvb.media.NoComponentSelectedEvent.`
- `StoredApplicationService.getLocator` - return value.

- `InternetClient.getServiceContentLocators` - return value.

### 11.11.4.3 Stored services

The following methods used in the present document shall accept or return instances of Objects which describe a stored service.

- `org.dvb.application.storage.StoredApplicationService.getLocator.javax.tv.`

### 11.11.4.4 Content referencing for IPTV

All the methods listed in clauses 11.11.4.1, "MPEG/GEM specific service" and 11.11.4.2, "Generic service" of the present document shall accept or return instances of Objects which describe unicast and multicast GEM IP services.

- `service.Service.getLocator` - when the Service is a `StoredApplicationService.`

### 11.11.4.5 Content referencing for OTT services

All the methods listed in clauses 11.11.4.1, "MPEG/GEM specific service" and 11.11.4.2, "Generic service" of the present document shall accept or return instances of Objects, which describe OTT services addressed by HTTP locators.

An HTTPLocator is defined as described in Table 60 and detailed in the Annex AU extension below. This locator references either Media content, a Manifest file as described in clause 7.2.5, ”Streaming Manifest” or an application.

If an HTTP locator references a manifest file, it can be used to obtain a service. This service can then be selected and the platform will perform all necessary steps (manifest parsing, switching from one representation to another based on certain conditions, …).

In GEM terminals where the HTTPS protocol (clause 6.3.7.3, "HTTPS") is supported, an HTTP locator can be used also to address content delivered via HTTPS,

## 11.11.5 Program event

The following methods used in GEM shall accept or return instances of Objects which describe a program event.

- `javax.tv.service.guide.ProgramEvent.getLocator().`
- `javax.tv.service.SIManager.retrieveProgramEvent().`
- `javax.tv.service.guide.ProgramSchedule.retrieveProgramEvent().`

The following methods are *not* required by GEM:

- `org.davic.net.ca.CAModule.buyEntitlement();`
- `org.davic.net.ca.CAModule.queryEntitlement();`
- `org.dvb.si.SIEvent.getDvbLocator().`

The semantics for these methods only apply if the GEM terminal specification requires a given method.

## 11.11.6 MPEG elementary stream

The following methods used in the present document shall accept or return instances of Objects which describe an MPEG elementary stream. Methods below which accept as an input parameter an array of locators shall also accept GEM locators including a reference to multiple components.

- `javax.tv.net.InterfaceMap.getLocalAddress().`
- `javax.tv.service.selection.InvalidServiceComponentException` - constructor.
- `javax.tv.media.MediaSelectControl` - all methods accepting or returning instances of javax.tv.locator.Locator.

- `javax.tv.media.MediaSelectEvent` and subclasses - constructor.
- `javax.tv.media.MediaSelectPermission` - constructor.
- `javax.tv.service.selection.ServiceContext.select()`.
- `org.dvb.dsmcc.ServiceDomain.attach()`.
- `org.dvb.dsmcc.ServiceDomain.getLocator()`.
- `javax.tv.media.MediaSelectControl.getCurrentSelection()`.
- `javax.tv.service.navigation.ServiceComponent.getLocator()`.
- `javax.tv.media.MediaSelectEvent.getSelection()` (also subclasses).
- `org.davic.media.MediaLocator` - constructor - shall also accept multiple component tag GEM locator.
- `org.dvb.locator.NetworkInterfaceBoundMediaLocator` - constructor - shall also accept multiple component tag GEM locator.
- `javax.media.MediaLocator` - constructor - shall also accept multiple component tag GEM locator.
- `javax.media.manager.createPlayer( MediaLocator )` - shall also accept multiple component tag GEM locator.
- `javax.tv.service.selection.InvalidServiceComponentException.getInvalidServiceComponent()`.
- `javax.tv.service.selection.ServiceContentHandler.getServiceContentLocators()`.

NOTE: The numeric identifiers in the locator are matched in the DVB-SI tables as follows: the numeric identifiers original_network_id, transport_stream_id (if present) and service_id are matched against the corresponding fields in the SDT. If present, the network_id are matched against the corresponding field in the NIT.

Methods which accept a locator or an array of locators for an MPEG elementary stream as an input parameter shall also accept all other GEM locators which include the information to identify a MPEG elementary stream. If present, `event_id` and `path_segments` shall be retained but not used except where there is both a method which accepts such a locator as input and a query method specified to return that input. In this case, the specified semantics of the query method shall be respected.

The methods on `javax.media.MediaSelectControl` shall accept locators for elementary streams as input in GEM terminal specifications where the functional equivalents for broadcast streaming video (defined in clause 7.2.2, "Video") and audio (defined in clause 7.2.1, "Audio") are carried as MPEG elementary streams.

The following methods are *not* required by the present document:

- `org.dvb.si.SIDatabase.retrievePMTElementaryStreams();`
- `org.dvb.si.PMTElementaryStream.getDvbLocator();`
- `org.davic.net.ca.DescramblingStoppedEvent.getServiceLocator();`
- `org.davic.net.ca.DescramblingStartedEvent.getServiceLocator()`.

The semantics for these methods only apply if the GEM terminal specification requires a given method.

### 11.11.7 File

The following methods used in the present document shall accept or return locators which reference files.

- `org.davic.media.MediaLocator` -constructor - for audio files intended to be played from memory.

- `javax.media.MediaLocator` - constructor - for audio files intended to be played from memory.
- `javax.media.Manager.createPlayer(MediaLocator)` - for audio files intended to be played from memory.
- `javax.media.Manager.createDataSource(MediaLocator)` - for audio files intended to be played from memory.
- `org.dvb.dsmcc.ServiceDomain.getURL(Locator)` - locators referring to File or Directory entities, as defined in table 59.

Apart from the above, all file references shall be encapsulated in instances of java.net.URL.

### 11.11.8 Directory

The following method shall return an instance of the GEM locator referencing a directory.

- `org.dvb.application.AppIcon.getLocator()`

### 11.11.9 Drip feed decoder

The following methods used in the present document shall accept or return locators which reference the "drip feed" decoder.

- `javax.media.MediaLocator` - constructor
- `javax.media.Manager.createDataSource(MediaLocator)`

### 11.11.10 Void

### 11.11.11 Methods working on many locator types

The following methods used in the present document work on many locator types. The locator types which each method is required to support are listed for each of the methods concerned.

- `javax.tv.locator.LocatorFactory.transformLocator` - transforms a transport independent locator into a transport dependent one:
  - required to accept instances of `org.davic.net.Locator` describing a transport independent service;
  - required to return instances of `org.davic.net.Locator` describing a transport dependent service.
- `javax.tv.locator.LocatorFactory.createLocator` - creates a locator from a string:
  - required to accept valid GEM locators (see clause 14.8) and return corresponding instances of `org.davic.net.Locator`.
- `javax.tv.service.SIManager.registerInterest` - accepts a locator referencing one or more `SIElements` as input.
- `javax.tv.service.SIManager.retrieveSIElement` - accepts a locator referencing one or more `SIElements` as input:
  - both these methods are required to accept locators referencing:-Bouquet, Network, Event, ElementaryStream, Service, TransportStream.
- `javax.tv.service.SIElement.getLocator`:
  - returns a locator for "this SIElement" as specified by the JavaTV specified sub-interfaces.

GEM terminal specifications may add rules similar to the above for locator types that they specify.

### 11.11.12 Support for the HTTP Protocol in DVB-J

In GEM terminals where an HTTP protocol (clause 6.3.7.1, "HTTP 1.1" or clause 6.3.7.2, "GEM profile of HTTP 1.0") is supported, the following classes and methods shall support the HTTP protocol concerned. In GEM terminals where the HTTPS protocol (clause 6.3.7.3, "HTTPS") is supported, the following class and methods shall support that protocol.

- The constructor for javax.media.MediaLocator - for referencing audio files intended to be played from memory.
- Methods on javax.media.Manager accepting javax.media.MediaLocator as input parameters - for constructing JMF players for audio files intended to be played from memory.
- Methods in the present document which accept instances of java.net.URL are required to accept instances which encapsulate an "http" URL and behave according to their specification. - e.g org.havi.ui.HSound.load(java.net.URL).
- On GEM terminals supporting applications downloaded over the interaction channel as defined in clause 9.6 "Services and applications not related to conventional DVB services", the method `LocatorFactory.createLocator(String)` shall additionally accept Strings containing URLs using the "http" and "https" protocols as being valid and return a corresponding Locator. This method shall only validate the string to the extent that this is possible without network access.
- On GEM terminals supporting applications downloaded over the interaction channel as defined in clause 9.6 "Services and applications not related to conventional DVB services", the method `SIManager.getService(Locator)` shall accept such Locators as being valid and return a corresponding `javax.tv.service.Service`. This method shall only validate the locator to the extent that this is possible without network access.

When HTTP URLs are used with instances of DVBClassLoader to load DVB-J classes over the interaction channel in a signed application, the requirements of the GEM security model shall be complied with before a class is allowed to be successfully loaded from such a URL.

### 11.11.13 GEM Applications

The following methods used in the present document shall accept or return instances of Locators which identify a GEM application:

- `javax.tv.service.selection.ServiceContext.select(Locator [] ).`

## 11.12 Stand-alone Applications

### 11.12.1 Common behavior

The extended GEM application model allows several forms of applications to execute in a stand-alone mode independent of any broadcast services. The environment of these stand-alone applications shall be as follows:

- Instances of `org.dvb.si.SIDatabase` may not return any SI objects if the corresponding network interface isn't tuned to anything. It is implementation dependent whether some network interfaces are left tuned to some previously used transport stream.

For DVB-J applications, the behaviour defined in clause 9.6.4, "Common behaviour" shall mean the following:

- The method `ServiceContext.getServiceContentHandlers` shall not return a `ServiceContentHandler` for any such existing video or audio for a stand-alone stored application.
- A stand-alone application creating and starting a JMF player shall receive a higher priority for resources than the presentation of any previously existing video and audio.

## 11.12.2 Stored services

The GEM specification supports two forms of stored applications as defined in clause 9.1.9, "Cached applications" and clause 9.6.2, "Stored services".

### 11.12.2.1 Stored application APIs

This is defined in annex AG, "Stored application APIs" of the present document.

### 11.12.2.2 Modified behaviour of GEM 1.0 APIs

The following APIs defined in GEM 1.0 shall have extended semantics to support listing and launching of stored services as defined in clause 9.6.2, "Stored services". The modifications to these APIs are as follows:

- The class javax.tv.service.navigation.ServiceTypeFilter when used with a ServiceType "`org.dvb.application.StoredApplicationServiceType.STORED_APPLICATION_SERVICE`' shall return a `ServiceList` containing all currently installed stored services when passed to SIManager.`filterServices`.

NOTE: Implementations may choose to expose other installed services (e.g. manufacturer resident ones) through this API.

- Instances of `javax.tv.service.Service` returned from such a `ServiceList` shall be instances of `org.dvb.application.StoredApplicationService` with the modified semantics as defined for that interface.

- The method `ServiceContext.select(Service selection)` shall accept instances of `org.dvb.application.StoredApplicationService` and attempt to start the stored application service concerned subject to the GEM security model.

- Attempting to select a stored application service which contains no autostart applications shall fail with a `SelectionFailedEvent` with reason code `CONTENT_NOT_FOUND`.

- The file system APIs shall see a file system comprised of the files defined in the application description file as needing to be stored for this application. Calling new `DSMCCObject(".")` or new `java.io. File(".")` shall instantiate the directory object that refers to the outermost (top level) directory listed in the application description file (see clause 10.8.3, "Application description file") when the application was stored. Other file system behaviour shall be as defined in clause 9.6.2, "Stored services".

- The `org.dvb.dsmcc.DSMCCObject` class shall work for access to files within a stored application. The methods with modified semantics in these circumstances are as follows:

  - `abort()` - shall always throw a `NothingToAbortException`.

  - `asynchronousLoad(AsynchronousLoadingEventListener)` - if the file exists, succeeds immediately (with `SuccessEvent` being generated) otherwise fails with `InvalidPathNameException`.

  - `isObjectKindKnown()` - always returns true.

  - `isStream()` - always returns false.

  - `isStreamEvent()` - always returns false.

  - `loadDirectoryEntry(AsynchronousLoadingEventListener)` - always succeeds immediately with `SuccessEvent` being generated.

  - `prefetch(both signatures)` - always returns false.

  - `setRetrievalMode(int)` - silently ignored.

  - `synchronousLoad()` - if the file exists, succeeds immediately, otherwise fails with `InvalidPathnameException`.

Additionally:

- `ObjectChangeEvents` shall not be generated.
- The `unload()` method shall not be considered as removing stored files from where they are stored.
- Access to files shall not fail for reasons of a service domain not being in an attached state, even though a stored filesystem cannot be represented by a DSMCC Service Domain.
- `File.getCanonicalPath()` and `File.getAbsolutePath()` cannot be relied upon to be return the same strings for the broadcast and stored cases.
- Within a single application instance, the absolute paths returned by `java.io.File.getAbsolutePath()` shall be accepted by constructors for `java.io.File` and sub-classes to reference the same underlying file as the one on which `getAbsolutePath()` was called.

#### 11.12.2.3 Permissions

##### 11.12.2.3.1 FilePermission

A read permission shall be granted for the subtree corresponding to the outermost (top level) directory listed in the application description file and recursively all directories below that.

#### 11.12.2.4 Stored application management API

This is the org.dvb.application.privileged package defined in annex AT, "Application Management API".

## 11.13 Void

## 11.14 Internet Access

### 11.14.1 Internet client control APIs

These are defined in the `org.dvb.internet` package (see annex AH, "Internet client APIs").

Internet clients accessible to GEM applications shall appear in the list of services maintained by the `SIManager` which is returned by calls to the method `SIManager.filterServices`. For each accessible internet client, the corresponding sub-class of `InternetClientService` shall be returned from calls to that method both when passed an instance of `ServiceTypeFilter` constructed with the type `InternetServiceType.INTERNET_CLIENT` and when passed null.

The method `ServiceContext.select(Service selection)` shall accept instances of `InternetClientService` and its sub-classes returned by the mechanism described above as being valid input.

NOTE 1: When a internet client application is selected in the same service context as a GEM application, the mechanism by which that internet client application terminates is implementation dependent. Examples of termination may include the end-user closing the browser (by some mechanism), the end user selecting a broadcast TV service using the navigator or the end user selecting a broadcast TV service by selecting a link to a "dvb:" URL in a WWW page or email. See also clause 9.7, "Lifecycle of internet access applications".

GEM terminals capable of supporting GEM applications and internet client applications simultaneously shall support the creation of a second service context by GEM applications using `ServiceContextFactory.createServiceContext()` in addition to the one used for "normal" GEM applications. This second service context need only support the selection of instances of `InternetClientService`. Selection of other services in this second service context would fail with a `SelectionFailedEvent` with reason code `INSUFFICIENT_RESOURCES`.

NOTE 2: This does not prevent GEM terminals with multiple independent video outputs supporting a service context per video output capable of supporting GEM services however in this case, these service contexts would be created by the GEM navigator.

NOTE 3: Support of multiple service contexts on the same video output each allowing presentation of GEM services is not excluded in the present document but is not fully specified here.

## 11.14.2 Internet applet support

Internet applet support is defined as follows.

### 11.14.2.1 HTML tags

Internet applets shall be supported using the "<applet>" and "<object>" tag with the semantics defined in HTML 4 [69].

### 11.14.2.2 Java Platform

The Java platform for Internet applets shall be PP 1.1 [77].

### 11.14.2.3 Void

### 11.14.2.4 Void

### 11.14.2.5 Security

The present document does not require support for signed applets or for the code signing mechanism and APIs as defined in PP 1.1 [77].

If applets can access Java APIs defined as part of clause 11, "DVB-J platform", apart from those listed in clause 11.14.2, "Internet applet support", they shall have the permissions required to be available to unsigned GEM applications. Of the permissions available to signed GEM applications, they shall have only the following:

- `java.util.PropertyPermission` to `"read"` all properties defined for the `java.lang.system.getProperties()` method.
- `java.net.SocketPermission` to `"connect"` to all ports on the server from which the applet was originally downloaded.

# 11.15 APIs defined in OCAP

This clause is optional in all profiles of GEM. GEM terminal specifications may include all or part of the specification elements of this clause, or may include none at all. If specification elements are included, functional equivalents may be specified where necessary.

## 11.15.1 Introduction (informative)

The present document includes a number of APIs defined in OCAP [5] but not all of them. Many classes in OCAP are wholly or substantially specific to the US cable TV environment and would not be usable elsewhere. The following table summarize the classes included from OCAP and their relation with the present document.

**Table 42: Classes from OCAP in the present document**

| Name of class | Included in present document | Not included in present document | |
|---|---|---|---|
| | | Specific to US cable environment | Other reasons |
| org.ocap.application.AppFilterHandler | Yes | | |
| org.ocap.application.AppFilter | Yes | | |
| org.ocap.application.AppManagerProxy | Partially | | |
| org.ocap.application.AppPattern | Yes | | |
| org.ocap.application.AppSignalHandler | Yes | | |
| org.ocap.application.OcapAppAttributes | Partially | | |
| org.ocap.application.PermissionInformation | Yes | | |
| org.ocap.application.SecurityPolicyHandler | Yes | | |
| org.ocap.event.EventManager | | | Yes |
| org.ocap.event.UserEventAction | | | Yes |
| org.ocap.event.UserEventFilter | | | Yes |
| org.ocap.event.UserEvent | | | Yes |
| org.ocap.hardware.CopyControl | | Yes | |
| org.ocap.hardware.Host | | Yes | |
| org.ocap.hardware.IEEE1394Node | | Yes | |
| org.ocap.hardware.pod.HostParamHandler | | Yes | |
| org.ocap.hardware.pod.PODApplication | | Yes | |
| org.ocap.hardware.pod.POD | | Yes | |
| org.ocap.hardware.PowerModeChangeListener | | | Yes |
| org.ocap.hardware.VideoOutputPort | | Yes | |
| org.ocap.media.AlternativeMediaPresentationEvent | | | Yes |
| org.ocap.media.AlternativeMediaPresentationReason | | | Yes |
| org.ocap.media.ClosedCaptioningAttribute | | Yes | |
| org.ocap.media.ClosedCaptioningControl | | Yes | |
| org.ocap.media.ClosedCaptioningEvent | | Yes | |
| org.ocap.media.ClosedCaptioningListener | | Yes | |
| org.ocap.media.FilterResourceAvailableEvent | | Yes | |
| org.ocap.media.ForcedDisconnectedEvent | | Yes | |
| org.ocap.media.MediaAccessAuthorization | | | Yes |
| org.ocap.media.MediaAccessConditionControl | | | Yes |
| org.ocap.media.MediaAccessHandler | | | Yes |
| org.ocap.media.MediaAccessHandlerRegistrar | | | Yes |
| org.ocap.media.MediaPresentationEvaluationTrigger | | | Yes |
| org.ocap.media.MediaPresentationEvent | | | Yes |
| org.ocap.media.MediaTimer | | | Yes |
| org.ocap.media.MediaTimerListener | | | Yes |
| org.ocap.media.NormalMediaPresentationEvent | | | Yes |
| org.ocap.media.NotPresentedMediaInterface | | | Yes |
| org.ocap.media.VBIFilterEvent | | Yes | |
| org.ocap.media.VBIFilterGroup | | Yes | |
| org.ocap.media.VBIFilter | | Yes | |
| org.ocap.media.VBIFilterListener | | Yes | |
| org.ocap.mpeg.PODExtendedChannel | | Yes | |
| org.ocap.net.OcapLocator | | Yes | |
| org.ocap.net.OCRCInterface | | Yes | |
| org.ocap.OcapSystem | Yes | | |
| org.ocap.resource.ApplicationResourceUsage | | | Yes |
| org.ocap.resource.ResourceContentionHandler | | | Yes |
| org.ocap.resource.ResourceContentionManager | | | Yes |
| org.ocap.resource.ResourceUsage | | | Yes |
| org.ocap.service.AbstractService | Yes | | |
| org.ocap.service.AbstractServiceType | Yes | | |
| org.ocap.service.ServiceContextResourceUsage | | | Yes |
| org.ocap.service.ServiceTypePermission | Yes | | |
| org.ocap.si.Descriptor | | | Yes |
| org.ocap.si.DescriptorTag | | Yes | |
| org.ocap.si.PATProgram | | Yes | |
| org.ocap.si.PMTElementaryStreamInfo | | Yes | |
| org.ocap.si.ProgramAssociationTable | | Yes | |
| org.ocap.si.ProgramAssociationTableManager | | Yes | |

| Name of class | Included in present document | Not included in present document | |
|---|---|---|---|
| | | Specific to US cable environment | Other reasons |
| org.ocap.si.ProgramMapTable | | Yes | |
| org.ocap.si.ProgramMapTableManager | | Yes | |
| org.ocap.si.StreamType | | Yes | |
| org.ocap.si.TableChangeListener | | | Yes |
| org.ocap.si.Table | | | Yes |
| org.ocap.storage.DetachableStorageOption | Yes | | |
| org.ocap.storage.ExtendedFileAccessPermissions | Yes | | |
| org.ocap.storage.LogicalStorageVolume | Yes | | |
| org.ocap.storage.RemovableStorageOption | Yes | | |
| org.ocap.storage.StorageManagerEvent | Yes | | |
| org.ocap.storage.StorageManager | Yes | | |
| org.ocap.storage.StorageManagerListener | Yes | | |
| org.ocap.storage.StorageOption | Yes | | |
| org.ocap.storage.StorageProxy | Yes | | |
| org.ocap.system.EASHandler | | Yes | |
| org.ocap.system.EASModuleRegistrar | | Yes | |
| org.ocap.system.event.DeferredDownloadEvent | Yes | | |
| org.ocap.system.event.ErrorEvent | Yes | | |
| org.ocap.system.event.RebootEvent | Yes | | |
| org.ocap.system.event.ResourceDepletionEvent | Yes | | |
| org.ocap.system.event.SystemEvent | Yes | | |
| org.ocap.system.event.SystemEventListener | Yes | | |
| org.ocap.system.event.SystemEventManager | Yes | | |
| org.ocap.system.MonitorAppPermission | Partially | | |
| org.ocap.system.RegisteredApiManager | | | Yes |
| org.ocap.system.SystemModuleHandler | | Yes | |
| org.ocap.system.SystemModule | | Yes | |
| org.ocap.system.SystemModuleRegistrar | | Yes | |
| org.ocap.test.OCAPTest | | Yes | |
| org.ocap.ui.event.OCRcEvent | | | Yes |
| org.ocap.ui.HSceneBinding | | | Yes |
| org.ocap.ui.HSceneChangeRequestHandler | | | Yes |
| org.ocap.ui.HSceneManager | | | Yes |

## 11.15.2 OCAP Annex G - the org.ocap.application package

This package shall be supported with the following modifications.

- For OcapAppAttributes, the following language shall not apply and the behaviour specified for the methods in GEM shall be supported.

  For this version of the OCAP profile, the getProfiles() method always returns ocap.profile.basic_profile

- For OcapAppAttributes, the following language shall only apply to applications signalled in an XAIT or added to the AppsDatabase by registerUnboundApp. It shall not apply to applications signalled in an AIT.

  org.dvb.application.AppsDatabase must return an instance of OcapAppAttributes by the getAppAttributes methods.

- The method AppManagerProxy.setApplicationPriority shall have no effect.

- The method AppManagerProxy.registerUnboundApp shall accept as valid input XAITs formatted in both the MPEG-2 section and table format as specified in OCAP and the XML format as specified in clause 10.16.2, "XAIT" of MHP [1].

## 11.15.3 OCAP Annex P - the org.ocap.service package

This package shall be supported except for ServiceContextResourceUsage.

### 11.15.4 OCAP Annex Q - the org.ocap.system package

From this package, MonitorAppPermission shall be supported. The following permission names shall be supported.

- "registrar".
- "servicemanager".
- "service".
- "handler.appFilter".
- "security".
- "reboot".
- "systemevent".

### 11.15.5 OCAP annex O - the org.ocap package

This package shall be supported. The requirements relating to initialising the APIs for conformance testing do not apply since those APIs are not required by the present document.

### 11.15.6 OCAP annex U - the org.ocap system.event

The org.ocap.system.event package shall be supported.

# 12 Security

## 12.1 Introduction

This clause covers the following areas of security:

- Authentication of applications.
- Security policies for applications.
- Authentication and privacy of the return channel communications.
- Certificate management.

### 12.1.1 Overview of the security framework for applications

The security framework enables a receiver to authenticate the source of application code or other files. In the case of application code files, the authentication advises the receiver what access rights should be granted to an application for sensitive resources, see clause 12.6, "Security policy for applications" for more detail.

The system uses 3 different authentication messages:

- Cryptographic hash codes:
  - This provides a summary of a quantity of data - typically a subset of the total set of data under consideration.
- Signatures:
  - These deliver a master hash code (computed over all of the appropriate data) that has been "signed" by an authorising organization. The signing process securely associates the master hash code with the signatory. The hash code process shows that the data has not been tampered with since it was signed by the signatory.
- Certificates:
  - These provide a "chain of trust" from the authorising organization up to some trusted third party (the root certificate authority) that is well known to the receiver.

The messages are delivered within files of the file system so this authentication scheme is applicable to any hierarchical file system whether operating over the broadcast or return channels.

Applications that are eligible to be trusted shall be identified with an application_id from range of values allocated for signed applications (see clause 10, table 13, "Value ranges for application_id"). Applications that are not eligible to be trusted shall be identified with an application_id from the unsigned applications range. For an application with an application_id from the signed application range that requires signing to establish trust, if it is not signed it is considered to have failed authentication. An application with an application_id from the unsigned application range is treated as not eligible to be trusted even if the files might be transmitted with signatures.

### 12.1.2 Overview of return channel security

In this version of the specification general purpose protocols and a standard cryptographic suite derived from internet standards are used.

### 12.1.3 Establishing trusted applications

One mechanism by which an application may be trusted, and thus request additional permissions, is for that application to be signed. Terminal specifications based on GEM may introduce additional mechanisms for establishing that an application is trusted. These mechanisms may involve some form of codesigning.

Any security framework, whether it involves codesigning or not, shall:

- Require that trusted applications be identified with an `application_id` from the signed applications range, as described in clause 12.1.1.
- Refuse to grant permissions outside of the set granted to unsigned applications, unless those permissions are explicitly requested in the signalling.
- Must be completely specified in the GEM terminal specification.

## 12.2 Authentication of applications

For profiles and/or targets that do not require compliance with the functional equivalent named "application authentication" as defined in clause 15.6, "Functional equivalents", GEM terminal specifications may define a functional equivalent for authentication of applications. This functional equivalent shall provide a mechanism to establish that code is sufficiently trusted to enable granting any of the permissions which a GEM application may request. Further, there shall be a mechanism to sign individual files or sets of files with a certificate that may be different from any that is used to determine the trust level of the application.

NOTE: Data files that are signed may have their certificates retrieved by applications using the `getSigners` methods described in clause P.2.1, "DSMCCObject".

### 12.2.1 Overview of authentication messages

Three different message types are used: "Hash codes", "Signatures" and "Certificates". Each message is placed in a file. The placement of the files depends on their function and is specified under the appropriate headings under clause 12.4, "Detail of application authentication messages".

#### 12.2.1.1 Hash codes

The present document describes application of hash codes to the following types of information:

- Files.
- Directories.

The hash computation considers the content and attributes of the objects rather than transport specific information. As a result, the authentication is independent of the underlying transport protocol.

In the case of a directory the hash value depends on the hash values of the objects bound to it, and so provides a hash of all of the objects to be authenticated in the "tree" below it.

#### 12.2.1.2 Signatures

The data authenticated is a hierarchical file system (for example DSM-CC OC). The root of an authenticated "tree" carries one or more signatures. This allows one or more organizations to sign a set of information.

The root of the authenticated "tree" can be the root directory of the file system or the "top" directory of a "subtree".

The signature:

- References a certificate containing the public key required to decode the signature.
- Identifies the hash algorithm used.
- And the value of the signature.

### 12.2.1.3 Certificates

The certificate provides a public key that can be used to decode a hash code contained in a signature and so enable a tree/subtree to be verified. The certificate itself is signed by a higher certification authority.

To correctly authenticate a subtree there must be a valid "chain" of certificates from the signature to a root certificate as is illustrated in figure 14.

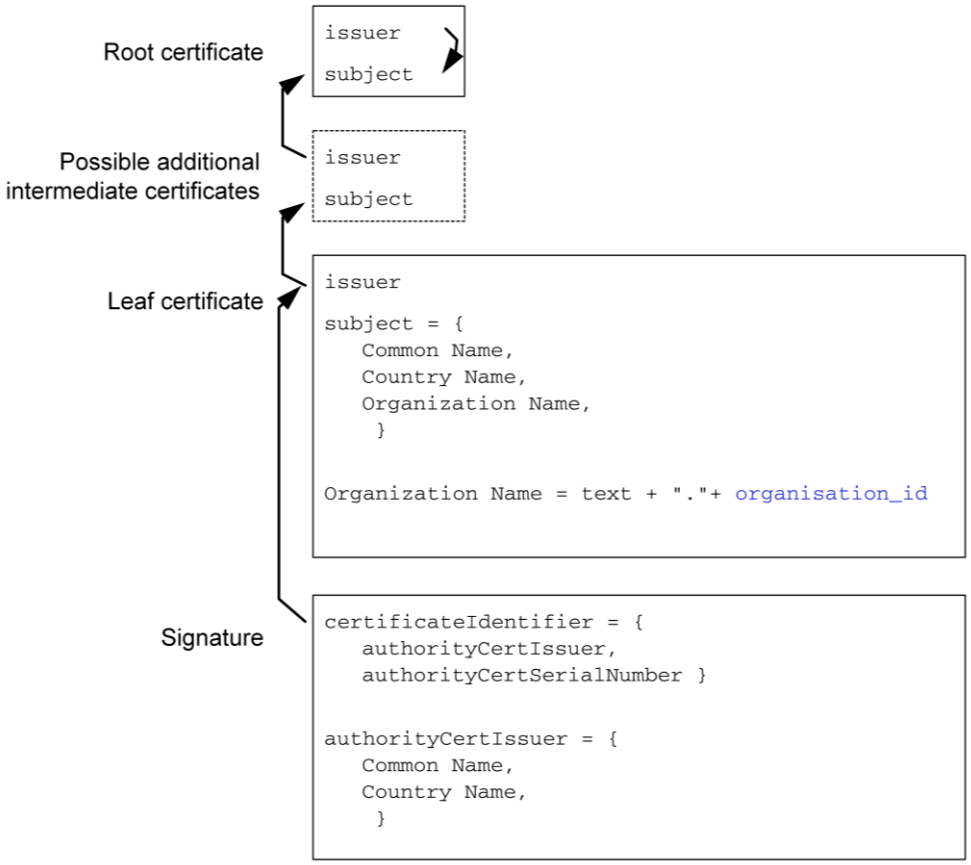



**Figure 14: Certificate chain illustrated**

### 12.2.1.4 Authentication of hierarchical file systems

The solution here is based on authentication of a hierarchical structure of objects. Hashcodes are computed systematically and accumulatively across some or all of the objects in the hierarchy. A signature at the top of the hierarchy identifies the source of the objects.

The framework provides a flexible and non-time consuming method enabling the authentication of subtrees of a file system with a single signature. Since checking signature is far more time-consuming than checking hashcode values, this mechanism is more efficient than signing each object of a subtree.

Further, only the objects that are loaded need real-time hashcode checking.

This mechanism does NOT mandate that the whole subtree is authenticated.

NOTE: The broadcaster can choose which files in the file system are authenticated. For example code files might be authenticated and asset files might be left without authentication.

Finally, this framework embraces key distribution by specifying a certificate mechanism, the aim of this is to certify that the key used to compute the signature is valid and used by a certified service provider.

# 12.3 Message transport

The authentication messages are transported in files.

In no cases shall a service transfer be required to access the file content. In the case that the file system is an object carousel this means that the IOR for the authentication message files shall always use a BIOP profile body and never a Lite options profile body.

# 12.4 Detail of application authentication messages

NOTE 1: In the case where the MHP application authentication mechanism is used, the exact file names, locations and syntaxes described in this clause is supported. This includes the requirement in clause 12.4.3.1 that the last certificate in a CertificateFile be the root certificate.

NOTE 2: As specified in clause 12.2, "Authentication of applications" of the present document, an application authentication mechanism other than MHP's can be used to determine if an application is signed.

Three data structures are defined for communicating authentication information:

- HashFile.
- SignatureFile.
- CertificateFile.

These are placed in files in the file system. The location of the file depends on its function.

## 12.4.1 HashFile

### 12.4.1.1 Description

The HashFile lists all of the elements of the current directory except itself, and possibly except signature files as described in the following. On transport protocols supporting the listing of files in a directory (e.g. Object Carousel), signature files shall either all be listed with a digest type of non-authenticated or shall all be omitted. On transport protocols not supporting the listing of files in a directory (e.g. HTTP), signature files shall be listed and shall have a digest_type of non-authenticated.Those elements to be authenticated are associated with hashcodes. The syntax of the HashFile is shown in table 43.

**Table 43: Syntax of the Hashfile**

| Syntax | Num. Bits | Format |
|---|---|---|
| Hashfile () { | | |
| digest_count | 16 | uimsbf |
| for( i=0; i<digest_count; i++ ) { | | |
| digest_type | 8 | uimsbf |
| name_count | 16 | uimsbf |
| for( j=0; j<name_count; j++ ) { | | |
| name_length | 8 | uimsbf |
| for( k=0; k<name_length; k++ ) { | | |
| name_byte | 8 | bslbf |
| } | | |
| } | | |
| for( j=0; j<digest_length; j++ ) { | | |
| digest_byte | 8 | bslbf |
| } | | |
| } | | |
| } | | |
| Other data may follow but can be ignored by implementations conforming to this profile. | | |

digest_count: This 16 bit value identifies the number of digest values in this hash file.

digest_type: This 8 bit value identifies the digest computation rules and the digest algorithm, if any, used for the associated objects. Table 44 lists the allowed values for this field. The digest computation rules are defined in clause 12.4.1.3, "Digest value computation rules".

**Table 44: Values of digest_type**

| Value | Digest len. | Aalgorithm |
|---|---|---|
| 0 | 0 | Non authenticated |
| 1 | 16 | Digest computation rules without prefix and with MD-5 as defined in RFC 1321 [44] |
| 2 | 20 | Digest computation rules without prefix and with SHA-1 as defined in FIPS-180-1 [57] |
| 3 | 20 | Digest computation rules with prefix and with SHA-1 as defined in FIPS-180-1 [57] |
| Other values | | Reserved for future use |

name_count: This 16 bit value identifies the number of object names associated with the digest value. The value of this field shall be greater than zero.

name_length: This 8 bit value identifies the number of bytes in the object name.

name_byte: This 8 bit value holds one byte of the object name.

Each name shall be the name of an object in the directory that contains the HashFile. So, file names are the names of files in the directory and directory names are the names of direct sub-directories of the directory. No path information shall be included in the name.

The names carried by this field are binary identical to the payload part of names in the file system. So, any name matching process can be binary and ignorant of character encoding, letter case etc. Also, terminating null characters are not considered to be part of the file name.

digest_length: This integer value gives the number of bytes in each digest value. It depends upon the digest type as tabulated in table 44. GEM terminals shall support all digest algorithms.

NOTE: Non-authenticated objects have a zero length digest.

digest_byte: This 8 bit value holds one byte of the digest value. See clause 12.4.1.3, "Digest value computation rules".

### 12.4.1.2 HashFile location and naming conventions

An application comprises files containing data that can be spread across various directories and is contained within a subtree of the file hierarchy. A HashFile will be put in each directory containing objects that need to be authenticated.

The name of the HashFile shall be:

```
"dvb.hashfile"
```

There shall only be one instance of HashFile per directory that contains authenticated resources.

See clause 12.7, "Example of creating an application that can be authenticated".

### 12.4.1.3 Digest value computation rules

The digest value is computed over the objects named in the HashFile in the order listed in the HashFile. The length of the list of objects associated with each digest value may be one or more.

Each list of objects may contain an arbitrary mix of different object types (e.g. a mixture of file and directory names). The digest value is computed by first initialising the digest algorithm in an algorithm specific way and then applying the relevant data for each object to the algorithm in order. The relevant data for each object depends on its type (File or Directory) and on the value of digest_type and is specified in table 45.

**Table 45: Data required for digest value computation**

| Object type | Digest type | entry_type | Relevant data |
|---|---|---|---|
| File | 1 or 2 | not applicable | The entire content of the file. |
| Directory | | | The content of the HashFile of the named directory. |
| File | 3 | 1 | A prefix concatenated with the entire content of the file. The prefix is made of the entry_type encoded as a 32 bit uimsbf and concatenated with the file length in bytes encoded as a 32 bit uimsbf. |
| Directory | | 0 | A prefix concatenated with the content of the HashFile of the named directory. The prefix is made of the entry_type encoded as a 32 bit uimsbf and concatenated with the file length in bytes encoded as a 32 bit uimsbf. |
| NOTE: All other values of entry type are reserved for future use. | | | |

#### 12.4.1.3.1 Example

Consider two files `file1` and `file2` and one directory `dir1`. Using digest type 3, the digest value of `file1` + `file2` + `dir1` is equal to:

```
SHA-1 ( (uimsbf 32) 1 + (uimsbf 32) FileLength ( file1) + contents of file1
+ (uimsbf 32) 1 + (uimsbf 32) FileLength(file2) + contents of file2
+(uimsbf 32) 0 +(uimsbf 32) FileLength(HashFile(dir1))
+ contents of HashFile(dir1) ).
```

### 12.4.1.4 Warning concerning grouping of objects under a single digest (informative)

Broadcasters should be made aware that during the construction of the object carousel, grouping objects under one digest can significantly and adversely affect the performance and memory requirements of an application on a terminal. For example, if three objects are grouped under the same digest, implementations must load all three objects even if only one is needed.

GEM in general recommends that while setting up a signed application for broadcast, only one digest be associated with each object requiring authentication. Grouping of multiple files under one hash value should only be employed if those files must always be loaded together and simultaneously. Additionally, the rate of change of the contents of an object should be carefully considered because grouping a rapidly changing object with more slowly changing objects for the purposes of hashing will negatively impact performance. When attempting such grouping, the resource limits of the GEM terminal should always be carefully considered.

### 12.4.1.5 Special authentication rules

a) For objects which are directories, if the digest_type is non-zero there shall be a HashFile in the sub-directory listed. If the HashFile is absent then the authentication fails.

b) Each HashFile shall provide a complete list of all the objects named in the directory except itself and any possible signature files mentioning each name exactly one time. The behaviour varies depending on the ability of the transport protocol to support directory listing.

- On transport protocols supporting the listing of files in a directory (e.g. Object Carousel), the authentication shall fail if the set of objects listed in the HashFile is different to the set of objects in the directory.

- On transport protocols that don't support the listing of files in a directory (e.g. HTTP), the set of names in the HashFile acts as a filter for the set of objects that can be accessed. Attempts to access a file neither named in the HashFile nor the HashFile itself, nor the signature files shall fail (that is behave as if the file is not available). The file system for loading applications from the interaction channel via HTTP is a case of such a file system, see clause 6.4.1.4, "Directory listing in this file system".

These rules apply regardless of the value of digest_type associated with the object.

c) If the digest_count is equal to 0, the file HashFile shall be ignored. Every entry in the directory will be non authenticated.

d) If a name_length is equal to zero (or if a name is not found in the directory containing the HashFile), all the names associated with the current digest value shall be considered as incorrectly authenticated. It means the authentication checking will fail for these entries.

e) If the object is a Stream or StreamEvent then the digest_type shall be zero.

f) Objects associated with a digest_type that the receiver does not support shall be treated by that receiver as if the value of digest_type was zero (not authenticated).

g) There is no requirement for certificates files to have a digest_type other than zero (not authenticated). However, if they do have a non-zero digest_type they don't receive exceptional treatment.

NOTE: There is no security benefit in including a certificate file with a digest_type other than zero.

h) When a permission request file exists, its entry in the HashFile cannot have a digest type of 0. GEM terminals shall ignore a permission request file with a digest type of 0.

## 12.4.2 SignatureFile

### 12.4.2.1 Description

The SignatureFile is a File containing one digital signature. It contains the following ASN.1 DER structure:

```
Signature  ::=  SEQUENCE  {
    certificateIdentifier                    AuthorityKeyIdentifier,
    hashSignatureAlgorithm                   OBJECT IDENTIFIER,
    signatureValue                           BIT STRING  }
```

certificateIdentifier : As defined in the ITU-T Recommendation X.509 [52] extension for the AuthorityKeyIdentifier field. It identifies the certificate that carries the certified public key that is used to check the signature.

```
AuthorityKeyIdentifier ::= SEQUENCE {
    keyIdentifier                            [0] KeyIdentifier OPTIONAL,
    authorityCertIssuer                  [1] GeneralNames OPTIONAL,
    authorityCertSerialNumber                [2] CertificateSerialNumber OPTIONAL }
```

Implementations are not required to use the possibly present `keyIdentifier` element of the `AuthorityKeyIdentifier`. The `AuthorityKeyIdentifier` structure shall contain both the `authorityCertIssuer` and `authorityCertSerialNumber` elements.

The `authorityCertIssuer` shall contain the field "`directoryName`", this field shall be equal to the `issuerName` of the certificate that carries the public key used to check the signature.

hashSignatureAlgorithm: this field identifies the hash algorithm that is used.

NOTE: The encryption algorithm used to compute the signature is already described in the SubjectKeyInfo field of the certificate that certifies this key, and thus that only the identification of the hash algorithm is needed. The supported algorithms are MD5 and SHA-1.

```
md5 OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) US(840) rsadsi(113549)
    digestAlgorithm(2) 5 }

sha-1 OBJECT IDENTIFIER ::=
    { iso(1) identified-organization(3) oiw(14) secsig(3)
    algorithm(2) 26 }
```

signatureValue: See clause 12.11.1.3, "signatureValue".

### 12.4.2.2 SignatureFile location and naming conventions

The SignatureFile is located in the root directory of the subtree that it signs. There can be several SignatureFiles, as there can be several entities that sign the structure. See clause 12.4.4, "Integration".

By convention, the name of a SignatureFile is:

"dvb.signaturefile."<x>

where the <x> is a textual representation of an integer decimal number without leading zeroes. The range of values represented in any single directory shall start with 1 and increment in steps of 1. The first unused integer value in the ascending sequence indicates the end of the range.

The purpose of this x is to allow the hash file of an authenticated subtree to be signed by more than one entity. It is equal to the x value in the file name of the corresponding certificate file. See clauses 12.4.3.5, "CertificateFile location and naming conventions" and 12.4.4, "Integration".

#### 12.4.2.3 Supported algorithms

Signing data is a two-step process:

a) First a hash is computed over the data.

b) The resulting hashvalue is then encrypted using an encryption algorithm.

As indicated in clause 12.4.1, "HashFile," GEM defines two possible hash algorithms: MD5 and SHA-1.

The encryption algorithm used to compute the signature is indicated in the certificate that carries this key.

#### 12.4.2.4 Signature computation rules.

The hash is computed over the content of the HashFile contained in this root directory.

NOTE: This is the same principle as for the classical hash computation described in clause 12.4.1.3, "Digest value computation rules".

#### 12.4.2.5 Authentication rules

See signatureValue.

### 12.4.3 CertificateFile

#### 12.4.3.1 Description

The CertificateFile contains all of the certificates in the certificate chain up to, and including, the root certificate. The leaf certificate is placed first in the file. The last certificate in the file is the root certificate. The root certificate is included in this file only for consistency. How the GEM terminal determines its policy related to this root CA is implementation dependent. The encoding of the certificate is defined in ITU-T Recommendation X.509 [52]. Below is defined the profile of ITU-T Recommendation X.509 [52] for use in authenticating GEM applications. This profile is based on RFC 2459 [55].

The syntax of the CertificateFile is shown in table 46.

**Table 46: Syntax of the CertificateFile**

| Syntax | Num. Bits | Format |
|---|---|---|
| Certificatefile () { | | |
| certificate_count | 16 | uimsbf |
| for( i=0; i<certificate_count; i++ ) { | | |
| certificate_length | 24 | uimsbf |
| certificate() | | |
| } | | |
| } | | |

certificate_count: This 16-bit integer carries the number of certificates in the certificate file.

certificate_length: This 24-bit integer specifies the number of bytes in the certificate.

certificate(): This field carries a single g data structure as defined by ITU-T Recommendation X.509 [52]. See clause 12.11.1, "Main part of the certificate".

#### 12.4.3.2 ASN.1 encoding

The basic specification of the ASN.1 DER encoding used in RFC 2459 [55] is given in ASN.1 [54]. However, RFC 2459 [55] defines some extensions which are required to implement GEM.

#### 12.4.3.3 Supported algorithms

There are various algorithm identifiers in the certificate structure. The OID of the algorithm used in the SubjectPublicKeyInfo structure shall be RSA.

The values for AlgorithmIdentifier used both in the certificate structure and in the TBSCertificate structure that are supported in GEM are listed under clause 12.5.1, "signatureAlgorithm".

#### 12.4.3.4 Name matching

The only allowed encoding of attributes of distinguished names shall be UTF8String.

NOTE: The use of this encoding allows name matching to be a binary comparison.

#### 12.4.3.5 CertificateFile location and naming conventions

As described in clause 12.2.1.3, "Certificates", a key can be authenticated through a "certificate chain".

A certificate chain is a hierarchy of certificates that enable the implementation to verify the validity of the key used to check a signature. However, for consistency, the file shall carry all of the certificate chain up to, and including, the root certificate.

The certificate file that leads to the public key of a signature shall be placed in the same directory as that signature file.

The name of a CertificateFile is:

```
'dvb.certificates.'<x>
```

where the <x> corresponds to the x value in the file name of the signature file verified by this certificate. Hence in the root directory of an authenticated subtree there shall be one certificate file for each signature file. See clauses 12.4.2.2, "SignatureFile location and naming conventions" and 12.4.4, "Integration".

#### 12.4.3.6 Authentication rules

Certificates are considered to be the same if they have bitwise identical contents.

Where a GEM application is authenticated by multiple signature file / certificate file pairs, the GEM terminal shall check all the signature file / certificate file pairs before deciding that a file fails authentication. It is dependent on receiver policy whether additional signature file / certificate file pairs are checked once one satisfactory pair has been found.

### 12.4.4 Integration

Logically a file is authenticated as follows:

a) Confirm that the file is listed in the hash file located in the same directory as the file to be authenticated.

b) Verify that the file contents and the corresponding digest value are consistent.

c) Recursively ascend the directory hierarchy checking that the hash file in each directory is authenticated by the hash file in its parent directory until a directory is found that contains one or more signature files.

   Such a directory is termed the root directory of an authenticated subtree.

NOTE 1: This means that there is no requirement to progress above the root of the authenticated subtree to examine further hash files. If such a directory has a digest type other than "Non authenticated" in its parent directory, this is only significant when verifying the hash file of the parent directory.

NOTE 2: The presence of more than one signature file enables more than one set of organizations to authenticate a subtree.

NOTE 3: When authenticating a signed application, a GEM terminal can select any one of these certificates to use to authenticate all subsequent classes or files loaded.

Where a GEM terminal trusts more than one of these certificates, it should attempt to select the most trusted of these. Apart from this, the mechanism used to make this selection is implementation dependent.

d) For a signature file locate the corresponding certificate file (where the x portion of the signature file's file name identifies the certificate file to be used).

e) If the root certificate in the corresponding certificate file is unknown to the GEM receiver, return to step (d) and repeat for the other signature files.

f) Use the corresponding certificate file to verify that the signature correctly signs the hash file.

g) Verify that the attributes of the Subject field include content that matches the signalling information as per the requirements of clause 12.5.6.

h) Follow the certificate chain contained within the certificate file verifying each link in the chain until the link to the root certificate is found.

i) If the identified root certificate and all the intermediate certificates leading to it are "satisfactory", accept the files as being authenticated.

"Satisfactory" depends on the policies implemented in the receiver and other constraints expressed in the present document. In particular the requirements in clause 11.2.3, "Class Loading" for using the "same" leaf certificate to authenticate DVB-J class files shall be observed.

j) If the identified root certificate or any of the intermediate certificates leading to it are not "satisfactory", return to step (d) and repeat for the other signature files.

k) Dependent on receiver policy return to step (d) and repeat for other signature files.

NOTE 4: The above is a logical description of the process and does not constrain implementations to perform these steps in this exact order. E.g. hash files may be verified when descending an directory hierarchy rather than ascending one.

A file system may contain several independent authenticated subtrees, each tree with its own subtree root directory.

NOTE 5: For the method `org.dvb.dsmcc.DSMCCObject.getSigners`, the semantics defined in that method take precedence over these rules where there are conflicts.

# 12.5 Profile of X.509 certificates for authentication of applications

This clause identifies how ITU-T Recommendation X.509 [52] is profiled when used for authentication of broadcast GEM applications. This profile is a variation (in general a sub-set) of the internet profile defined in RFC 2459 [55]. This clause identifies the differences from the profile in RFC 2459 [55]. Clause 12.11, "The internet profile of X.509" summarizes the profile in RFC 2459 [55].

## 12.5.1 signatureAlgorithm

GEM supports 2 signature algorithms: MD5 with RSA and SHA-1 with RSA.

### 12.5.1.1 MD5 with RSA

The signature algorithm with MD5 and the RSA encryption algorithm is defined in RFC 2313 [53]. As defined in RFC 2313 [53], the ASN.1 OID used to identify this signature algorithm is:

```
md5WithRSAEncryption OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 4 }
```

#### 12.5.1.2 SHA-1 with RSA

The signature algorithm with SHA-1 and the RSA encryption algorithm is implemented using the padding and encoding conventions described in RFC 2313 [53]. The message digest is computed using the SHA-1 hash algorithm. The ASN.1 object identifier used to identify this signature algorithm is:

```
sha-1WithRSAEncryption OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 5 }
```

#### 12.5.1.3 parameters

For both of the two supported algorithms the parameters component shall be the ASN.1 type NULL.

### 12.5.2 signatureValue

The RSA signature generation process and the encoding of the result is described in detail in RFC 2313 [53].

### 12.5.3 version

The version field of the certificate shall signal v3. All implementations shall support the v3 extensions as required by clause 12.5.9, "Extensions".

### 12.5.4 issuer

#### 12.5.4.1 minimum requirement

For GEM at least a Common Name attribute shall be provided. The text value of the attribute shall be non-empty. It shall be suitable for direct presentation to the user.

#### 12.5.4.2 certificate authority responsibility

The senior certificate authority who signs a certificate shall oversee the attribute information to ensure that the information is suitable.

### 12.5.5 validity

The only allowed format for encoding time in the validity field is GeneralizedTime.

### 12.5.6 subject

The subject field is a "distinguished name". The following requirements are specified by GEM:

- The only allowed encoding attributes of the subject is `UTF8String` (see clause 12.4.3.4, "Name matching").
- The minimum set of attributes that shall be present in the subject are:
  - `commonName`
  - `countryName`
- If the certificate is a "leaf certificate" (see figure 14, "Certificate chain illustrated") then the subject shall also contain an `organizationName`.
- When encoded the `organizationName` carries organization specific text post fixed by the `organisation_id` of the authenticated files. This integer value is represented as a fixed length 8 character hexadecimal string (with leading zeros where required). So the `organizationName` takes the form:
  - `text + "." + organisation_id`

Except for the presence of leading zeros, this is the same as clause 14.5, "Text encoding of application identifiers".

This field reproduces the `organisation_id` value from the application's application identifier, see clause 10.4.3, "Content of the Application Description". Before the application starts executing, the value of the `organisation_id` of at least one of the leaf certificates used in the authentication of the application's root subdirectory must match that in the application identifier. If this is not true, then the application is considered to have failed authentication. Applications which fail authentication at this point shall not be started on GEM terminals.

If the certificate is a "leaf certificate" then the subject may also contain up to four `organizationalUnitName` entries.

When encoded each `organizationalUnitName` entry carries a comma separated list of up to five original network identifiers each entry of which is a fixed length 5-character hexadecimal string encoded as follows:

- The first character is a network type identifier encoded as a single hexadecimal digit with one or more bits set and with the bits having the following meanings:

| | |
|---|---|
| xxx1 | terrestrial network |
| xx1x | cable network |
| x1xx | satellite network |
| 1xxx | internet |

- The next four characters are a hexadecimal representation, including leading zeros where necessary, of the original_network_id as defined in EN 300 468 [14].

Before the application starts executing, the network type and the original_network_id of the service carrying the application shall be matched against the entries in the comma separated lists in the `organizationalUnitName` entries in each of the leaf certificates used in the authentication of the application's root directory. The original_network_id of the service, shall be found by first looking in the transport_stream_loop of the NIT for the transport stream carrying the service and only if the service is not listed in that loop, looking in the SDT.

Certificates where none of the entries match shall not be used to authenticate applications. If no match is found in any of these leaf certificates then the application is considered to have failed authentication. Applications that fail authentication at this point shall not be started on GEM terminals.

NOTE: Where a particular original_network_id is uniquely used in a particular market, this feature enables certificate authorities to apply market specific policies when deciding which organisations are entitled to have certificates issued to them for that market. The extent to which this provides real additional real security depends on factors outside the scope of the present document.

## 12.5.7 SubjectPublic Key Info

GEM supports a single public key algorithm (RSA) for the subject public key.

The key lengths that implementation are required to support are addressed in annex G, "Minimum platform capabilities".

### 12.5.7.1 rsaEncryption

The OID rsaEncryption identifies RSA public keys:

```
rsaEncryption OBJECT IDENTIFIER ::= { pkcs-1 1}

pkcs-1 OBJECT IDENTIFIER  ::=  { iso(1) member-body(2) us(840)
    rsadsi(113549) pkcs(1) 1 }
```

The parameters field shall have ASN.1 type NULL.

### 12.5.7.2 subjectPublicKey

The RS subjectPublicKey BIT STRING shall be encoded using the ASN.1 type RSAPublicKey:

```
RSAPublicKey ::= SEQUENCE {
    modulus                                 INTEGER, -- n
    publicExponent                          INTEGER -- e -- }
```

The semantics of the modulus (n) and the public exponent (e) are defined in RFC 2313 [53].

## 12.5.8 Unique Identifiers

X.509 defines the `issuerUniqueID` and `subjectUniqueID` extensions.

CAs conforming to this profile shall not generate certificates with unique identifiers.

GEM terminals conforming to this profile are not required to be capable of parsing unique identifiers and making comparisons.

## 12.5.9 Extensions

The following restrictions and semantics are placed on the use of certificate extensions when used to authenticate applications.

**Table 47: Profile for standard certificate extensions**

| Extension | In broadcasts | In implementations | Semantic |
|---|---|---|---|
| Authority key identifier | Opt. | Opt. | Shall not be marked critical |
| Subject key identifier | Opt. | Opt. | Shall not be marked critical |
| Key usage | Mand. | Mand. | If the keyUsage extension is marked critical, for the leaf certificate the bit digitalSignature shall be set, for other certificates the bit keyCertSign shall be set. If these bits are not set then the certificate shall be ignored by the implementation. |
| Private key usage period | Opt. | Opt. | Shall not be marked critical |
| Certificate policies | Opt. | Opt. | Shall not be marked critical |
| Policy mappings | Opt. | Opt. | Shall not be marked critical |
| Subject Alternative Name | Mand. | Opt. | Shall not be marked critical<br>The subject name unambiguously identifies the subject.<br>It is recommended that DVB GEM implementations can read rfc822Name (email address) |
| Issuer Alternative Name | Mand. | Opt. | Shall not be marked critical<br>The issuer name unambiguously identifies the issuer.<br>It is recommended that DVB GEM implementations can read rfc822Name (email address). |
| Subject Directory attributes | Opt. | Opt. | Shall not be marked critical |
| Basic Constraints | Opt. | Mand. | May be marked critical |
| Name Constraints | Opt. | Mand. | May be marked critical.<br>DVB GEM decoders shall be able to recognize name constraints when GeneralName are either directoryName or rfc822 names. |
| Policy Constraints | Opt. | Opt. | Shall not be marked critical |
| Extended key usage field | Opt. | Opt. | Shall not be marked critical |
| CRL Distribution points | Opt. | Opt. | Shall not be marked critical |

**Table 48: Key for table 47**

| Keyword | When applied to | |
|---|---|---|
| | broadcasts | receivers |
| Mandatory | Certificates shall include these extensions. | Receivers shall observe the semantic for this information when provided. |
| Optional | Certificates may or may be not include these extensions.<br>The GEM specification does not define the use of these extensions and hence broadcasters should not use them. | Receivers can ignore these extensions if present. |
| Critical/not-critical | Broadcasters should not mark the receiver optional extensions as critical. | Certificates containing unrecognized critical extensions shall be considered as invalid.<br>Receivers should recognize all the extensions that can be critical. |

# 12.6 Security policy for applications

## 12.6.1 General principles

As described in clause 12.1.3, mechanisms other than MHP codesigning may be used to determine if additional permissions should be granted to applications. As a consequence, in clause 12.4 the term "signed application" is to be interpreted as meaning an application that has been packaged in such a way that it is eligible for being granted additional permissions, either via the MHP signing mechanisms or through other mechanisms. "Unsigned application" is to be interpreted as meaning an application that has not.

An alternative permission request mechanism may be used in order to request additional permissions over and above the ones defined in the present document. Any such alternative permission request mechanism shall not conflict with the MHP one and shall be completely defined in the GEM terminal specification with the following constraints:

- the permission requests shall be contained in a file encoded in XML format;
- the alternative permission request file shall be located in the directory that contains the initial file of the application;
- the formal public identifier of the alternative DTD shall identify the organization that specifies the alternative mechanism;
- the name of an alternative permission request file (PRF) shall be prefixed with a name identifying the GEM terminal specification, e.g. ocap.<application_name>.perm;
- the alternative DTD shall include at least all of the same XML elements and attributes as those defined in the DVB PRF.

Any additional permissions that are needed beyond those defined in GEM shall have their XML element names, global attribute names to be used with any element type, and local attribute names to be used with existing GEM defined element types prefixed by a string identifying the GEM terminal specification.

NOTE 1: Any local attributes on newly defined prefixed (qualified) XML element types that are used locally to those element types and are not intended to be used as global attributes should not be prefixed in accordance with the Per-Element-Type Partition defined by XML Namespaces [28], annex A.2, "XML Namespace Partitions".

This clause specifies the resource access policy for the downloaded applications. The resource access policy depends on two factors:

- The access rights requested by the broadcaster through the signalling.
- The access rights granted by the user.

The ultimate access rights that are granted to the applications are the intersection of the access rights requested by the broadcaster and the access rights granted by the user.

Unsigned applications have limited access to platform resources.

Unless specified elsewhere in the present document, signed applications have the same access rights as unsigned applications. An application broadcaster can request additional permissions to access specific resources by providing a signed "Permission Request File" along with the application. The syntax and semantics of the Permission Request File are defined in the following clauses. The permission request file may also contain a credential that indicates that a persistent file owned by another organization may be accessed. If the "Permission Request File" is not correctly authenticated the application is not granted any additional permissions but is not prevented from starting for this reason.

The way the user grants rights to the downloaded applications is implementation dependent and is not addressed by GEM.

For DVB-J applications, accessing a resource consists of method calls. Each method call that results in accessing the resource shall throw a security exception as defined in clause 11.10, "Java permissions" and the specifications of the java APIs concerned. For each resource subject to access restriction, the application can test whether it has been granted permissions to access it by using the corresponding java Permission class (see clause 11.8, "Security").

For a DVB-J application to be correctly authenticated, all the class files that the application consists of need to be signed, the signatures need to verify (see clause 12.4.4, "Integration") and the application_id needs to be from within the range allocated to signed applications (see table 13, "Value ranges for application_id"). If, during the loading or execution of the application the GEM detects a signed file containing a class that failed to pass the authentication process (e.g. because its actual hash value does not match the expected hash value), then the class shall be considered as not available.

When an application requests to retrieve data from a signed file, and the GEM terminal attempts to authenticate that file, and the authentication process fails, then the API concerned shall fail in a manner consistent with the defined behaviour of that API when the file exists but has no content in it.

NOTE 2: In order to be efficient, if a directory D contains objects that are likely to frequently change, it is advised to put a signature file in this directory D and to mark the directory D as non authenticated in the hashfile located in the parent directory of D. By doing so, it will limit the propagation of modifications to just one directory.

The authentication of a file is evaluated each time that the file is loaded from a transport connection. File version information in the transport system cannot be assumed to be secure.

Applications authors should be aware that deciding whether to grant a permission or not may, depending on the implementation, involve prompting and asking the end user. The latest point in time when the implementation must decide if an application has a permission or not is when the application either queries the presence of this permission for the first time or when it invokes an action that requires the permission for the first time. Application authors should be aware that in these situations, an implementation may prompt and ask the user. Depending on the implementation, this prompting (if necessary) can also happen at any point in time prior to this (e.g. at the application start up time).

A GEM terminal is required to be able to operate in a mode where it grants permission to provide access to all of the functionality required by the profiles and options that it supports when appropriately requested (e.g. via the permission request file). The mechanism for causing the terminal to operate in this mode is implementation-dependent. The granting of permissions for accessing functionality outside of the claimed GEM profile and options is not required.

NOTE 3: In the case of permissions represented by a Java class in DVB-J, this means that it will be possible to have such a permission granted if the corresponding class is required in the given profile, and it can be requested in the permission request file.

NOTE 4: This means that, for example, it is possible to grant the permission associated with dialling a phone number on a terminal that supports the interactive broadcast profile, even if the terminal implements the interaction channel using a cable modem. In this case, the dialling APIs will fail in a manner consistent with GEM.

## 12.6.2 Permission request file

### 12.6.2.0 General

The following rules shall apply for the processing of PRFs by a GEM terminal:

1) If there is only a DVB PRF present, the GEM terminal shall use it.

2) If there is a platform-specific PRF (e.g. an OCAP PRF) present on the corresponding target platform (i.e. an OCAP terminal), then the GEM terminal shall use it exclusively.

3) If there is a platform-specific PRF present on a non-corresponding target platform, then it shall be ignored. In this case, if a DVB PRF is present, the GEM terminal shall use it.

NOTE 1: The policy for granting of permissions outlined in clause 12.6.1, "General principles" is required to be supported. It is possible that these policy decisions will be made by an element that is downloaded to the terminal, e.g. OCAP [5] defines a "monitor application" that makes policy decisions. In cases such as this, the downloaded element is required to implement a policy consistent with GEM.

NOTE 2: The exact syntax of the permission request file specified in clause 12.6.2 is required to be supported. Because of the rules in clause 14.3, "XML notation", it cannot be extended by adding tag definitions.

NOTE 3: If a terminal cannot support functionality implied by a permission tag, it still supports the syntax of the permission tag. E.g. `capermission` tag is supported, even if support for the GEM CA APIs is not present.

### 12.6.2.1 File encoding

#### 12.6.2.1.1 XML

The Permission Request File is an XML File. Its syntax is defined by the following DTD. The `Name` used in the document type declaration shall be "`permissionrequestfile`".

#### 12.6.2.1.2 MHP/GEM 1.0

The `PublicLiteral` to be used for specifying this DTD in document type declarations of the XML files is:

```
"-//DVB//DTD Permission Request File 1.0//EN"
```

and the URL for the SystemLiteral is:

```
"http://www.dvb.org/mhp/dtd/permissionrequestfile-1-0.dtd".
```

The DTD is:

```
<!ELEMENT permissionrequestfile
    (file?,capermission?,applifecyclecontrol?,returnchannel?,tuning?,
    servicesel?,userpreferences?,network?, dripfeed?, persistentfilecredential*)>
<!ATTLIST permissionrequestfile
 orgid CDATA #REQUIRED
 appid CDATA #REQUIRED
>

<!ELEMENT file EMPTY>
<!ATTLIST file
    value                           (true|false) "true"
>
<!ELEMENT capermission (casystemid)+>
<!ELEMENT casystemid EMPTY>
<!ATTLIST casystemid
    entitlementquery                            (true|false) "false"
    id                          CDATA #REQUIRED
    mmi                         (true|false) "false"
    messagepassing                          (true|false) "false"
    buy                         (true|false) "false"
>

<!ELEMENT applifecyclecontrol EMPTY>
<!ATTLIST applifecyclecontrol
    value                           (true|false) "true"
>

<!ELEMENT returnchannel (defaultisp?,phonenumber*)>
<!ELEMENT defaultisp EMPTY>
<!ELEMENT phonenumber (#PCDATA)>
```

```
<!ELEMENT tuning EMPTY>
<!ATTLIST tuning
    value                           (true|false) "true"
>

<!ELEMENT servicesel EMPTY>
<!ATTLIST servicesel
    value                           (true|false) "true"
>

<!ELEMENT userpreferences EMPTY>
<!ATTLIST userpreferences
    write                           (true|false) "false"
    read                            (true|false) "true"
>

<!ELEMENT network (host)+>
<!ELEMENT host (#PCDATA)>
<!ATTLIST host
    action                          CDATA #REQUIRED
>

<!ELEMENT dripfeed EMPTY>
<!ATTLIST dripfeed
    value                           (true|false) "true"
>

<!ELEMENT persistentfilecredential (grantoridentifier,expirationdate,filename+,signature,
certchainfileid)>
<!ELEMENT grantoridentifier EMPTY>
<!ATTLIST grantoridentifier
    id                          CDATA #REQUIRED
>
<!ELEMENT expirationdate EMPTY>
<!ATTLIST expirationdate
    date                            CDATA #REQUIRED
>
<!ELEMENT filename (#PCDATA)>
<!ATTLIST filename
    write                           (true|false) "true"
    read                            (true|false) "true"
>
<!ELEMENT signature (#PCDATA)>
<!ELEMENT certchainfileid (#PCDATA)>
```

#### 12.6.2.1.3 MHP/GEM 1.1

The `PublicLiteral` to be used for specifying this DTD in document type declarations of the XML files is:

```
"-//DVB//DTD Permission Request File 1.1//EN"
```

and the URL for the SystemLiteral is:

```
"http://www.dvb.org/mhp/dtd/permissionrequestfile-1-1.dtd".
```

The DTD is:

```
<!ELEMENT permissionrequestfile
    (file?,capermission?,applifecyclecontrol?,returnchannel?,tuning?,
    servicesel?,userpreferences?,network?, dripfeed?, persistentfilecredential*,
    applicationstorage?,smartcardaccess?,provider?)>

<!ATTLIST permissionrequestfile
 orgid CDATA #REQUIRED
 appid CDATA #REQUIRED
>

<!ELEMENT file EMPTY>
<!ATTLIST file
    value                           (true|false) "true"
>

<!ELEMENT capermission (casystemid)+>
<!ELEMENT casystemid EMPTY>
<!ATTLIST casystemid
    entitlementquery                (true|false) "false"
```

```
    id                              CDATA #REQUIRED
    mmi                             (true|false) "false"
    messagepassing                  (true|false) "false"
    buy                             (true|false) "false"
>

<!ELEMENT applifecyclecontrol EMPTY>
<!ATTLIST applifecyclecontrol
    value                           (true|false) "true"
>

<!ELEMENT returnchannel (defaultisp?,phonenumber*)>
<!ELEMENT defaultisp EMPTY>
<!ELEMENT phonenumber (#PCDATA)>

<!ELEMENT tuning EMPTY>
<!ATTLIST tuning
    value                           (true|false) "true"
>

<!ELEMENT servicesel EMPTY>
<!ATTLIST servicesel
    value                           (true|false) "true"
>

<!ELEMENT userpreferences EMPTY>
<!ATTLIST userpreferences
    write                           (true|false) "false"
    read                            (true|false) "true"
>

<!ELEMENT network (host)+>
<!ELEMENT host (#PCDATA)>
<!ATTLIST host
    action                          CDATA #REQUIRED
>

<!ELEMENT dripfeed EMPTY>
<!ATTLIST dripfeed
    value                           (true|false) "true"
>

<!ELEMENT persistentfilecredential (grantoridentifier,expirationdate,filename+,signature,
certchainfileid)>
<!ELEMENT grantoridentifier EMPTY>
<!ATTLIST grantoridentifier
    id                              CDATA #REQUIRED
>
<!ELEMENT expirationdate EMPTY>
<!ATTLIST expirationdate
    date                            CDATA #REQUIRED
>
<!ELEMENT filename (#PCDATA)>
<!ATTLIST filename
    write                           (true|false) "true"
    read                            (true|false) "true"
>
<!ELEMENT signature (#PCDATA)>
<!ELEMENT certchainfileid (#PCDATA)>


<!-- ..................... new elements in MHP 1.1 ................ -->
<!ELEMENT applicationstorage (applicationstorageorg)*>
<!ATTLIST applicationstorage
    manageservice (true|false) "false"
    createservice (true|false) "false"
    deleteservice (true|false) "false"
    managecache (true|false) "false"
>

<!ELEMENT applicationstorageorg EMPTY>
<!ATTLIST applicationstorageorg
    orgid CDATA #REQUIRED
    storeservice (true|false) "false"
    removeservice (true|false) "false"
    storecache (true|false) "false"
    removecache (true|false) "false"
>
```

```
<!ELEMENT smartcardaccess EMPTY>

<!ELEMENT priviligedrce EMPTY>
<!ATTLIST privilegedrce
    value (internal|external) "internal"
>
<!ELEMENT provider (name)+ >
<!ELEMENT name (#PCDATA)>
<!ATTLIST name
    scope (application|system) "application"
>
```

#### 12.6.2.1.4 MHP/GEM 1.2

The `PublicLiteral` to be used for specifying this DTD in document type declarations of the XML files is:

```
    "-//DVB//DTD Permission Request File 1.2//EN"
```

and the URL for the SystemLiteral is:

```
    "http://www.dvb.org/mhp/dtd/permissionrequestfile-1-2.dtd".
```

The DTD is:

```
<!ELEMENT permissionrequestfile
    (file?,capermission?,applifecyclecontrol?,returnchannel?,tuning?,
    servicesel?,userpreferences?,network?, dripfeed?, persistentfilecredential*,
    applicationstorage?,smartcardaccess?,provider?, servicetypepermission?,
    monitorapplication? )>

<!ATTLIST permissionrequestfile
 orgid CDATA #REQUIRED
 appid CDATA #REQUIRED
>

<!ELEMENT file EMPTY>
<!ATTLIST file
    value                           (true|false) "true"
>

<!ELEMENT capermission (casystemid)+>
<!ELEMENT casystemid EMPTY>
<!ATTLIST casystemid
    entitlementquery                            (true|false) "false"
    id                          CDATA #REQUIRED
    mmi                         (true|false) "false"
    messagepassing                          (true|false) "false"
    buy                         (true|false) "false"
>

<!ELEMENT applifecyclecontrol EMPTY>
<!ATTLIST applifecyclecontrol
    value                           (true|false) "true"
>

<!ELEMENT returnchannel (defaultisp?,phonenumber*)>
<!ELEMENT defaultisp EMPTY>
<!ELEMENT phonenumber (#PCDATA)>


<!ELEMENT tuning EMPTY>
<!ATTLIST tuning
    value                           (true|false) "true"
>

<!ELEMENT servicesel EMPTY>
<!ATTLIST servicesel
    value                           (true|false) "true"
>

<!ELEMENT userpreferences EMPTY>
<!ATTLIST userpreferences
    write                           (true|false) "false"
    read                            (true|false) "true"
>
```

```
<!ELEMENT network (host)+>
<!ELEMENT host (#PCDATA)>
<!ATTLIST host
    action                          CDATA #REQUIRED
>

<!ELEMENT dripfeed EMPTY>
<!ATTLIST dripfeed
    value                           (true|false) "true"
>

<!ELEMENT persistentfilecredential (grantoridentifier,expirationdate,filename+,signature,
certchainfileid)>
<!ELEMENT grantoridentifier EMPTY>
<!ATTLIST grantoridentifier
    id                          CDATA #REQUIRED
>
<!ELEMENT expirationdate EMPTY>
<!ATTLIST expirationdate
    date                            CDATA #REQUIRED
>
<!ELEMENT filename (#PCDATA)>
<!ATTLIST filename
    write                           (true|false) "true"
    read                            (true|false) "true"
>
<!ELEMENT signature (#PCDATA)>
<!ELEMENT certchainfileid (#PCDATA)>


<!-- ..................... new elements in MHP 1.1 ................ -->
<!ELEMENT applicationstorage (applicationstorageorg)*>
<!ATTLIST applicationstorage
    manageservice (true|false) "false"
    createservice (true|false) "false"
    deleteservice (true|false) "false"
    managecache (true|false) "false"
>

<!ELEMENT applicationstorageorg EMPTY>
<!ATTLIST applicationstorageorg
    orgid CDATA #REQUIRED
    storeservice (true|false) "false"
    removeservice (true|false) "false"
    storecache (true|false) "false"
    removecache (true|false) "false"
>

<!ELEMENT smartcardaccess EMPTY>

<!ELEMENT priviligedrce EMPTY>
<!ATTLIST privilegedrce
    value (internal|external) "internal"
>

<!ELEMENT provider (name)+ >

<!ELEMENT name (#PCDATA)>
<!ATTLIST name
    scope (application|system) "application"
>

<!-- ..................... new elements in MHP 1.2 ................ -->

<!ELEMENT servicetypepermission EMPTY>
    <!ATTLIST servicetypepermission
    type (broadcast | abstract.mso) "broadcast"
    action (own | all) "all"
    value (true | false) "false"
>

<!ELEMENT monitorapplication EMPTY>
<!ATTLIST monitorapplication
    value (true | false) "false"
    name (registrar | service | servicemanager | security | reboot | handler.appFilter |
systemevent)
    #IMPLIED
>
```

#### 12.6.2.1.5 Number representation

##### 12.6.2.1.5.1 Hex

Unless stated otherwise (e.g. clause 14.5, "Text encoding of application identifiers") each hexadecimal value has the following form "0x" Nhex where "N" specifies the fixed number of significant digits in the value and hex is specified by the following BNF:

```
hex = digit | "A" | "B" | "C" | "D" | "E" | "F" | "a" | "b" | "c" | "d" | "e" | "f"
digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

### 12.6.2.2 File integrity

If the permission file is not parsable as defined by the XML parsing rules (see clause 14.3, "XML notation") it shall be ignored and hence no additional permissions are granted (or not granted in the case of `SelectPermission`).

### 12.6.2.3 Example

```
<?xml version="1.0"?>
<!DOCTYPE permissionrequestfile PUBLIC "-//DVB//DTD Permission Request File 1.0//EN"
"http://www.dvb.org/mhp/dtd/permissionrequestfile-1-0.dtd">
<permissionrequestfile orgid="0x000023d2" appid="0x4020">

    <file value="true"></file>

    <capermission>
        <casystemid
            id="0x1111" messagepassing="true"
            entitlementquery="true" mmi="false">
        </casystemid>
    </capermission>

    <applifecyclecontrol value="true"></applifecyclecontrol>

    <returnchannel>
        <defaultisp></defaultisp>
        <phonenumber>+3583111111</phonenumber>
        <phonenumber>+3583111112</phonenumber>
        <phonenumber></phonenumber>
    </returnchannel>

    <tuning value="false"></tuning>

    <servicesel value="true"></servicesel>

    <userpreferences read="true" write="false"></userpreferences>

    <network>
        <host action="connect">hostname</host>
    </network>

    <persistentfilecredential>
        <grantoridentifier id="0x05"></grantoridentifier>
        <expirationdate date="24/12/2032"></expirationdate>
        <filename read="true" write="false">5/15/dir1/scores</filename>
        <filename read="true" write="false">5/15/dir1/names</filename>
        <signature>02320329329293293292149314392942394329423943294239432
        </signature>
        <certchainfileid>3</certchainfileid>
    </persistentfilecredential>

    <provider>
        <name scope="application"> 00000B.EMV_PK11.VISA_REVOLVER </name>
    </provider>

</permissionrequestfile>
```

### 12.6.2.4 Permission request file name and location

The format for the permission request file name is:

```
'dvb.'<application name>'.perm'
```

The prefix "dvb" identifies this as a well known file specified by GEM. The portion "application name" carries the file name of the initial file of the application excluding any file name extension or suffix. The initial file depends on the application type as is shown in table 49 for the types defined in the present document.

**Table 49: Application names for different application types**

| Application type | | Path from which file name shall be extracted |
|---|---|---|
| Value | Meaning | |
| 0x0001 | DVB-J | The name initial_class_name, see MHP [1], table 102 |
| 0x0002 | DVB-HTML | The name initial_path_bytes, see MHP [1], table 104 |

This file shall be located in the same directory as the initial file.

### 12.6.2.5 Permission Request file

#### 12.6.2.5.1 Minimum permissions

If the permission file does not contain a valid permissionrequestfile element it shall be ignored and hence no additional permissions are granted (or not granted in the case of `SelectPermission`).

#### 12.6.2.5.2 Syntax and semantics

```
<!ATTLIST permissionrequestfile
    orgid CDATA #REQUIRED
    appid CDATA #REQUIRED
>
```

`orgid`: This attribute is a hexadecimal string (hex_string) that conveys the organisation id of the associated application. The encoding of this value is specified in clause 14.5, "Text encoding of application identifiers".

`appid`: This attribute is a hexadecimal string (hex_string) that conveys the application id of the associated application. The encoding of this value is specified in clause 14.5, "Text encoding of application identifiers".

#### 12.6.2.5.3 Defaults

NOTE: As defined by XML 1.0 [60], the defaults for attributes defined in the DTD (see clause 12.6.2.1, "File encoding") only apply when the element is present but the attribute concerned is omitted.

### 12.6.2.6 Credentials

Support for this mechanism is optional.

Since the "`signature`" and "`certchainfileid`" element documented in table 50 is dependent on security files used in the MHP application authentication mechanism, stated elements may be modified depending on the application authentication mechanism used. Thus, GEM terminal specifications that do not use the functional equivalent named "Application Authentication" as defined in clause 15.6, "Functional equivalents" shall define an equivalent mechanism using the same XML syntax.

NOTE: The credentials mechanism allows applications from different sources (e.g. signers) to share data. This particular mechanism is not required, but writers of GEM terminal specifications are invited to consider how such data sharing might be enabled in their network and/or environment. Possible mechanisms might involve file permissions, or a network-based solution.

A credential contains a resource description and is used to allow the owner of this resource (the grantor) to grant to the permission request file's application the right to access it. In GEM, the only resource that can be contained in a credential is a file (or a set of files of a directory). Credentials shall not grant access to directories, only files within them. This type of credential is named `persistentfilecredential` in the XML DTD. The credential contains an expiration date that allow the grantor to grant access to its resource for a limited duration. The credential is signed by the grantor. The signature checking is done by the implementation by getting the certificate of the grantor. The certificate can be found thanks to the information contained in the `certchainfileid` element.

The certificate file that leads to the public key of the signature shall be placed either in the same directory as the permission request file containing the credential or in one of its parent directories. The directory containing the permission request file shall be searched first and then recursively ascend the directory hierarchy checking certificate files until one is found with the file name matching the contents of the `certchainfileid` field of the credential. If the root of the file system is reached without finding a matching certificate then the file access shall not be granted.

The grantoridentifier in the `persistentfilecredential` shall match the `organisation_id` contained in the Subject organisationName field of the leaf certificate for file access to be granted. The grantoridentifier shall match the organisation id of the owner of the file to which access is granted by this `persistentfilecredential`. If either of these is untrue, access shall not be granted.

The persistent file credential is transmitted using the following XML DTD syntax:

```
<!ELEMENT persistentfilecredential (grantoridentifier,expirationdate,filename+,signature,
certchainfileid)>
<!ELEMENT grantoridentifier EMPTY>
<!ATTLIST grantoridentifier
    id CDATA #REQUIRED
>
<!ELEMENT expirationdate EMPTY>
<!ATTLIST expirationdate
    date CDATA #REQUIRED
>
<!ELEMENT filename (#PCDATA)>
<!ATTLIST filename
    write (true|false) "true"
    read (true|false) "true"
>
<!ELEMENT signature (#PCDATA)>
<!ELEMENT certchainfileid (#PCDATA)>
```

Table 50 provide more information about the different elements.

**Table 50**

| Elements | Comments |
|---|---|
| `grantoridentifier` | This element contains the 32 bit organization id identifying the grantor organization. The encoding of the CDATA attribute id of this element is "0x" followed by the encoding defined in 14.5, "Text encoding of application identifiers". |
| `expirationdate` | This element contains the expiration date of this credential.The implementation should ignore the certificate if the date has expired. The `CDATA` attribute `date` shall follow the following BNF syntax:<br>`2dec "/" 2dec "/" 4dec ; e.g. "01/12/2001"`<br><br>where:<br>`dec = "0" \| "1" \| "2" \| "3" \| "4" \| "5" \| "6" \| "7" \| "8" \| "9"`<br><br>Where the first 2 digits are the day number in the month, the second two digits are the month number in the year and the last 4 digits are the year. All values are in accordance with the Gregorian calendar.<br>NOTE: Numbers start counting from one and not zero unlike the java.util.GregorianCalendar class. |
| `filename` | This element contains the filename path and the read/write access rights that are granted on the file. The element consists of a `CDATA` string following the following BNF syntax:<br>`filename = persistentfilename`<br>`  \| persistentpath "/"`<br>`persistentfilename`<br>`  \| persistentpath "/" "*"`<br>`  \| persistentstpath "/" "-"`<br><br>where:<br>"/" is the file separator<br>"*" used at the end of a pathname indicates all files contained in that directory<br>"-" used at the end of a pathname indicates all files contained in that directory and existing subdirectories recursively at the time |

| Elements | Comments |
|---|---|
| | the permission request file is parsed. `persistentpath`, `persistentfilesubstring` and `persistentfilename` are defined in clause 14.6.1, "Persistent storage".<br><br>The file path is relative to the path obtained from the `dvb.persistent.root property`.<br><br>Receivers are required to support the same restrictions on the lengths of `persistentfilesubstrings` and `persistentfilesuffixes` as specified in clause 14.6.1, "Persistent storage".<br><br>The file path shall not start with a "/". It is relative to the path obtained from the `dvb.persistent.root` property.<br>The element has two attributes read and write that can take the value "true" or "false". |
| `signature` | This element contains a signature from the grantor. Signature structure is as defined in clause 12.4.2, "SignatureFile".<br>The signature is encoded using the Base64 content transfer encoding as specified in section 6.8 of RFC 2045 [59]. |
| `certchainfileid` | This element contains the identifier of the leaf certificate, i.e.the "x" in the file name "dvb.certificates.x". |

The signature is computed on the binary concatenation of the following fields in the following order.

**Table 51**

| Fields | Binary content |
|---|---|
| `grantee_identifier.organisation_id` | 32 bits |
| `grantee_identifier.application_id` | 16 bits |
| `grantor_identifier (organisation_id)` | 32 bits |
| `expiration_date` | 10 characters (e.g. "10/12/2001") in ASCII |
| filenames and actions (in the order they appear in the XML document) i.e.:<br><br>`for (i=0;i<filenumber;i++) {`<br>`  read`<br>`  write`<br>`  filepath`<br>`}` | <br><br>`4 or 5 char ("true" or "false")`<br>`4 or 5 char ("true" or "false")`<br>`string in ASCII (without any string termination character)` |

The implementation will check the validity of the credential by checking the signature with the grantor's public key that can be found in the grantor's certificate. Certificates are carried by the grantee in the file format defined clause 12.4.3, "CertificateFile". The certification chain authenticates the grantor's certificate. This chain shall derive from one of the root authorities.

The right to access a file shall not include the right to create it.

### 12.6.2.7 File Access

#### 12.6.2.7.1 Unsigned applications

Unsigned applications have no access to the persistent storage.

#### 12.6.2.7.2 Policy for signed applications

No access to the persistent storage, unless otherwise indicated in the Permission File.

Applications may request access to persistent storage either by credentials or the "file" element or both. These two mechanisms are independent. Once an application has access to a file granted by one mechanism, the other mechanism need not be considered further. Specifically, once a credential to access a file is granted to an application, the permissions for the file encapsulated by FileAccessPermission are no longer relevant.

NOTE: The intended use-case for this is a file containing an end-user's credit card details which may be intended to only be accessed by applications with a credential, e.g. from a bank. For this use-case, the application that put the credit card details in the file needs to set all the FileAccessPermissions to false in order to achieve the desired result.

When the permission request file requests access to persistent storage and this is granted, the file access policy is derived from the policy used in the Unix world.

An application owns the files it has created. The root directory of the persistent file namespace is defined by the 'dvb.persistent.root' property as described in clause 11.5.6, "Persistent Storage API". The files owned by an application shall be located in sub-directories below this directory, specifically one of the following two choices:

a) The sub-directory whose name is the `organisation_id` of the application concerned.

b) A sub-directory of the immediately previously defined directory whose name is the application_id of the application concerned.

The encodings of the organisation identifier and application identifier are as defined in clause 14.5, "Text encoding of application identifiers". By default, files created by an application will have owner read/write access only.

An Application can modify the access rights to a file it owns as follows:

- It can grant a read-only access, a write-only access or a read-write access to all applications having the same organisationId value.
- It can grant a read-only access, a write-only access or a read-write access to all applications.
- Write access to a directory is required to add or remove an entry in a directory.
- Read access to a directory is required to list the contents of a directory or access any of the files contained within in it.
- To read the contents of a file or directory, permission to read that file or directory and all directories on the path to the root of the persistent file system is required.

An application shall be granted access to a file if it qualifies for such access by application, organization or world access permissions.

See org.dvb.io.persistent see annex K, "DVB-J persistent storage API".

#### 12.6.2.7.3 Permission request syntax

```
<!ELEMENT file EMPTY>
<!ATTLIST file
 value (true|false) "true"
>
```

### 12.6.2.8 CA API

#### 12.6.2.8.0 GEM Introduction

This clause discusses the permissions for conditional access. As discussed in clause 11.10, "Java permissions" of the present document, the class `CAPermission` is not required to be present for GEM terminal specifications that do not include the functional equivalent named "conditional access" as defined in clause 15.6, "Functional equivalents"; for such GEM terminal specifications, the policies specified in clauses 12.6.2.8.1 and 12.6.2.8.2 do not apply. However, the syntax of the `capermission` tag as specified in clause 12.6.2.8.3, "Conditional Access Permission syntax" shall be supported in all cases; if the "conditional access" functional equivalent is not included, it is to be silently ignored.

#### 12.6.2.8.1 Unsigned applications

An unsigned application cannot access the following methods:

- `CAModule.buyEntitlements`

- CAModule.openMessageSession
- CAModuleManager.addMMIListener
- CAModule.queryEntitltements
- CAModule.listEntitlements

#### 12.6.2.8.2 Signed applications

By default, an application has limited access to the CA API functions (same default rights as an unsigned application).

In particular, the following method calls are not accessible to an unsigned application:

- CAModule.buyEntitlements
- CAModule.openMessageSession
- CAModuleManager.addMMIListener
- CAModule.queryEntitlements
- CAModule.listEntitlements

The permission request file requests the MHP terminal to grant additional rights to the application with the ConditionalAccess Permission described below.

#### 12.6.2.8.3 Conditional Access Permission syntax

The ConditionalAccess Permission is optional. When not present, the application has the default access rights. When present, the permission request file overrides the default rights for this application.

```
<!ELEMENT capermission (casystemid)+>
<!ELEMENT casystemid EMPTY>
<!ATTLIST casystemid
    entitlementquery                          true|false) "false"
    id                          CDATA #REQUIRED
    mmi                         (true|false) "false"
    messagepassing                          (true|false) "false"
    buy                         (true|false) "false"
>
```

The string specifying the CA system IDs has the following syntax:

```
CAIds                   =   CASystemId | "[" CASystemId "-" CASystemId "]" | "*"
CASystemId              =   "0x" 4hex
```

See clause 12.6.2.1.5.1, "Hex".

### 12.6.2.9 Application lifecycle control policy

Applications shall not launch broadcast applications that are not signalled in the currently selected service for the service context in which the application wishing to do the launching is running.

#### 12.6.2.9.1 Unsigned applications

An unsigned broadcast application can launch any application visible in the listing API that is signalled in the currently selected service for the service context in which the application wishing to do the launching is running.

An unsigned application can control (pause, stop, resume) the lifecycle of an application it has launched.

An unsigned application cannot control the lifecycle of an application it has not launched.

#### 12.6.2.9.2 Default policy for signed applications

By default, a signed application has the same rights as an unsigned application as concerns the application lifecycle control policy.

These default rights can be overridden by the permission request file as described below.

#### 12.6.2.9.3 Syntax

```
<!ELEMENT applifecyclecontrol EMPTY>
<!ATTLIST applifecyclecontrol
 value (true|false) "true"
>
```

value: When the boolean value is set to `true`, this means that the application can control the lifecycle of all the applications signalled in the currently selected service for the service context in which the application wishing to do the launching is running. When set to `false` the policy is as in clause 12.6.2.9.2, "Default policy for signed applications".

### 12.6.2.10 Return channel access policy

NOTE: The return channel access policy and permission described in this clause are required to be supported. Attention is drawn to the note at the end of clause 12.6.1, "General principles" relating to the return channel permission and return channel connections where it is not necessary to dial a phone, e.g. cable modems.

#### 12.6.2.10.1 Unsigned applications

An unsigned application may not use the return channel.

#### 12.6.2.10.2 Signed applications

By default, a signed application may not access the return channel, unless otherwise specified by the permission request file. The syntax of the return channel permission is so that it describes the phone numbers that the application may try to dial.

#### 12.6.2.10.3 Return channel permission syntax

```
<!ELEMENT returnchannel (defaultisp?,phonenumber*)>
<!ELEMENT phonenumber (#PCDATA)>
<!ELEMENT defaultisp EMPTY>
```

The syntax of the phone number string is as defined below.

```
phonenumber             = "+" digit digits | digits
digits                  = "" | digit digits
digit                   = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

The presence of the `defaultisp` tag indicates that the application is allowed to use the default ISP connection. If this tag is not present, then the application shall not be able to specify connecting to the default target except where the connection parameters for that target are explicitly specified by the application and are included in the phone number string.

When a phone number is given, this number defines a prefix of the allowed phone numbers and applications are allowed to call to all numbers that start with one of the prefixes defined in the permission request file, subject to the MHP terminal granting this right (see clause 12.6.1, "General principles").

NOTE: By defining an empty phone number tag (i.e. empty string as the prefix), the application could try to call to any phone number.

### 12.6.2.11 Tuning access policy

This clause discusses the permissions for tuning access. As discussed in 11.10, "Java permissions" of the present document, the class `TunerPermission` is not required to be present for GEM terminal specifications that do not support the "Tuning API" (clause 11.6.3) listed in table Error! Reference source not found. of the present document;

for such GEM terminal specifications the policies specified in clauses 12.6.2.11.1 and 12.6.2.11.2 do not apply. If the "Tuning" functional equivalent is not included, it is to be silently ignored. However, the syntax of the `tuning` tag as specified in clause 12.6.2.11.3, "Tuner Permission syntax" shall be supported in all cases; if the tuning API is not included, it is to be silently ignored.

#### 12.6.2.11.1 Unsigned applications

An unsigned application may not tune using the Tuning API.

#### 12.6.2.11.2 Signed applications

By default, a signed application may not tune using the Tuning API. However, the right to tune can be requested with the Tuning permission that can be put in the permission request file.

#### 12.6.2.11.3 Tuner Permission syntax

The syntax of this permission is as follows:

```
<!ELEMENT tuning EMPTY><

<!ATTLIST tuning
 value (true|false) "true"
>
```

The value true requests the permission to tune using the Tuning API.

### 12.6.2.12 Service selection policy

#### 12.6.2.12.1 Unsigned applications

Unsigned applications may not select a new service.

#### 12.6.2.12.2 Signed applications

By default, signed applications can select any new service, unless otherwise specified in the permission request file.

Service selection might require return channel access. If an application attempts to do a service selection using a locator that refers to an AIT File delivered over the return channel, and a return channel has not already been established or the application does not have permission to establish a connection via the default isp (see clause 12.6.2.10, "Return channel access policy"), then the service selection shall fail in a manner consistent with the defined failure mode for service selection when a security violation is encountered (e.g. a DVB-J application using the service selection API shall be thrown a SecurityException).

NOTE: See also clause 12.14.1, "Permission for application loading from the return channel".

#### 12.6.2.12.3 Service Selection Permission

The syntax of the service selection permission is as follows:

```
<!ELEMENT servicesel EMPTY>

<!ATTLIST servicesel
 value (true|false) "true"
>
```

If no service selection permission is present in the permission request file, the application has the default right, i.e. it can select any service.

The value "false" in this item in the permission request file, denies the right to select a new service.

### 12.6.2.13 Media API access policy

The media API is inside the sandbox.

#### 12.6.2.14 Inter-application communication policy

##### 12.6.2.14.1 Unsigned applications

Unsigned applications are allowed to communicate with each other through the inter-application communication API. However, unsigned and signed applications shall only be able to communicate with each-other using the inter-application communication API in the "dvb:/ixc/" namespace.

##### 12.6.2.14.2 Signed applications

Signed applications signalled in the same service are allowed to communicate with each other through the interapplication communication API. Signed applications shall only be able to communicate with unsigned applications using the inter-application API in the "dvb:/ixc/" namespace.

#### 12.6.2.15 User Setting and Preferences access policy

##### 12.6.2.15.1 Unsigned applications

Read access to:

- User Language.
- Parental Rating.
- DefaultFontSize.
- Country Code.

An unsigned application cannot access (neither read nor write) other preferences.

##### 12.6.2.15.2 Signed applications

By default, same as unsigned applications. The permission request file may include items that request read access to all user preferences and/or write access to all user preferences.

##### 12.6.2.15.3 Permission syntax

```
<!ELEMENT userpreferences EMPTY>

<!ATTLIST userpreferences
 write (true|false) "false"
 read (true|false) "true"
>
```

True value in the write and read attribute means that the permission for writing and reading, respectively, is requested.

#### 12.6.2.16 Network permissions

##### 12.6.2.16.1 Unsigned applications

Unsigned applications can not have access to the return channel and therefore can not access remote network hosts.

##### 12.6.2.16.2 Signed applications

For signed applications, the permission request file can contain a set of permissions that specify the hosts and actions for which permissions are requested.

##### 12.6.2.16.3 Permission syntax

```
<!ELEMENT network (host)+>
<!ELEMENT host (#PCDATA)>
<!ATTLIST host
```

```
 action CDATA #REQUIRED
>
```

The two strings that specific the host and the action shall be formatted as specified in the `java.net.SocketPermission` class as defined in clause 11.8, "Security".

### 12.6.2.17 Dripfeed permissions

#### 12.6.2.17.1 Unsigned applications

Has no access to drip feed mode.

#### 12.6.2.17.2 Default policy for signed applications

No access to drip feed mode unless otherwise indicated in the Permission file. When an application is granted access to the drip feed mode, it is able to instantiate a `DripFeedDataSource`.

#### 12.6.2.17.3 Permission request syntax

```
<!ELEMENT dripfeed EMPTY>
<!ATTLIST dripfeed
    value (true|false) "true"
>
```

### 12.6.2.18 Privileged Runtime Code Extension Permission

NOTE: DVB-J prohibits runtime code extension from data sources less secure than the original program code by virtue of the fact that interoperable DVB-J applications cannot directly create a ClassLoader, so this permission does not apply to DVB-J. See clause 11.10.1.4, "java.lang.RuntimePermission" and DVBClassLoader in annex I, "DVB-J fundamental classes".

#### 12.6.2.18.1 Unsigned Applications

Unsigned applications shall be permitted to do runtime code extension with string data from any source.

#### 12.6.2.18.2 Signed applications with no permission request file

Signed applications with no permission request file shall be permitted to do runtime code extension with string data from any source.

#### 12.6.2.18.3 Signed applications with a permission request file

By default, a signed application with a permission request file is prohibited from doing any runtime code extension from a string. The permission request file may include items that request the ability to do runtime code extension with internal strings only, or with both internal an external strings.

#### 12.6.2.18.4 Permission request syntax

```
<!ELEMENT priviligedrce EMPTY>
<!ATTLIST privilegedrce
    value (internal|external) "internal"
>
```

### 12.6.2.19 Application storage

#### 12.6.2.19.1 Unsigned applications

Have no permission to store applications.

#### 12.6.2.19.2 Default policy for signed applications

By default signed applications do not have permission to store any applications.

Permission to store applications can be requested using the permission request file.

#### 12.6.2.19.3 Permission request syntax

```
<!ELEMENT applicationstorage (applicationstorageorg)*>
<!ATTLIST applicationstorage
    manageservice: (true|false) "false"
    createservice: (true|false) "false"
    deleteservice: (true|false) "false"
    managecache: (true|false) "false"
>
```

manageservice: When set to "true", requests a permission to manage a stored application service with the organization ID of the current application. If set to false, this permission is not requested.

createservice: When set to "true", requests a permission to create a stored application service with the organization ID of the current application. If set to false, this permission is not requested.

deleteservice: When set to "true", requests a permission to remove a stored application service with the organization ID of the current application. If set to false, this permission is not requested.

managecache: When set to "true", requests a permission to manage an application cache with the organization ID of the current application. If set to false, this permission is not requested.

```
<!ELEMENT applicationstorageorg EMPTY>
<!ATTLIST applicationstorageorg
    orgid CDATA #REQUIRED
    storeservice (true|false) "false"
    removeservice (true|false) "false"
    storecache (true|false) "false"
    removecache (true|false) "false"
>
```

orgid: The `organisation_id` of the organization whose application and stored services this permission request concerns. This field is encoded in hexadecimal form as specified in clause 14.5, "Text encoding of application identifiers". Alternatively, this field may be set to "*" to indicate all organization IDs.

storeservice: When set to "true", requests a permission to store applications of the organization identified by the `organisation_id` into a stored service. If set to false, this permission is not requested.

removeservice: When set to "true", requests a permission to remove applications of the organization identified by the `organisation_id` from a stored service. If set to false, this permission is not requested.

storecache: When set to "true", requests a permission to store applications of the organization identified by the `organisation_id` into a cache. If set to false, this permission is not requested.

removecache: When set to "true", requests a permission to remove applications of the organization identified by the `organisation_id` from a cache. If set to false, this permission is not requested.

### 12.6.2.20 Non-CA smart card access

#### 12.6.2.20.1 Unsigned applications

Have no permission to access smart cards.

#### 12.6.2.20.2 Default policy for signed applications

By default signed applications do not have permission to access smart cards.

Permission to access smart cards can be requested using the permission request file.

#### 12.6.2.20.3 Permission request syntax

```
<!ELEMENT smartcardaccess EMPTY>
```

When the smartcardaccess element is included in the permission request file, the permission to access smart cards is requested. If the element is not included in the permission request file, this permission is not requested.

### 12.6.2.21 Provider Management

#### 12.6.2.21.1 Unsigned applications

Have no permission to install or remove providers.

#### 12.6.2.21.2 Signed applications

By default, signed applications have no permissions to add or remove providers.

Permission to insert or to remove a provider can be requested by requested using the permission request file.

An application shall only be granted permissions to insert or to remove providers that have the same organisation identifier as the application.

#### 12.6.2.21.3 Permission request syntax

```
<!ELEMENT provider (name)+ >
<!ELEMENT name (#PCDATA)>
<!ATTLIST name
    scope (application|system) "application"
>
```

The provider name shall be prefixed with the GEM organisation identifier of the organisation which is in charge of it. The syntax for the GEM orgId shall be compliant with clause 12.6.2.1.5.1, "Hex".

The provider name can be "GEM_orgId.Provider_name.Card_name". For example "0x0000000B. EMV_PK11.VISA_REVOLVER".

### 12.6.2.22 Service type selection policy

#### 12.6.2.22.1 Unsigned applications

Service type selection policy is irrelevant for unsigned applications since they are denied the ability to select a new service by clause 12.6.2.12.1, "Unsigned applications".

#### 12.6.2.22.2 Signed applications

By default a signed application running in an AbstractService shall be granted a this permission of type "abstract.mso" and other signed applications shall be granted this permission with type of "broadcast".

NOTE: The type "broadcast" includes stand-alone stored services 9.6.2, "Stored services" and return channel signalled services 9.6.1, "Applications loaded from the interaction channel".

Entries in the permission request file may be used to add extra permissions or to over ride the default permission.

#### 12.6.2.22.3 Permission request file syntax

```
<!ELEMENT servicetypepermission EMPTY>
<!ATTLIST servicetypepermission
    type (broadcast | abstract.mso) "broadcast"
    action (own | all) "all"
    value (true | false) "false"
>
```

#### 12.6.2.23 Privileged application access

##### 12.6.2.23.1 Unsigned applications

Unsigned applications shall not be able to access any privileged APIs.

##### 12.6.2.23.2 Signed applications

Only applications successfully authenticated as defined in clause 12.18, "Authentication of privileged applications" shall be able to access privileged APIs.

Multiple instances of the "monitorapplication" element may appear, one for each type of access that is requested.

##### 12.6.2.23.3 Permission request file syntax

```
<!ELEMENT monitorapplication EMPTY>
<!ATTLIST monitorapplication
    value (true | false) "false"
    name (registrar | service | servicemanager | security | reboot | handler.appFilter |
systemevent)
    #IMPLIED
>
```

## 12.7 Example of creating an application that can be authenticated

### 12.7.1 Scenario Example

This clause is informative and gives an example of how a file system carrying an application can be organized.

In this example, the file system carries single signed application Xlet1.

The main class of the application is the file `root/Xlet1/classes/Xlet1.class`, other files comprising the application are the following classes:

- `root/Xlet1/classes/foo1.class`
- `root/Xlet1/classes/subclasses/sub1.class`
- `root/Xlet1/classes/subclasses/sub2.class`

Xlet1 consumes data located in the file `root/Xlet1/data/Xlet1.dat`. This file is not authenticated.

Each subdirectory that contains signed files contains a hashfile.

Assumptions in this example:

- All the *.class files are signed.
- The file `root/Xlet1/data/Xlet1.dat` is not signed.
- The only digest algorithm used is MD5.

The file structure is shown below.

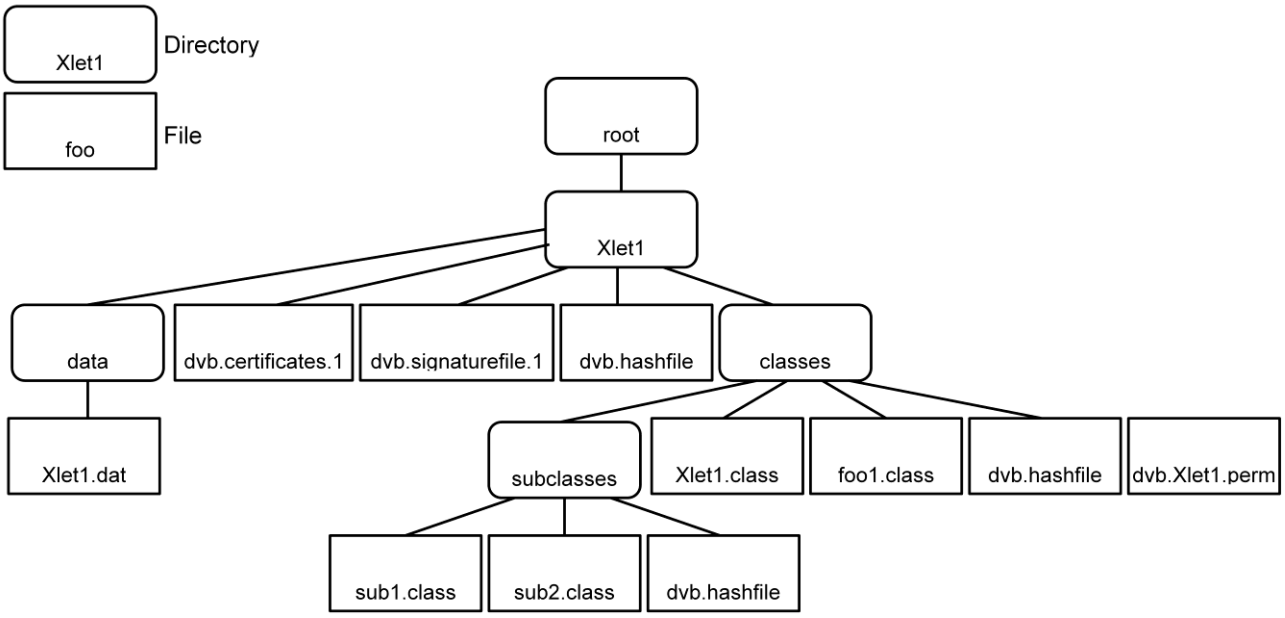



**Figure 15: Example of an authenticated file tree**

## 12.7.2 Hashes and signature computations:

### 12.7.2.1 Computation of the hashes of the root/Xlet1/classes/subclasses directory

a) Initialize the MD5 algorithm.

b) Apply the contents of the file "`root/Xlet1/classes/subclasses/sub1.class`" to the MD5 algorithm then apply the contents of the file "`root/Xlet1/classes/subclasses/sub2.class`" to the MD5 algorithm. The cumulative result from these two files is the result H0.

c) Construct the contents of the hash file for the directory `root/Xlet1/classes/subclasses/` as follows:

**Table 52: root/Xlet1/classes/subclasses/dvb.hashfile**

| Field | Comment |
|---|---|
| 1 | One digest |
| 1 | Type of digest algorithm = MD5 |
| 2 | Number of entries over which a MD5 hash has been computed |
| `sub1.class` | List of names of entries |
| `sub2.class` | |
| H0 | MD5 hash of files `sub1.class` and `sub2.class` |

NOTE: In this example we have chosen that the digests for this directory result from processing the concatenation of the contents of the files in the directory.

d) Create the hash file "`root/Xlet1/classes/subclasses/dvb.hashfile`" with the above contents.

### 12.7.2.2 Computation of the hashes of the of root/Xlet1/classes directory

e) Initialize the MD5 algorithm, compute the MD5 hash H2 using the contents of the file "`root/Xlet1/classes/subclasses/dvb.hashfile`".

f) Initialize the MD5 algorithm, compute the MD5 hash H3 using the contents of the file "`root/Xlet1/classes/Xlet1.class`".

g) Initialize the MD5 algorithm, compute the MD5 hash H4 using the contents of the file "root/Xlet1/classes/foo1.class".

h) Initialise the MD5 algorithm, compute the MD5 hash H5 using the contents of the file "root/Xlet1/classes/dvb.Xlet1.perm".

i) Construct the contents of the hash file for the directory root/Xlet1/classes/ as follows:

**Table 53: root/Xlet1/classes/dvb.hashfile**

| Field | Comment |
|---|---|
| 3 | Three digests |
| 1 | Type of digest algorithm = MD5 |
| 1 | Number of entries over which a MD5 hash has been computed |
| subclasses | List of names of entries |
| H2 | MD5 hash of subclasses directory |
| 1 | Type of digest algorithm = MD5 |
| 1 | Number of entries over which a MD5 hash has been computed |
| Xlet1.class | List of names of entries |
| H3 | MD5 hash of file Xlet1.class |
| 1 | Type of digest algorithm = MD5 |
| 1 | Number of entries over which a MD5 hash has been computed |
| foo1.class | List of names of entries |
| H4 | MD5 hash of file foo1.class |
| 1 | Type of digest algorithm = MD5 |
| 1 | Number of entries over which a MD5 hash has been computed |
| dvb.Xlet1.perm | List of names of entries |
| H5 | MD5 hash of file dvb.Xlet1.perm |

NOTE: In this example we have chosen to individually hash each object in this directory.

j) Create the hash file "root/Xlet1/classes/dvb.hashfile" with the above contents.

#### 12.7.2.3 Computation of the hashes of the of root/Xlet1 directory

k) Initialize the MD5 algorithm, compute the MD5 hash H5 using the contents of the file "root/Xlet1/classes/dvb.hashfile".

l) Construct the contents of the hash file for the directory root/Xlet1/classes/ as follows:

**Table 54: root/Xlet1/dvb.hashfile**

| Field | Comment |
|---|---|
| 3 | Three digests |
| 1 | Type of digest algorithm = MD5 |
| 1 | Number of entries over which a MD5 hash has been computed |
| classes | List of names of entries |
| H5 | MD5 hash of the classes directory |
| 0 | Type of digest algorithm = non authenticated data |
| 1 | Number of entries over which a MD5 hash has been computed |
| data | List of names of entries |
| <no digest in this case> | |
| 0 | Type of digest algorithm = non authenticated data |
| 1 | Number of entries over which a MD5 hash has been computed |
| dvb.certificates.1 List | List of names of entries |
| <no digest in this case> | |

NOTE: the root/Xlet1/classes entry is the only authenticated entry of the root/Xlet1 directory

m) Create the hash file "root/Xlet1/dvb.hashfile" with the above contents.

#### 12.7.2.4 Computation of the signature

n) Initialize the MD5 algorithm, compute the MD5 hash H6 using the contents of the file `root/Xlet1/dvb.hashfile`.

o) RSA-encrypt H6 with the private key corresponding to the public key that can be found in the leaf certificate in the file "`root/Xlet1/dvb.certificates.1`". In this example this has serial number 0123456.

p) ASN.1 encode the following structure:

- AuthorityCertIssuerName: Name of the CA.
- AuthorityCertSerialNumber: 0123456.
- HashSignatureAlgorithm: MD5.
- SignatureValue: result of step (o).

q) Put this structure into the signature file "`root/Xlet1/dvb.signaturefile.1`".

NOTE: The file system could contain other Xlets in their own authenticated sub-trees. If these use the same certificate chain as Xlet1 (above) then logically the certificates file is replicated at the root of each of the authenticated sub-trees. Some file systems support symbolic links. These may in some cases improve the efficiency of the broadcast.

## 12.8 Void

## 12.9 Certificate management

### 12.9.1 Certificate Revocation Lists

For profiles and/or targets that do not require compliance with the functional equivalent named "application authentication" as defined in clause 15.6, "Functional equivalents" GEM terminal specifications are not required to implement certificate revocation lists; such GEM terminal specifications are therefore not required to comply with the requirements of this clause and its subclauses.

#### 12.9.1.1 Introduction (informative)

Certificates may be revoked prior to their expiration time, e.g. if the broadcaster's private key is assumed to be compromised, or the broadcaster is no longer to be certified by the CA. Each CA publishes a list of revoked certificates, called a CRL (Certificate Revocation List). This contains the list of the serial number of revoked certificates.

During the validation process of a certificate chain, the CRL of each certification authority on the certification path is checked.

The number of CRLs to be supported per level is defined in annex G, "Minimum platform capabilities".

#### 12.9.1.2 Distribution of CRLs (informative)

Two routes from the broadcaster to the GEM terminal can be envisaged:

- Via the return channel.
- In the broadcast MPEG stream.

##### 12.9.1.2.1 Distribution via return channel

In the certificate extension fields, there is an optional field called "CRL Distribution points". This field can hold a URL pointing to a crlFile which can be downloaded.

This approach is suitable for obtaining CRLs relating to TLS authentication of return channel exchanges. It is not suitable for delivering CRLs for broadcast application authentication as:

- Not all GEM profiles require return channel support.
- GEM terminals may be able to receive broadcast applications when not connected to a return channel.
- It would not be acceptable to require a return channel session to authenticate each broadcast application.

#### 12.9.1.2.2 Distribution via MPEG stream

For a GEM terminal without a working return channel, the only way to deliver CRLs is via the broadcast MPEG Transport Stream.

### 12.9.1.3 CRL retention

#### 12.9.1.3.1 Requirement

GEM terminals shall retain CRLs or the information they contain (including serial numbers) in persistent storage because:

- This enhances security as it defends against attacks with signals that filter out the CRL and use a revoked certificate.
- It is more efficient to cache CRLs rather than downloading the CRLs for authentication of each application.

#### 12.9.1.3.2 Storage requirement

The minimum amount of persistent storage required to store CRLs is addressed in clause 12.12, "Platform minima".

#### 12.9.1.3.3 Storage management

The broadcast CRL and RCMM signalling (see clause 12.9.2, "Root certificate management") manages the use of the persistent storage.

If the CRL of a non-root certificate CA becomes too large the CA's certificate itself can be revoked by its parent CA who adds it to their CRL. For root certificates, the RCMM mechanism can be used.

### 12.9.1.4 CRL file location and naming convention

For CRLs that are authenticated by a non-root certificate the format of the name of files carrying CRLs shall be:

```
"dvb.crl.x"
```

In this case the "x" portion of the filename corresponds to the "x" portion of a certificate filename for the certificate file that authenticates the CRL.

For CRLs that are authenticated by a root certificate the format of the name of files carrying CRLs shall be:

```
"dvb.crl.root.x"
```

In this case the "x" portion of the filename is just a discriminator to ensure non-collision of CRL file names in the event that there is more than one in this directory.

The CRL filename may not be constant through time or across broadcasts. So, implementations shall not rely on this filename when caching the CRL.

All CRL files shall be located in a subdirectory of the root of the file system called "dvb.crl". The location of certificate files authenticating the CRL files shall follow the same rules as for the location of certificates relative to a signature file. That is the certificate files shall either be in the dvb.crl directory or in the root directory. See clause 12.4.3.5, "CertificateFile location and naming conventions".

#### 12.9.1.5 Operational model

Receivers are expected to inspect the set of CRL files periodically and cache the revocation information for future use. This inspection process can assume that a file system will not normally carry certificates revoked by CRL files carried in the same file system. So, inspection of the CRL set in a file system is not a precondition to authenticating other files in that file system.

GEM does not address the reliability with which implementations are required to consider certificate revocations when authenticating files.

#### 12.9.1.6 Examples

##### 12.9.1.6.1 Revocation of a broadcaster's certificate

If the broadcaster B's certificate is compromised:

a) The certification authority CA01 adds the serial number of broadcaster B's certificate to its CRL.

b) CA01 then sends the new CRL file to the broadcasters that use CA01.

c) These broadcasters (broadcaster A for instance) broadcast the new CRL file.

As soon as a GEM terminal has downloaded the new CRL file (after selecting one of broadcaster A's channel), the GEM terminal's CRL cache in persistent storage is updated. The GEM terminal is then protected against any malicious usage of the compromised certificate.

To continue authenticating applications broadcast "B" will require a new certificate.

##### 12.9.1.6.2 Revocation of a CA's certificate.

If the CRL of CA01 becomes too big, CA01's certificate could be revoked:

a) The root certification authority RCA0 adds the serial number of CA01's certificate to its CRL.

b) RCA0 sends the new CRL file to the broadcasters that use RCA0.

c) These broadcasters broadcast the new CRL file.

As soon as a GEM terminal has downloaded the new CRL file, its CRL cache in persistent storage for RCA0 is updated and CA01's CRL is removed from the cache (as CA01 has been revoked).

#### 12.9.1.7 CRL format

Each CRL file contains the CRL of one certification authority.

The encoding of the CRL is defined in the ITU-T Recommendation X.509 [52] is reproduced below for information. The fields Version, Time, CertificateSerialNumber correspond to fields with the same names in the certificate see clause 12.11, "The internet profile of X.509":

```
CertificateList ::= SEQUENCE {
        tbsCertList                     TBSCertList,
        signatureAlgorithm              AlgorithmIdentifier,
        signatureValue BIT STRING }

TBSCertList ::= SEQUENCE {
        version                         Version OPTIONAL,
                                        -- if present, shall be v2
        signature                       AlgorithmIdentifier,
        issuer                          Name,
        thisUpdate                      Time,
        nextUpdate                      Time OPTIONAL,
        revokedCertificates             SEQUENCE OF SEQUENCE {
        userCertificate                 CertificateSerialNumber,
        revocationDate                  Time,
        crlEntryExtensions              Extensions OPTIONAL
                                        -- if present, shall be v2
        } OPTIONAL,
```

```
        crlExtensions                       [0] EXPLICIT Extensions OPTIONAL
                                            -- if present, shall be v2
}
```

### 12.9.1.8 Profile of CRL

The profile of fields that correspond to fields in the certificate follow the profile in clause 12.5, "Profile of X.509 certificates for authentication of applications". This applies to the following fields:

signatureAlgorithm : follows clause 12.5.1, "signatureAlgorithm".

signatureValue: follows clause 12.5.2, "signatureValue".

version: follows clause 12.5.3, "version".

signature: follows clause 12.5.1, "signatureAlgorithm".

issuer: follows clause 12.5.4, "issuer".

thisUpdate: Publication date of this CRL. Follows the encoding of Time used for validity. See clause 12.5.5, "validity".

nextUpdate: Publication date of the next version of the CRL. Follows the encoding of Time used for validity. See clause 12.5.5, "validity".

userCertificate: SerialNumber of the revoked certificate. It is used to identify the revoked certificate.

The serial number shall be unique for a given CA. So, the pair [issuerName, serialNumber] shall be unique.

revocationDate: Date of revocation for a given certificate. Follows the encoding of Time used for validity. See clause 12.5.5, "validity".

crlExtensions: The syntax of the Extensions element is reproduced in clause 12.11.1.13, "Extensions". This element allows multiple extension elements to be carried. The set of extensions defined for certificates and CRLs is reproduced at clause 12.11.2, "Standard certificate extensions".

The following crlExtension shall be supported:

- AuthorityKeyIdentifier as defined in clause 12.4.2.1.

  This extension identifies the certificate that is needed to check the signature of the CRL.

  The crlExtension `AuthorityKeyIdentifier` shall be included in every CRL. GEM implementations shall use the `AuthorityKeyIdentifier` to find the certificate that carries the public key that is used to check the signature of the CRL and ignore any CRLs where this check fails or where the AuthorityKeyIdentifier is not present.

Other crlExtensions are optional.

### 12.9.1.9 CRL Processing

CRLs, or the information they contain (including serial numbers), shall be kept in persistent storage in the GEM terminal.

When a GEM terminal finds a file with the file name format given in clause 12.9.1.4, "CRL file location and naming convention", it shall do the following sequence:

a) Get the field thisUpdate and compare it with the last update for the CRL. If thisUpdate is not after the last update, ignore the CRL.

b) The signature of the CRL is verified and the signing certificate is verified to be from a trusted source. If the certificate cannot be trusted, the CRL is ignored. For CRLs which are authenticated by a non-root certificate, the certificate shall be trusted only if the chain from that certificate to the root in its certificate file is verified and the root certificate is one of those resident in the GEM receiver. For CRLs which are authenticated by a root certificate, the certificates which can be trusted shall be the root certificates resident in the GEM receiver.

c) Process the content of the CRL message:

- Store the list of serial numbers of revoked certificates in persistent storage.
- Update the stored value of thisUpdate with the value from the CRL.

d) If a CA's certificate has been revoked, remove its CRL (or the information which that CRL contained) if it was stored in the GEM terminal.

During the validation process of a certificate chain, the CRL of each certification authority on the certification path is checked.

## 12.9.2 Root certificate management

For profiles and/or targets that do not require compliance with the functional equivalent named "application authentication" as defined in clause 15.6, "Functional equivalents", GEM terminal specifications are not required to implement root certificate management; such GEM terminal specifications are therefore not required to comply with the requirements of this clause and its subclauses.

NOTE: When application authentication mechanism defined in GEM terminal specification does not require a root certificate in the terminal, RCMM will not be necessary.

### 12.9.2.1 Introduction

Clause 12.9.2 specifies an MHP root certificate management protocol that fulfils the requirements of GEM, but another protocol syntax and another distribution means are possible. Broadly speaking, GEM places requirements on both the semantics of a root certificate management message and requirements underlying its processing. GEM does not, however, mandate the syntax of the protocol message and its distribution means; this is left for GEM terminal specifications to define, either by using the MHP definition or by defining a functional equivalent.

Every compliant MHP terminal will have to maintain a set of X.509 root certificates in persistent storage. These root certificates will be placed in the MHP terminal by its manufacturer during the manufacturing process.

It could be necessary to update the set of root certificates for MHP terminals that are already deployed. Possible reasons that could require such an update include:

- A root certificate becoming compromised.
- Technical developments (such as the emergence of factorization algorithms) that require the use of greater key lengths in root certificates to provide adequate security.
- Retirement of a certificate due to its age.

So, It is necessary to have a standard mechanism to update this set.

NOTE: A manufacturer specific mechanism could require a different message for each manufacturer and so would be more expensive in terms of broadcast bandwidth.

The mechanism specified here uses messages called RCMM (Root Certificate Management Messages). These messages contain a set of new root certificates to add and a reference to the root certificates to remove.

### 12.9.2.2 Security of RCMM

The reference to the `nextNbOfSignature` field is to be interpreted as referring to the version of `nextNbOfSignature` from the present document, as specified in clause 12.9.2.3.

RCMM are authenticated by multiple signatures. An RCMM message will be accepted by an MHP terminal if and only if it has at least N signatures.

The initial value of N and the maximum value of N that MHP terminals will ever be asked to support are specified in clause 12.12, "Platform minima".

The use of multiple signatures guarantees that the set of root certificates can be updated securely even if one of the root certificates has been compromised.

RCMM can update the number of signatures required for future RCMM using the nextNbOfSignature field.

The RCMM message shall be signed with the key of the certificates to be removed.

### 12.9.2.3 Format of RCMM

RCMM encoding format as specified in MHP [1], clause 12.9.2.3 is not required for GEM terminal specifications. MHP [1], clause 12.9.2.3 shall apply to GEM terminal specifications that include the functional equivalent named "RCMM" as defined in clause 15.6, "Functional equivalents." GEM terminal specifications that do not include this MHP definition shall specify a functional equivalent that satisfies the requirements of this clause.

The unsigned RCMM message shall contain information sufficient to derive the following :

**Table 55: Root Certificate Management Message description**

| Function | Type |
|---|---|
| thisUpdate | Date |
| nextNbOfSignatures | 8-bit Unsigned Integer |
| addedCertificates | set of certificates |
| removedCertificates | set of certificate references |

GEM-based terminal specifications are allowed to define additional fields in the RCMM message.

The signatures shall subsequently be computed on the whole content of the unsigned RCMM, including possible extensions defined by GEM-based terminal specifications.

thisUpdate: Date of issue of the message.

nextNbOfSignatures: This field is used to change the minimum number of valid signatures required for an RCMM message. This value will be applied to the next RCMMs not to itself.

addedCertificates: List of root certificates to be added in persistent storage.

removedCertificates: Reference of the root certificates to be removed from persistent storage.

### 12.9.2.4 Distribution of RCMM

This clause is not required for GEM terminal specifications. It shall apply to GEM terminal specifications that include the functional equivalent named "RCMM" as defined in clause 15.6, "Functional equivalents." GEM terminal specifications that do not include this MHP definition shall specify a functional equivalent that satisfies the requirements of this clause.

RCMM messages shall be distributed to the MHP terminals in the broadcast MPEG Transport Stream. The distribution mechanism to be specified in a GEM terminal specification that do not support the functional equivalent named "RCMM" as defined in clause 15.6, "Functional equivalents" shall take into account the need to broadcast a set of historic RCMM messages in addition to the latest one.

### 12.9.2.5 RCMM Processing

References to RCMM messages named "dvb.rcmm.<x>" shall be interpreted as the set of historic RCMM messages as mentioned in clause 12.9.2.4 of the present document. References to RCMM message named "dvb.rcmm" shall be interpreted as the most recent RCMM message.

When an MHP terminal finds RCMM messages in a root directory, it shall perform the following sequence of operations:

a) If there has never been any RCMM processing (last update to be stored in the receiver not initialized), then process sequentially according to steps (b) to (j) all `dvb.rcmm.<x>` messages in ascending order including the `dvb.rcmm` message as the final step. Otherwise, identify the `dvb.rcmm.<x>` message whose field thisUpdate is the closest after the last update and process sequentially according to steps (b) to (j) all following `dvb.rcmm.<x>` in ascending order including the `dvb.rcmm` message as the final step. If the field thisUpdate from the `dvb.rcmm` message is not after the last update, ignore the messages.

For each RCMM message identified in step (a), perform the following:

b) If there is a nextNbOfSignatures in the RCMM, and if nextNbOfSignatures will be greater than the number of remaining root CAs (i.e. the number of root CA that would remain if the RCMM was processed), ignore the RCMM.

c) Get the number signatures from the RCMM, if this number is lower than the minimum required, ignore the message.

d) If the RCMM contains references to root certificates to remove, check this RCMM is signed with the keys belonging to the certificates to be removed. If at least one of these signatures is missing, ignore the RCMM message.

e) Check all the signatures of the RCMM. If any of the signatures is invalid, ignore the message.

f) Process the content of the RCMM message, add and remove root certificates according to the RCMM message.

g) Store in persistent storage the date of thisUpdate as the date of the last update.

h) If a root certificate is removed, the CRLs associated are also removed.

i) If there is a nextNbOfSignatures in the RCMM, replace the minimum number of signature required by the nextNbOfSignatures.

j) Since this process will permanently modify the terminal behaviour and is highly sensitive for security, this is allowed to have an implementation dependent enabling / confirmation step prior to performing the action (e.g. it may prompt the end user for verification). Such an enabling / confirmation step is allowed to reject this RCMM action.

The implementation shall ensure the following:

- The integrity of the persistent storage shall be kept if the power supply fails for any reason during the processing of any RCMM message in a `dvb.rcmm.<x>` file or the `dvb.rcmm` file. i.e. If the power is switched off during the processing of the RCMM message, the set of root certificates shall remain as if processing of that RCMM message had not started.

  I.e. iIf the power is switched off during the processing of the RCMM message, the set of root certificates shall remain as if processing of the RCMM message had not started.

- The minimum amount of persistent storage reserved to store the list of root certificate is specified in clause 12.12, "Platform minima". If there is not enough persistent storage available to process an RCMM message, it shall be ignored (to prevent inconsistencies).

NOTE: GEM is intentionally silent about the impact of RCMM processing on MHP applications which are already running.

#### 12.9.2.6 Example: Renewal of a root certificate

Let's assume the root certification authority RCA has two certificates in each MHP termninal:RC0 and RC1. If RC0 is compromised, RCA may wish to renew this certificate using the following steps:

a) RCA generates a new key pair and a new self signed certificate RC2.

b) RCA provides new certificates signed by RC1 to all the entities authenticated by RCA.

c) Wait until all entities authenticated by RCA have switched to the new certificates and have stopped using RC0.

d) RCA generates an RCMM message to add RC2 and to remove RC0. This RCMM will be double signed by RC0 and RC1 keys.

e) RCA will delivers this RCMM message to the broadcasters to update the MHP terminals.

(The broadcasting period should be long enough to update almost all of the MHP terminals in the field.)

f) RCA will provide RC1 and RC2 to set top box manufacturers as the new list of root certificates to put in set top boxes.

### 12.9.3 Test certificates

There shall be a means to make test root certificates (as required by clause 12.12, "Platform minima") available while the terminal is being tested. The method for doing so shall be implementation dependent; it may involve either replacing non-test root certificates, or storing test root certificates in addition to non-test root certificates.

Test root certificates shall only be functional when the terminal is being tested. Objects signed with test root certificates shall not be used other than for the purposes of testing.

GEM is intentionally silent on the test root certificates to be used.

NOTE: The test certificates guarantee that there is a way to cause a GEM terminal to grant all permissions requested by an application. The handling of these test certificates is decided by the authors of a GEM terminal specification.

## 12.10 Security on the return channel

General purpose security for the return channel is provided by the TLS (Transport Layer Security) protocol as described in RFC 2246 [58].

### 12.10.1 GEM application functionality

When implementing return channel security a GEM application shall:

- Implement the cipher suites identified in clause 12.10.2, "TLS cipher suites".

A GEM application is not required to implement the following:

- The functionality of being a server for the TLS protocol.
- Compliance with SSL 3.0.

GEM terminals shall support client authentication for TLS where the source of keys is provided to the class `javax.net.ssl.SSLContext` via a `javax.net.ssl.KeyManager`.

### 12.10.2 TLS cipher suites

The minimum set of cypher tools that implementations of the GEM profile of TLS shall implement are:

- RSA ().
- MD5.
- SHA-1.
- DES.
- AES.

More detail of this requirement is given in table 56 (see RFC 2246 [58] for definition of the terms) which identifies which methods are required in a GEM implementation. TLS cipher suites using AES are defined in RFC 3268 [79].

**Table 56: Profile of cipher suites that implementations are required to support**

| CipherSuite | Key Exchange | Cipher | Hash | Value (hex) | GEM status |
|---|---|---|---|---|---|
| TLS_NULL_WITH_NULL_NULL (note) | NULL | NULL | NULL | 00, 00 | Required |
| TLS_RSA_WITH_NULL_MD5 | RSA | NULL | MD5 | 00, 01 | Required |
| TLS_RSA_WITH_NULL_SHA | RSA | NULL | SHA-1 | 00, 02 | Required |
| TLS_RSA_EXPORT_WITH_DES40_CBC_SHA | RSA_EXPORT | DES40_CBC | SHA-1 | 00, 08 | Required |
| TLS_RSA_WITH_DES_CBC_SHA | RSA | DES_CBC | SHA-1 | 00, 09 | Required |
| TLS_RSA_WITH_3DES_EDE_CBC_SHA | RSA | 3DES_EDE_CBC | SHA-1 | 00, 0A | Required |
| TLS_RSA_WITH_AES_128_CBC_SHA | RSA | AES | SHA-1 | 00, 2F | Required |
| TLS_RSA_WITH_AES_256_CBC_SHA | RSA | AES | SHA-1 | 00, 35 | Required |
| NOTE: This cipher suite is only used by a TLS implementation during the negotiation of a connection. It is not required to be enabled as a cipher suite that is available for a negotiated connection. | | | | | |

NOTE: The ciphers TLS_RSA_WITH_NULL_MD5 and TLS_RSA_WITH_NULL_SHA provide integrity checking but without confidentiality (i.e. data are in clear). Data encryption is very time consuming for a server. They are useful for applications in which data don't need to be encrypted but in which data integrity is very important. For these applications, the server will only have to compute a HMAC for every message exchanged.

## 12.10.3 Downloading of certificates for TLS

### 12.10.3.1 Introduction

Before the TLS connection can be established, a GEM application has to ensure that the certificate list sent by a server contains at least one trusted certificate. In computer environment, this is simply done by checking the list of certificates against one certificate that is resident in the computer.

In the GEM environment, a downloadable application can establish a TLS session. This can be used for e.g. sensitive transactions. In such a scenario, the application knows which server to connect to, and also knows one certificate against which it can check that a given certificate chain contains the expected certificate that it knows and trusts.

The API that is used by a downloadable application is described in clause 11.8.2, "APIs for return channel security". The process of server authentication involves the checking of the certificate chain sent by the TLS server.

This clause specifies how the GEM terminal identifies and manages the TLS certificates that are downloaded along with the application and how verification (or otherwise) is presented to the application).

### 12.10.3.2 Usage of certificate in TLS

#### 12.10.3.2.1 When certificates are delivered with the application

One or several TLS root certificates can be optionally broadcast along with the application.

When the certificate chain sent by the TLS server is not compatible with any of the TLS root certificates sent with the application an `IOException` will be thrown.

##### 12.10.3.2.1.1 Certificate file naming and location

To facilitate certificate chain checking the name of the certificate file shall be:

```
dvb.tls.organisation_id.application_id.x
```

where:

- "x" is an optional string discriminating certificates where necessary.
- The encoding of `organisation_id` and application_id are specified in clause 14.5, "Text encoding of application identifiers".

Location of the TLS certificates is application type dependent. See table 57.

**Table 57: TLS certificate locator semantics**

| Application_type | Description |
|---|---|
| 0x0000 | Reserved. |
| 0x0001 | For DVB-J the TLS certificate(s) are placed in the base directory of the application as defined in MHP [1] clause 10.9.2 "DVB-J application location descriptor". |
| 0x0002 | For DVB-HTML the TLS certificate(s) are placed at the physical root of the application as defined in MHP [1] clause 10.10.2 "DVB-HTML application location descriptor". |
| 0x0003 to 0xFFFF | Reserved for future use. |

#### 12.10.3.2.1.2 Certificate authentication

To be considered valid TLS certificates the certificate files shall be authenticated members of the same authenticated sub-tree as the application.

#### 12.10.3.2.1.3 Certificate file format

Each TLS certificate file contains a single certificate. The file format is the same as that used by certificate files for application authentication, see clause 12.4.3, "CertificateFile".

### 12.10.3.2.2 When no certificates are provided

When there are no TLS certificates sent with the application then the implementation will allow connection to be established to any server. The application can then use the JSSE API (see clause 11.8.2, "APIs for return channel security") to retrieve the certificate chain and check that it contains what the application requires. In such a case both name and public keys need to be checked by the application if the application wants to be sure of the remote server.

### 12.10.3.2.3 CRL distribution points

GEM implementations are free to ignore this field during TLS server authentication over the return channel.

## 12.11 The internet profile of X.509 (informative)

The text that follows summarizes the technical features of RFC 2459 [55] and references the different profile decisions made for the different GEM application areas.

## 12.11.1 Main part of the certificate

### 12.11.1.1 Certificate

```
Certificate  ::=  SEQUENCE  {
        tbsCertificate          TBSCertificate,
        signatureAlgorithm      AlgorithmIdentifier,
        signatureValue          BIT STRING }
```

### 12.11.1.2 signatureAlgorithm

The signatureAlgorithm field contains the identifier for the cryptographic algorithm used by the CA to sign this certificate.

An algorithm identifier is defined by the following ASN.1 structure:

```
AlgorithmIdentifier  ::=  SEQUENCE {
                    algorithm           OBJECT IDENTIFIER,
                    parameters          ANY DEFINED BY algorithm OPTIONAL }
```

The algorithm identifier is used to identify a cryptographic algorithm. The OBJECT IDENTIFIER component identifies the algorithm. The contents of the optional parameters field will vary according to the algorithm identified.

This field must contain the same algorithm identifier as the signature field in the sequence TBSCertificate.

See clause 12.5.1, "signatureAlgorithm".

NOTE: RFC 2459 [55], section 7.2 lists the signature algorithms supported by that profile.

For all of the currently specified algorithms possible non-NULL parameters shall be ignored.

#### 12.11.1.3 signatureValue

The signatureValue field contains a digital signature computed upon the ASN.1 DER encoded tbsCertificate. The ASN.1 DER encoded tbsCertificate is used as the input to the signature function. This signature value is then ASN.1 encoded as a BIT STRING and included in the Certificate's signature field.

NOTE: RFC 2459 [55], section 7.2 describes in detail this process for the algorithms supported by that profile.

By generating this signature, a CA certifies the validity of the information in the tbsCertificate field. In particular, the CA certifies the binding between the public key material and the subject of the certificate.

See clause 12.5.2, "signatureValue".

#### 12.11.1.4 tbsCertificate

The field contains the names of the subject and issuer, a public key associated with the subject, a validity period, and other associated information. The tbscertificate may also include extensions.

The pair issuer / serialNumber uniquely identifies the certificate.

```
TBSCertificate ::=  SEQUENCE  {
        version         [0]  EXPLICIT Version DEFAULT v1,
        serialNumber         CertificateSerialNumber,
        signature            AlgorithmIdentifier,
        issuer               Name,
        validity             Validity,
        subject              Name,
        subjectPublicKeyInfo SubjectPublicKeyInfo,
        issuerUniqueID  [1]  IMPLICIT UniqueIdentifier OPTIONAL,
                             -- If present, version shall be v2 or v3
        subjectUniqueID [2]  IMPLICIT UniqueIdentifier OPTIONAL,
                             -- If present, version shall be v2 or v3
        extensions      [3]  EXPLICIT Extensions OPTIONAL
                             -- If present, version shall be v3 }
```

#### 12.11.1.5 version

This field describes the version of the encoded certificate. When extensions are used, as expected in this profile, use X.509 version 3 (value is 2). If no extensions are present, but a UniqueIdentifier is present, use version 2 (value is 1). If only basic fields are present, use version 1 (the value is omitted from the certificate as the default value).

Implementations should be prepared to accept any version certificate. At a minimum, conforming implementations must recognize version 3 certificates.

```
Version ::=  INTEGER { v1(0), v2(1), v3(2) }
```

Generation of version 2 certificates is not expected by implementations based on this profile.

See clause 12.5.3, "version".

### 12.11.1.6 serialNumber

The serial number is an integer assigned by the CA to each certificate. It must be unique for each certificate issued by a given CA (i.e. the issuer name and serial number identify a unique certificate).

```
CertificateSerialNumber ::=  INTEGER
```

### 12.11.1.7 signature

This field contains the algorithm identifier for the algorithm used by the CA to sign the certificate.

This field must contain the same algorithm identifier as the signatureAlgorithm field in the sequence Certificate. The contents of the optional parameters field will vary according to the algorithm identified.

If the signatureAlgorithm is different from the algorithm identifier in the signature field, the signature shall be rejected as being inconsistent.

See clause 12.5.1, "signatureAlgorithm".

NOTE: RFC 2459 [55], section 7.2 lists the supported signature algorithms for that profile.

### 12.11.1.8 issuer

The issuer field identifies the entity who has signed and issued the certificate. The issuer field must contain a non-empty distinguished name (DN). The issuer field is defined as the X.501 type Name. ITU-T Recommendation X.501 [51] Name is defined by the following ASN.1 structures:

```
Name ::= CHOICE {
            RDNSequence }

RDNSequence ::= SEQUENCE OF RelativeDistinguishedName

RelativeDistinguishedName ::=
            SET OF AttributeTypeAndValue

AttributeTypeAndValue ::= SEQUENCE {
            type        AttributeType,
            value       AttributeValue }

AttributeType ::= OBJECT IDENTIFIER

AttributeValue ::= ANY DEFINED BY AttributeType

DirectoryString ::= CHOICE {
            teletexString                   TeletexString (SIZE (1..MAX)),
            printableString                 PrintableString (SIZE (1..MAX)),
            universalString                 UniversalString (SIZE (1..MAX)),
            utf8String                      UTF8String (SIZE (1.. MAX)),
            bmpString                       BMPString (SIZE (1..MAX)) }
```

The Name describes a hierarchical name composed of attributes, such as country name, and corresponding values, such as US. The type of the component AttributeValue is determined by the AttributeType; in general it will be a DirectoryString.

See clause 12.5.4, "issuer".

### 12.11.1.9 validity

The certificate validity period is the time interval during which the CA warrants that it will maintain information about the status of the certificate. The field is represented as a SEQUENCE of two dates: the date on which the certificate validity period begins notBefore) and the date on which the certificate validity period ends (notAfter). Both notBefore and notAfter may be encoded as UTCTime or GeneralizedTime.

CAs conforming to this profile must always encode certificate validity dates through the year 2049 as UTCTime; certificate validity dates in 2050 or later must be encoded as GeneralizedTime.

```
Validity ::= SEQUENCE {
        notBefore      Time,
        notAfter       Time }

Time ::= CHOICE {
        utcTime        UTCTime,
        generalTime    GeneralizedTime }
```

A subject certificate's validity is limited by the validity period of the issuer's certificate.

#### 12.11.1.9.1 UTCTime

The universal time type, UTCTime, is a standard ASN.1 type intended for representation of dates and time. UTCTime specifies the year through the two low order digits and time is specified to the precision of one minute or one second. UTCTime includes either Z (for Zulu, or Greenwich Mean Time) or a time differential.

For the purposes of this profile, UTCTime values must be expressed Greenwich Mean Time (Zulu) and must include seconds (i.e. times are YYMMDDHHMMSSZ), even where the number of seconds is zero. Conforming systems must interpret the year field (YY) as follows:

- Where YY is greater than or equal to 50, the year shall be interpreted as 19YY; and
- Where YY is less than 50, the year shall be interpreted as 20YY.

#### 12.11.1.9.2 GeneralizedTime

The generalized time type, GeneralizedTime, is a standard ASN.1 type for variable precision representation of time. Optionally, the GeneralizedTime field can include a representation of the time differential between local and Greenwich Mean Time.

For the purposes of this profile, GeneralizedTime values must be expressed Greenwich Mean Time (Zulu) and must include seconds (i.e. times are YYYYMMDDHHMMSSZ), even where the number of seconds is zero. GeneralizedTime values must not include fractional seconds.

See clause 12.5.5, "validity".

### 12.11.1.10 subject

The subject field identifies the entity associated with the public key stored in the SubjectPublicKeyInfo field. It thus represents the entity whose public key is certified. It is encoded as a Distinguished Name (see clause 12.11.1.8, "issuer"). This name must be unique for each subject entity certified by one CA as defined by the issuer field.

#### 12.11.1.10.1 issuerUniqueID

This field is optional and appears in X.509 v2 or v3 only. It enables to reuse the IssuerName over time. It is redundant with the issuer Name, and it is proposed here that this field be not parsed by the client.

#### 12.11.1.10.2 subjectUniqueID

This field is optional and appears in X.509 v2 or v3 only. It enables to reuse the subjectName over time. It is redundant with the subject Name, and it is proposed here not to use this field.

The subject name may be carried in the subject field and/or the subjectAltName extension. If the subject is a CA (e.g. the basic constraints extension, as discussed in RFC 2459 [55], section 4.2.1.10, is present and the value of cA is TRUE,) then the subject field must be populated with a non-empty distinguished name matching the contents of the issuer field (see RFC 2459 [55], section 4.1.2.4) in all certificates issued by the subject CA. If subject naming information is present only in the subjectAltName extension (e.g. a key bound only to an email address or URI), then the subject name must be an empty sequence and the subjectAltName extension must be critical.

Where it is non-empty, the subject field must contain an X.500 Distinguished Name (DN). The DN must be unique for each subject entity certified by the one CA as defined by the issuer name field. A CA may issue more than one certificate with the same DN to the same subject entity.

The subject name field is defined as the X.501 type Name. Implementation requirements for this field are those defined for the issuer field (see RFC 2459 [55], section 4.1.2.4). When encoding attribute values of type DirectoryString, the encoding rules for the issuer field must be implemented. Implementations of GEM must be prepared to receive subject names containing the attribute types required for the issuer field. Implementations of the present document should be prepared to receive subject names containing the recommended attribute types for the issuer field. The syntax and associated object identifiers (OIDs) for these attribute types are provided in the ASN.1 modules in RFC 2459 [55] Appendices A and B. Implementations of the present document may use these comparison rules to process unfamiliar attribute types (i.e. for name chaining). This allows implementations to process certificates with unfamiliar attributes in the subject name.

In addition, legacy implementations exist where an RFC 822 [26] name is embedded in the subject distinguished name as an EmailAddress attribute. The attribute value for EmailAddress is of type IA5String to permit inclusion of the character '@', which is not part of the PrintableString character set. EmailAddress attribute values are not case sensitive (e.g. "fanfeedback@redsox.com" is the same as "FANFEEDBACK@REDSOX.COM").

Conforming implementations generating new certificates with electronic mail addresses must use the rfc822Name in the subject alternative name field (see RFC 2459 [55], section 4.2.1.7) to describe such identities. Simultaneous inclusion of the EmailAddress attribute in the subject distinguished name to support legacy implementations is deprecated but permitted.

#### 12.11.1.11 SubjectPublic Key Info

This field is used to carry the public key which is certified and identifies the algorithm with which the key is used. The algorithm is identified using the AlgorithmIdentifier structure (see clause 12.11.1.2, "signatureAlgorithm") and the public key is represented as a bitstring.

```
SubjectPublicKeyInfo  ::=  SEQUENCE  {
        algorithm            AlgorithmIdentifier,
        subjectPublicKey     BIT STRING  }
```

See clause 12.5.7, "SubjectPublic Key Info".

NOTE: RFC 2459 [55], section 7.3 lists the supported algorithms for that profile.

#### 12.11.1.12 Unique Identifiers

These fields may only appear if the version is 2 or 3. The subject and issuer unique identifiers are present in the certificate to handle the possibility of reuse of subject and/or issuer names over time. This profile recommends that names not be reused for different entities and that Internet certificates not make use of unique identifiers. CAs conforming to this profile should not generate certificates with unique identifiers. Applications conforming to this profile should be capable of parsing unique identifiers and making comparisons.

```
UniqueIdentifier  ::=  BIT STRING
```

See clause 12.5.8, "Unique Identifiers".

#### 12.11.1.13 Extensions

This field may only appear if the version is 3 and is optional. If present, this field is a SEQUENCE of one or more certificate extensions.

```
Extensions  ::=  SEQUENCE SIZE (1..MAX) OF Extension

Extension  ::=  SEQUENCE  {
        extnID      OBJECT IDENTIFIER,
        critical    BOOLEAN DEFAULT FALSE,
        extnValue   OCTET STRING  }

The extensions are defined in ITU-T Recommendation X.509 [52].
```

See clause 12.5.9, "Extensions".

## 12.11.2 Standard certificate extensions

See table 47, "Profile for standard certificate extensions".

### 12.11.2.1 Authority key identifier

This extension is used where an issuer has multiple signing keys. It provides a means of finding the public key that can be used to check the signature of the certificate.

The identification can be based on either the keyIdentifier or the on the pair (authorityCertIssuer, AuthorityCertSerialNumber). It is recommended to use the pair (authorityCertIssuer, AuthorityCertSerialNumber) instead of the keyIdentifier because there is no common agreement on the way to compute a unique KeyIdentifier.

```
AuthorityKeyIdentifier ::= SEQUENCE {
  keyIdentifier             [0] KeyIdentifier           OPTIONAL,
  authorityCertIssuer       [1] GeneralNames            OPTIONAL,
  authorityCertSerialNumber [2] CertificateSerialNumber OPTIONAL }
```

### 12.11.2.2 Subject key identifier

This extension provides of identifying certificates that contain a particular public key.

```
SubjectKeyIdentifier ::= KeyIdentifier
```

### 12.11.2.3 Key usage

The key usage extension defines the purpose of the key contained in the certificate.

```
keyUsage ::= BIT STRING {
  digitalSignature      (0),
  nonRepudiation        (1),
  keyEncipherment       (2),
  dataEncipherment      (3),
  keyAgreement          (4),
  keyCertSign           (5),
  cRLSign               (6),
  encipherOnly          (7),
  decipherOnly          (8)
}
```

### 12.11.2.4 Private key usage period

This field is used to defined a period of validity for the private key which is different from the period of validity of the certificate. This is only meaningful for digital signature keys.

```
privateKeyUsagePeriod EXTENSION ::= {
  SYNTAX        PrivateKeyUsagePeriod
  IDENTIFIED BY id-ce-privateKeyUsagePeriod }

PrivateKeyUsagePeriod ::= SEQUENCE {
  notBefore        [0]  GeneralizedTime OPTIONAL,
  notAfter         [1]  GeneralizedTime OPTIONAL }
  ( WITH COMPONENTS {..., notBefore PRESENT} |
    WITH COMPONENTS {..., notAfter PRESENT} )
```

### 12.11.2.5 Certificate policies

This field is used to define a policy defining the purpose for which the certificate may be used.

Applications may have a list of specific policies they will accept, the certificate validation software must be able to compare the policy OIDs found in the certificate to that list.

```
certificatePolicies EXTENSION ::= {
  SYNTAX      CertificatePoliciesSyntax
  IDENTIFIED BY id-ce-certificatePolicies }

CertificatePoliciesSyntax ::= SEQUENCE SIZE (1..MAX) OF PolicyInformation
```

```
PolicyInformation ::= SEQUENCE {
  policyIdentifier   CertPolicyId,
  policyQualifiers   SEQUENCE SIZE (1..MAX) OF
                     PolicyQualifierInfo OPTIONAL }

CertPolicyId ::= OBJECT IDENTIFIER

PolicyQualifierInfo ::= SEQUENCE {
  policyQualifierId          CERT-POLICY-QUALIFIER.&id
                             ({SupportedPolicyQualifiers}),
  qualifier                  CERT-POLICY-QUALIFIER.&Qualifier
          ({SupportedPolicyQualifiers}{@policyQualifierId})
          OPTIONAL }

SupportedPolicyQualifiers CERT-POLICY-QUALIFIER ::= { ... }
```

### 12.11.2.6 Policy mappings

This field is used to define equivalence between policies from different CA's policy Domain.

```
policyMappings EXTENSION ::= {
  SYNTAX      PolicyMappingsSyntax
  IDENTIFIED BY id-ce-policyMappings }

PolicyMappingsSyntax ::= SEQUENCE SIZE (1..MAX) OF SEQUENCE {
  issuerDomainPolicy    CertPolicyId,
  subjectDomainPolicy   CertPolicyId }
```

### 12.11.2.7 Subject Alternative Name

This field is used to define additional identities for the subject name of the certificate (such as an internet email address).

```
subjectAltName EXTENSION ::= {
  SYNTAX    GeneralNames
  IDENTIFIED BY id-ce-subjectAltName }

GeneralNames ::= SEQUENCE SIZE (1..MAX) OF GeneralName

GeneralName ::= CHOICE {
  otherName                 [0]  INSTANCE OF OTHER-NAME,
  rfc822Name                [1]  IA5String,
  dNSName                   [2]  IA5String,
  x400Address               [3]  ORAddress,
  directoryName             [4]  Name,
  ediPartyName              [5]  EDIPartyName,
  uniformResourceIdentifier [6]  IA5String,
  iPAddress                 [7]  OCTET STRING,
  registeredID              [8]  OBJECT IDENTIFIER }
```

### 12.11.2.8 Issuer Alternative Name

This field is similar to the previous one for the issuer identity.

```
issuerAltName EXTENSION ::= {
  SYNTAX      GeneralNames
  IDENTIFIED BY id-ce-issuerAltName }
```

### 12.11.2.9 Subject Directory attributes

This field is used to carry optional directory attributes associated with the subject.

```
subjectDirectoryAttributes EXTENSION ::= {
  SYNTAX      AttributesSyntax
  IDENTIFIED BY id-ce-subjectDirectoryAttributes }

AttributesSyntax ::= SEQUENCE SIZE (1..MAX) OF Attribute
```

### 12.11.2.10 Basic Constraints

This field indicates if the subject of the certificate is a certification authority and how deep a certification path may exist through that path.

This extension shall appear in every CA Certificates. It will be used to prevent unauthorized entities to act as a CA.

```
basicConstraints EXTENSION ::= {
  SYNTAX      BasicConstraintsSyntax
  IDENTIFIED BY id-ce-basicConstraints }

BasicConstraintsSyntax ::= SEQUENCE {
  cA                   BOOLEAN DEFAULT FALSE,
  pathLenConstraint    INTEGER (0..MAX) OPTIONAL }
```

### 12.11.2.11 Name Constraints

This field indicates a namespace within which all subject names in subsequent certificates in a certification path shall be located.

```
nameConstraints EXTENSION ::= {
  SYNTAX      NameConstraintsSyntax
  IDENTIFIED BY id-ce-nameConstraints }

NameConstraintsSyntax ::= SEQUENCE {
  permittedSubtrees          [0]  GeneralSubtrees OPTIONAL,
  excludedSubtrees           [1]  GeneralSubtrees OPTIONAL }

GeneralSubtrees ::= SEQUENCE SIZE (1..MAX) OF GeneralSubtree

GeneralSubtree ::= SEQUENCE {
  base                GeneralName,
  minimum        [0]  BaseDistance DEFAULT 0,
  maximum        [1]  BaseDistance OPTIONAL }

BaseDistance ::= INTEGER (0..MAX)
```

### 12.11.2.12 Policy Constraints

This field is only used in CA's certificate (i.e. not in end entity certificate). It is used to prohibit policy mapping or to require that each certificate in a path contain an acceptable policy identifier.

```
policyConstraints EXTENSION ::= {
  SYNTAX      PolicyConstraintsSyntax
  IDENTIFIED BY id-ce-policyConstraints }

PolicyConstraintsSyntax ::= SEQUENCE {
  requireExplicitPolicy   [0] SkipCerts OPTIONAL,
  inhibitPolicyMapping    [1] SkipCerts OPTIONAL }

SkipCerts ::= INTEGER (0..MAX)
```

### 12.11.2.13 Extended key usage field

This field is an extensions of the previous field keyUsage. It is used to define more purposes for which the certified public key may be used.

```
extKeyUsage EXTENSION ::= {
  SYNTAX      SEQUENCE SIZE (1..MAX) OF KeyPurposeId
  IDENTIFIED BY id-ce-extKeyUsage }

KeyPurposeId ::= OBJECT IDENTIFIER
```

#### 12.11.2.14 CRL Distribution points

This field defines how CRL (Certificate revocation list) information can be obtained.

```
cRLDistributionPoints EXTENSION ::= {
  SYNTAX         CRLDistPointsSyntax
  IDENTIFIED BY  id-ce-cRLDistributionPoints }

CRLDistPointsSyntax ::= SEQUENCE SIZE (1..MAX) OF DistributionPoint

DistributionPoint ::= SEQUENCE {
  distributionPoint  [0]  DistributionPointName OPTIONAL,
  reasons            [1]  ReasonFlags OPTIONAL,
  cRLIssuer          [2]  GeneralNames OPTIONAL }

DistributionPointName ::= CHOICE {
  fullName                  [0]  GeneralNames,
  nameRelativeToCRLIssuer   [1]  RelativeDistinguishedName }

ReasonFlags ::= BIT STRING {
  unused                (0),
  keyCompromise         (1),
  cACompromise          (2),
  affiliationChanged    (3),
  superseded            (4),
  cessationOfOperation  (5),
  certificateHold       (6) }
```

## 12.12 Platform minima

For GEM terminal specifications that do not adopt the functional equivalent named "application authentication" as defined in clause 15.6, "Functional equivalents", GEM terminal specifications may specify different Platform minima than what is described in this clause.

NOTE 1: When application authentication mechanism defined in GEM terminal specification does not require a root certificate in the terminal, the platform minima concerning the root certificate described in this clause do not apply.

NOTE 2: CRL and RCMM related minima do not apply for Packaged Media profile.

GEM platform hardware is required to support the following minimum sizes to support the GEM security model:

- A value of 2 for N in clause 12.9.2.2, "Security of RCMM".
- A minimum depth of 5 levels in the certificate chain.
- A minimum of 8 CRLs.
- A minimum of 10 entries per CRL.
- A minimum of 3 root certificates (regardless of the number of underlying root certificate authorities).
- A minimum of 3 root certificates for testing (see clause 12.9.3, "Test certificates").

## 12.13 Plug-ins

The security aspects for the plug-in are the same as for any other DVB-J application. The security aspects of the delegated application cannot exceed the permissions available to the plug-in.

So, the plug-in will normally request the set of permissions required to support a delegated application. For a delegated application that follows the security model defined in the present document, the plug-in shall implement a security system to grant a sub-set of these permissions to the delegated application (see annex AF, "Plug-in APIs"). For a delegated application whose security model does not follow the present document, the plug-in can optionally implement a private security system to manage a sub-set of these permissions to the delegated application.

# 12.14 Applications loaded from an interaction channel

Same as for applications delivered over the broadcast channel except where specified otherwise below.

## 12.14.1 Permission for application loading from the return channel

The following policy covers loading of applications over the interaction channel. It includes both initial loading to start and application and dynamic loading during the lifetime of an application. It covers both the code of an application and supporting data however loaded.

- The user enters a locator specifying the location of the AIT File into the navigator:

  In this case the user implicitly gives permission for the connection to retrieve the AIT File when they supply the locator.

  Applications launched by this mechanism inherit limited rights to use the return channel. Where the protocol ID is "3" the application gets rights to access the files specified by the transport protocol descriptor.

  Additional permissions can be granted by the normal permissions mechanism.

- A service has a broadcast Application Description and the transport protocol for the application is type "3" so ALL files are provided by the interaction channel.

  If the application launch is initiated by another application using the listing and launching API and it is necessary to open the interaction channel before the permissions of the launched application are evident then the permissions of the launching application are considered.

  If the application launch is initiated from service selection (autostart application on a service) then the behaviour depends on policy in the GEM terminal.

NOTE: This behaviour may, for example, be controlled by a terminal user preference.

- A service has a broadcast Application Description but a proportion of the files forming the application are provided via the interaction channel.

  If the permission request file and the authentication infrastructure are available without opening the interaction channel then these can be used to determine if opening the interaction channel is suitably authorized.

- An application selects a service using a locator pointing to an AIT File delivered over the return channel to perform service selection

  In this case the service selection succeeds only if the application has permission to do a service selection and the appropriate return channel access permission, as described in clause 12.6.2.12.2, "Signed applications".

If the permission request file is not available then, as above, the behaviour depends on policy in the GEM terminal.

# 12.15 Stored applications

At time of execution of an application, all files in stored applications must have been fully authenticated as defined by clause 12.4.4, "Integration". The extent to which this authentication is performed at time of storage and time of execution differs as described below. In all cases, the certificates used must be valid at the time of execution (e.g. they have not been expired or revoked, and the root certificates they use have not been removed).

The normal constraints on authentication, including clause 11.2.3, "Class Loading", shall be respected when executing an application. (This applies even if some class files are stored but others are not.) This may lead to some stored class files failing authentication even if all certificates are still valid.

Files that fail authentication shall be treated in the same manner as files that fail authentication from a carousel. If a stored file fails authentication the implementation shall not fall back to loading that file from carousel.

### 12.15.1 Stand-alone stored applications

At time of application storage the stored files must be authenticated as if they were being loaded by the application. At this stage the terminal shall not treat class files specially.

In the case that a certificate that was validated during the store process has been revoked or has expired, any additional certificate chains that were not validated at storage time shall now be fully validated using the same rules as would apply to such a file from broadcast - i.e. clause 12.4.3.6, "Authentication rules".

If stand-alone stored applications access broadcast files (e.g. using the org.dvb.dsmcc API), the full requirements of clause 12.4.4, "Integration" shall apply to those files.

At time of execution, it shall be checked that any certificates used to authenticate credentials have neither expired nor been revoked.

### 12.15.2 Cached applications

It is implementation dependent how much authentication is performed at the time of storage.

NOTE 1: As an optimisation, implementations may calculate hash values of files at the time they are stored.

NOTE 2: Cached applications which will run without access to an object carousel for the stored files (e.g. not_launchable_from_broadcast set to '1') may be authenticated as described for stand-alone stored applications. The time at which the various steps of the authentication process are performed is not visible to applications.

## 12.16 Void

## 12.17 Authentication of unbound applications

All unbound applications shall be signed and successfully authenticated. Unsigned applications or signed applications that fail authentication shall not be started as unbound applications, even if they have been successfully installed as such.

NOTE: This functionality is not mandatory in any profile of GEM.

## 12.18 Authentication of privileged applications

Privileged applications shall be subject to additional authentication beyond the normal authentication mechanisms. GEM intentionally does not require any specific additional authentication mechanisms.

NOTE 1: System software download may be used in two ways. Privileged applications may be distributed as part of a system software download and authenticated as part of the authentication of that download.

NOTE 2: Another possible additional authentication mechanism could be to require such applications to be signed by an additional certificate whose availability is very tightly controlled. See clause 14.2.2.2.2, "Signed applications" of OCAP [5] for an example of such a mechanism. Such additional certificates could be distributed as part of authenticated system software download.

NOTE 3: Additional mechanisms (e.g. not involving system software download) may be defined in future versions of GEM or by other specifications extending GEM.

NOTE 4: This functionality is not mandatory in any profile of GEM.

# 13 Graphics reference model

## 13.1 Introduction

The current document provides tools to control the positioning of: video on an output device, interface components such as buttons and lists, as well as raw graphical primitives.

Each screen connected to a GEM has three planes which are, from back to front, a background plane, a video plane and a graphics plane.

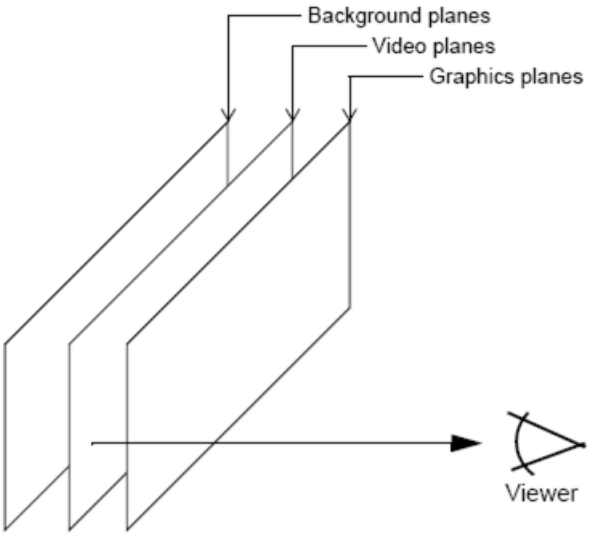



**Figure 16: Illustration of the different types of display planes**

The behaviour of the subtitle plane varies between implementations. API facilities are provided to allow applications to make the behaviour predictable. See clause 13.5, "Subtitles".

An application is provided with a contiguous rectangular region of the graphics plane in which it can draw (see clause 13.3.3, "HAVi devices and AWT components"). An application can place video, interface elements and graphics inside its rectangle on the graphic plane.

An application can also control video outside of the AWT hierarchy on the video plane, and place still images, video drips or solid colour in the background plane.

The GEM specification enables terminals to support multiple applications at any one time, each of which can have a sub area of the screen to which it can draw. The specification enables the areas to overlap. However, the minimum required support for these features is profile specific. See annex G, "Minimum platform capabilities".

NOTE: The mapping between planes in this reference model and planes in the hardware used in a GEM terminal is implementation dependent. In particular, it is certainly allowed to use planes which the terminal hardware considers to be "video" as what GEM considers to be "background" for the purposes of displaying MPEG stills and video drips in what GEM considers to be the background.

### 13.1.1 Interapplication interaction

If the presentations of different applications overlap, the areas obscured by other applications are clipped. Therefore where an application is translucent it will be blended with the video or background image behind it rather than being blended with another application.

# 13.2 General Issues

## 13.2.1 Coordinate Spaces

The present document includes a number of coordinate systems for different purposes and includes the means to transform between these as needed:

- Input video space.

  This considers post upsampling MPEG pixels.

- Device space.

  Logical pixels in the various display devices. There may be different device spaces for the various device types (e.g. video and graphics).

- Normalized screen space.

  Normalized coordinates relative to the output (`HScreen`).

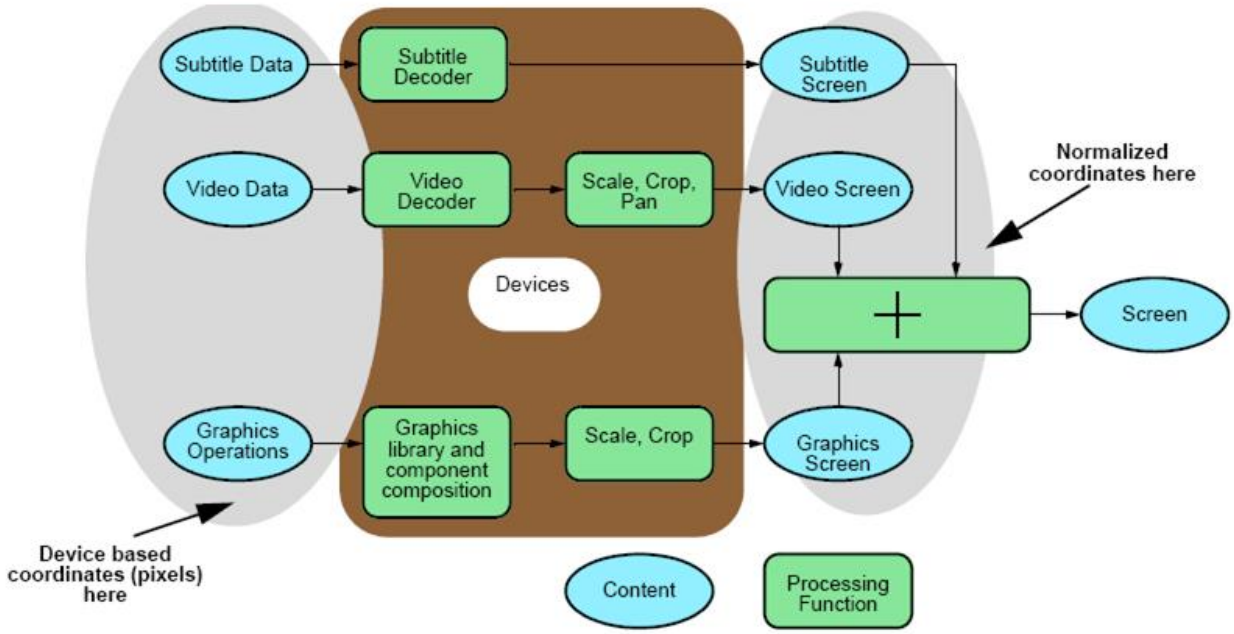



**Figure 17**

Various different interfaces provide access to different parts of the graphics and video systems using these coordinate spaces.

### 13.2.1.1 Normalized screen space

A normalized screen coordinate system supports references to positions and sizes in the video output from a GEM device without reference to any form of pixels.

This coordinate system describes the top left corner of the screen as {0, 0} and the bottom right corner of the screen as {1, 1}. This coordinate system is used in the positioning of graphics and video on the screen through the a number of classes in the `org.havi.ui` package and the JMF control `org.dvb.media.VideoPresentationControl`.

The normalized coordinate system is given through the `HScreen`.

### 13.2.1.2 User space

The coordinate space for graphics used in java.awt is defined by applications through the creation of an `HGraphicsDevice`. The root container, an instance of the class `org.havi.ui.HScene` can be placed within the normalized coordinate system. The `HSceneTemplate` class allows applications to express requirements for their top level container. Instances of this class are used by `HSceneFactory` to return instances of an `HScene`.

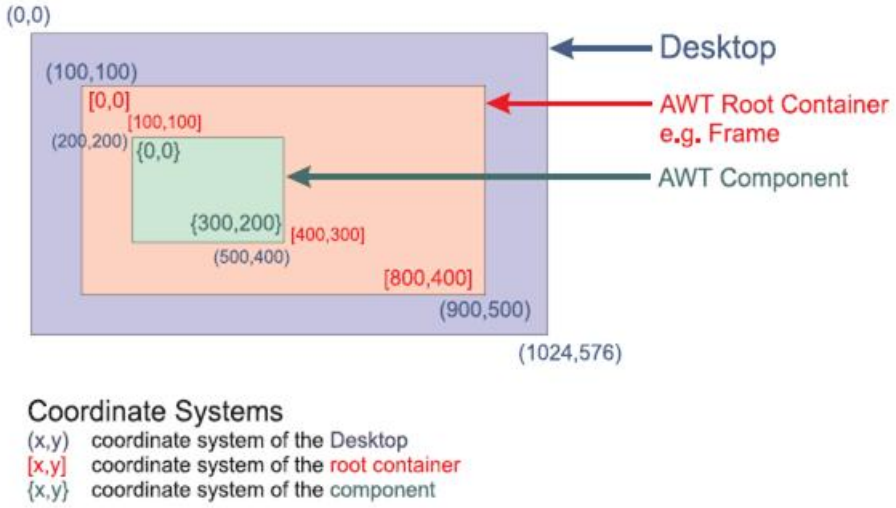



**Figure 18: AWT Coordinate system in a computer environment**

Figure 18 shows the AWT coordinate system in a normal computer environment. In a GEM device the `HGraphicsDevice` can be thought of being the desktop and the `HScene` as being the AWT RootContainer. Thus an `HScene` is placed within the coordinate system of the `HGraphicsDevice`. An `HScene` itself defines a new coordinate system which pixels are aligned to those of the `HGraphicsDevice` but the origin is translated (see figure 18).

The mapping between the user space and the normalized coordinate system is done given the size and location of the `HGraphicsDevice` in the normalized coordinate system and the resolution of the `HGraphicsDevice` in pixel.

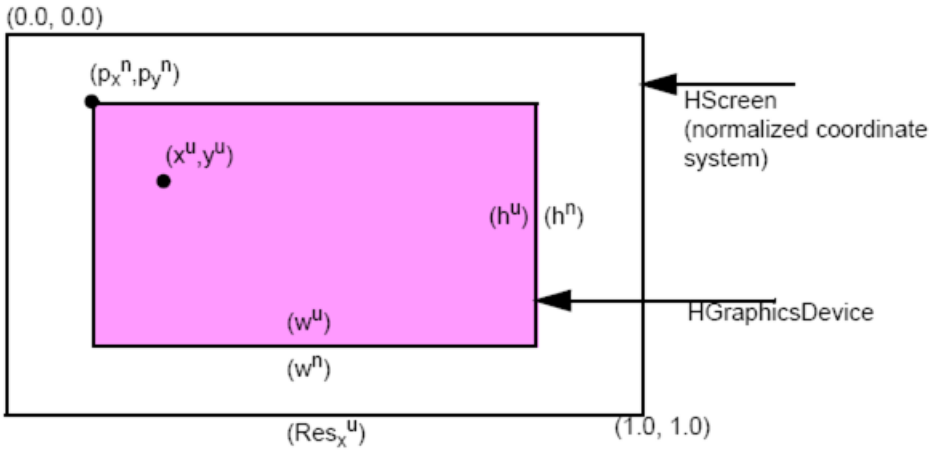



**Figure 19: Conversion between normalized and user coordinate systems**

$(x^n, y^n)$ = normalized coordinate system of `HScreen`.

$(x^u, y^u)$ = coordinate system of the `HGraphicsDevice` in pixels.

$x^{vu}$ = "virtual" user coordinate system in pixels relative to the `HScreen`.

Assume the HGraphicsDevice has the origin of ($p_x^n$, $p_y^n$) with a dimension of ($w^n$, $h^n$) and has a pixel resolution of $w^u$ x $h^u$ than the point ($x^u$, $y^u$) in normalized coordinates is given by:

$$x^n = \frac{x^{vu}}{\mathrm{Res}_x{}^{vu}} \qquad y^n = \frac{y^{vu}}{\mathrm{Res}_y{}^{vu}}$$

where:

$$\mathrm{Res}_x{}^{vu} = \frac{w^u}{w^n} \qquad \mathrm{Res}_y{}^{vu} = \frac{h^u}{h^n}$$

is the virtual resolution of the HScreen in (sub)pixels and:

$$x^{vu} = p_x{}^{vu} + x^u \qquad y^{vu} = p_y{}^{vu} + y^u$$

is the point (x, y) in the virtual coordinate space with:

$$p_x{}^{vu} = p_x{}^n \cdot \mathrm{Res}_x{}^{vu} \qquad p_y{}^{vu} = p_y{}^n \cdot \mathrm{Res}_y{}^{vu}$$

being the location of the HGraphicsDevice in the virtual coordinate system.

Given the Point ($x^n$, $y^n$) the point ($x^u$, $y^u$) is given by:

$$x^u = floor[(x^n - p_x{}^n) \cdot \mathrm{Res}_x{}^{vu} + 0.5]$$

and

$$y^u = floor[(y^n - p_y{}^n) \cdot \mathrm{Res}_y{}^{vu} + 0.5]$$

Given a HGraphicsDevice which is full screen this simplifies to:

$$\mathrm{Res}_x{}^{vu} = w^u \quad \text{and} \quad \mathrm{Res}_y{}^{vu} = h^u \quad , \quad x^{vu} = x^u \quad \text{and} \quad y^{vu} = y^u$$

thus:

$$x^n = \frac{x^u}{w^u} \qquad y^n = \frac{y^u}{h^u}$$

and

$$x^u = x^n \cdot w^u \qquad y^n = y^n \cdot h^u$$

Within the above calculations precision equivalent to a float shall be used. Conversion to integer shall only be used when the result is a point in a pixel oriented co-ordinate space (e.g. user space).

The resolution of an HGraphicsDevice is specified using the constant HScreenConfigTemplate.PIXEL_RESOLUTION with a Dimension on the setPreference(int,Object,int) method.

The constants HSceneTemplate.SCENE_PIXEL_LOCATION and HSceneTemplate.SCENE_PIXEL_DIMENSION allows applications to define the position and size of an HScene in the coordinate system of the HGraphicsDevice (in pixels). The constants HSceneTemplate.SCENE_SCREEN_LOCATION and HSceneTemplate.SCENE_SCREEN_DIMENSION allows applications to define the position and size which the HScene should occupy on the screen in normalized coordinates.

The combination of these defines the transformation between graphics pixels and the video output from the GEM device concerned. Only a limited set of transformations are required to be supported.

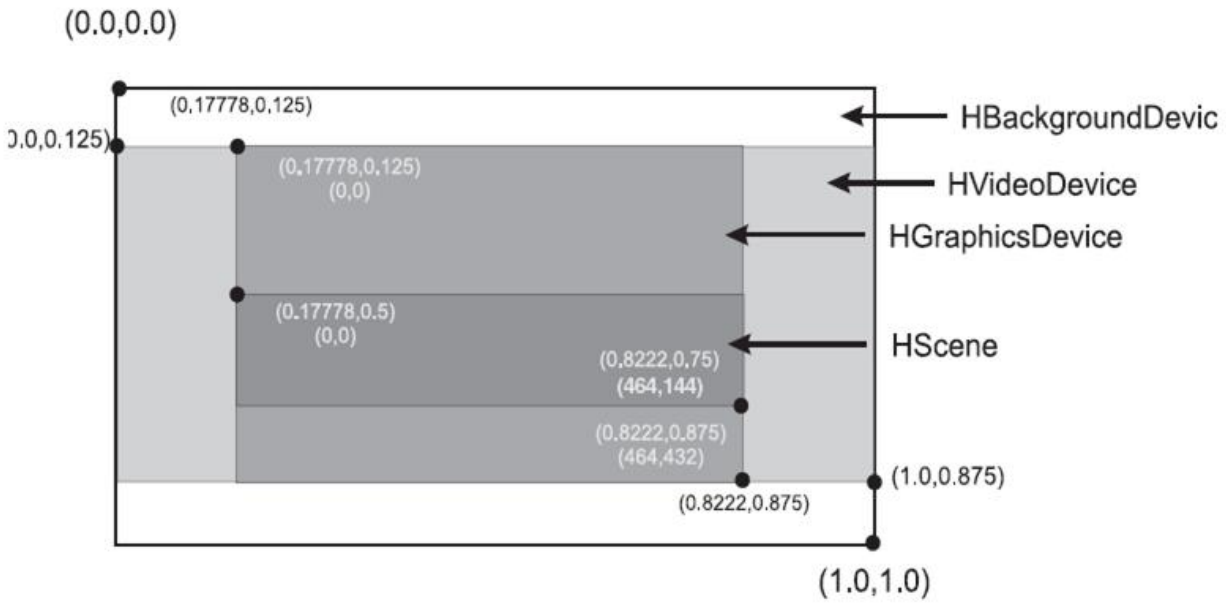



**Figure 20: Possible configuration of HAVi Devices**

Figure 20 shows a possible configuration of HAVi Devices. The `HBackgroundDevice` is configured to be full screen, the `HVideoDevice` to cover the area of (0.0, 0.125) to (1.0, 0.875).

When positioning video implementations may snap the video position to an adjacent line vertically (for example, to accommodate video and display field order) or an adjacent pixel horizontally (for example, to accommodate display chroma structure). The direction of "snapping" shall always be to minimize the error relative to the requested coordinate.

In figure 20 the `HGraphicsDevice` is configured to cover the area (0.17, 0.125) to (0.82, 0.875) with a pixel resolution of 464 x 432.

The `HScene` can be configured by setting the location of its origin and by defining its dimensions. The location of the `HScenes` origin can be specified by setting the preference `HSceneTemplate.SCENE_PIXEL_LOCATION` with a `Point` of (0,216) or setting the preference `HSceneTemplate.SCENE_SCREEN_LOCATION` with a `HScreenPoint` of (0.17, 0.5). The dimensions can be configured by setting the preference `HSceneTemplate.SCENE_PIXEL_DIMENSION` with a `Dimension` of (464, 144) or setting the preference `HSceneTemplate.SCENE_SCREEN_DIMENSION` with a `HScreenDimension` of (0.64, 0.25).

In some implementations and/or configurations pixels in the `HGraphicsDevice` may not correspond to discrete physical pixels in the actual display device. For example, this may be the case when the `HGraphicsDevice` is emulated.

### 13.2.1.3 Pixel Aspect Ratio

A pixel orientated coordinate system does not say anything about the pixel aspect ratio. The pixel aspect ratio ($AR_x^{pixel}$ / $AR_y^{pixel}$) is defined by the aspect ratio of the display ($AR_x^{display}$ / $AR_y^{display}$, 4:3 or 16:9), the area that is covered by the `HGraphicsDevice` ($w^n$, $h^n$) and the pixel resolution of the `HGraphicsDevice` ($w^u$, $h^u$) (see figure 21).

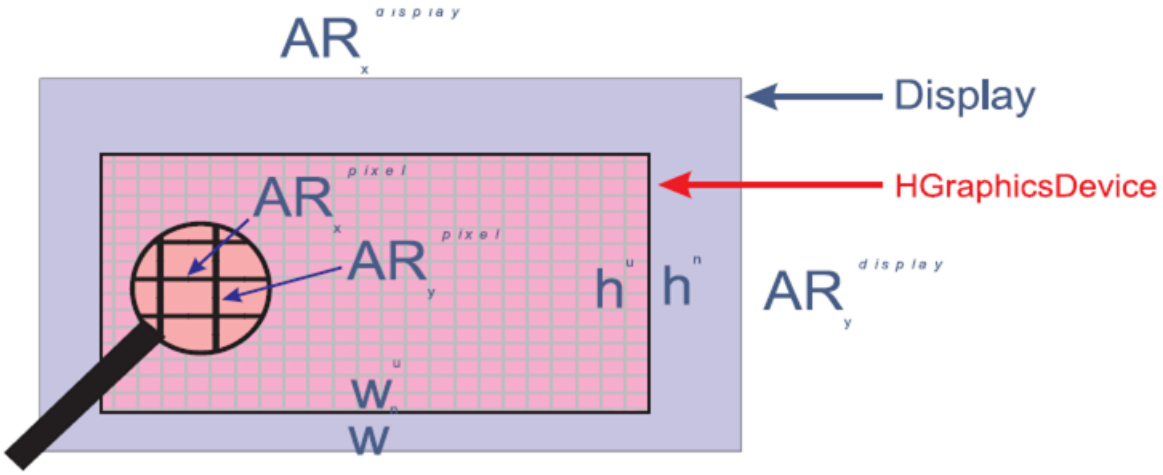



**Figure 21: Calculating the Pixel Aspect Ratio**

Table 58 shows typical resolutions of a full screen `HGraphicsDevice` and the corresponding pixel aspect ratios for different display aspect ratios. The supported resolutions are defined in annex G, "Minimum platform capabilities". Table 58 is an informative listing of typical resolutions and their pixel aspect ratios. This may not apply in all regions, e.g. regions with NTSC standard definition.

**Table 58: Typical Resolutions and their pixel aspect ratio (informative)**

| | 4:3 Display | 16:9 Display |
|---|---|---|
| **Resolution for full screen HGraphicsDevice** | **Pixel Aspect Ratio** | **Pixel Aspect Ratio** |
| **640 x 480** | 1:1 | 4:3 |
| **720 x 480** | 8:9 | 32:27 |
| **720 x 576** | 48/45 | 64/45 |
| **768 x 576** | 1:1 | 48:36 |
| **854 x 480** | 3:4 | 1:1 |
| **1 024 x 576** | 36:48 | 1:1 |

#### 13.2.1.4 Video space

The coordinate space for video is that defined by the input video signal after any scaling required by the platform (e.g. that required by ETR154). Scaling and clipping of video can be achieved using the JMF control `org.dvb.media.VideoPresentationControl`. The JMF control `VideoFormatControl` allows applications to query the various transformations being performed on video as part of its decoding and presentation.

### 13.2.2 Dependencies between HAVi screen device configurations (informative)

#### 13.2.2.1 Principles

Under some circumstances there are dependencies between configurations of the various HAVi screen devices that are composed together to be output as a single screen. Two example reasons for this are the following:

- The video output signal for a screen must have a single aspect ratio - either 4:3 or 16:9 for standard definition GEM terminals. As a consequence, all HAVi screen devices in that screen must be configured to the same aspect ratio.

- Hardware limitations on memory bandwidth may limit the possibilities of combining HAVi screen device configurations. For example, some devices may be able to support the (optional) graphics resolution of 1 920 x 1 080 only under limited (device specific) conditions, e.g. not simultaneously with the decoding of MPEG-4/AVC video.

- Limitations on scalers may limit the possibilities of combining HAVi screen device configurations. For example, software based scaling solutions may not be able to simultaneously support full screen graphics resolutions of 960 x 540 and 720 x 576 in a single hardware graphics plane.

The concept of sets of HAVi screen device configurations which can be simultaneously selected is supported through the method HScreen.getCoherentScreenConfigurations.

#### 13.2.2.2 Usage examples for standard definition, 625-line markets

If the aspect ratio of the video output signal is 4:3 then the configuration of each HAVi HScreenDevice that contributes to that signal must have a configuration shown in table 82, "Sets of coherent configurations for standard definition - 625-line markets" as having a screen aspect ratio of 4:3. Similarly if the aspect ratio of the video output signal is 16:9 then the configuration of each HAVi HScreenDevice that contributes to that signal must have a configuration shown in table 82, "Sets of coherent configurations for standard definition - 625-line markets" as having a screen aspect ratio of 16:9. DVB-J applications can obtain instances of such coherent configurations by creating HScreenConfigTemplate instances with PIXEL_ASPECT_RATIO set to either 16/15 (for 4:3) or 64/45 (for 16:9) depending on what is required.

#### 13.2.2.3 High definition usage examples

The minimum requirements for support of HD are defined in clause G.1.1.3, "High definition" of the present document. These requirements result in 5 coherent sets of HAVi screen device configurations. These are shown below for 625-line/50 Hz markets. For 525-line/60 Hz markets, 720 x 576, 64/45 and 56/45 should be replaced by the values in table 83 "Sets of coherent configurations for standard definition - 525-line markets".

Set 1 (HD in front of SD, SD is 16:9):

- HGraphicsConfiguration[0] includes PIXEL_RESOLUTION = 1 280 x 720.

- HGraphicsConfiguration[1] includes PIXEL_RESOLUTION = 720 x 576 and PIXEL_ASPECT_RATIO 64/45.

Set 2 (HD in front of SD, SD is 14:9):

- HGraphicsConfiguration[0] includes PIXEL_RESOLUTION = 1 280 x 720.

- HGraphicsConfiguration[1] includes PIXEL_RESOLUTION = 720 x 576 and PIXEL_ASPECT_RATIO 56/45.

Set 3 (SD in front of HD, SD is 16:9):

- HGraphicsConfiguration[0] includes PIXEL_RESOLUTION = 720 x 576 and PIXEL_ASPECT_RATIO 64/45.

- HGraphicsConfiguration[1] includes PIXEL_RESOLUTION = 1 280 x 720.

Set 4 (SD in front of HD, SD is 14:9):

- HGraphicsConfiguration[0] includes PIXEL_RESOLUTION = 720 x 576 and PIXEL_ASPECT_RATIO 56/45.

- HGraphicsConfiguration[1] includes PIXEL_RESOLUTION = 1 280 x 720.

Set 5 (960 x 540):

- HGraphicsConfiguration[0] includes PIXEL_RESOLUTION = 960 x 540.

- HGraphicsConfiguration[1] is not defined in GEM.

The following partial code fragment illustrates how the HAVi APIs can be used by DVB-J applications to ask for set #3 above and then configure the graphics system in that way.

```
HScreen s ;
HGraphicsConfigTemplate front = new HGraphicsConfigTemplate();
HGraphicsConfigTemplate rear = new HGraphicsConfigTemplate();
front.setPreference(
        HScreenConfigTemplate.PIXEL_RESOLUTION, new Dimension(720,576),
        HScreenConfigTemplate.REQUIRED);
front.setPreference(
        HScreenConfigTemplate.PIXEL_ASPECT_RATIO, new Dimension(64,45),
        HScreenConfigTemplate.REQUIRED);
rear.setPreference(
        HScreenConfigTemplate.PIXEL_RESOLUTION, new Dimension(1280,720),
        HScreenConfigTemplate.REQUIRED);
HGraphicsConfigTemplate input[] = { front, rear };
HScreenConfiguration c[] = s.getCoherentScreenConfigurations( input );
s.setCoherentScreenConfigurations(c);
```

For the case of set #5 above, HGraphicsConfiguration[1] is not defined. Some implementations may set the rear graphics plane to the DISABLED HGraphicsConfiguration. Other implementations may support 960 x 540 simultaneously with other graphics resolutions. Applications wishing to select set #5 should not constrain the rear graphics plane to DISABLED otherwise the call to getCoherentScreenConfigurations may fail on more capable GEM terminals.

#### 13.2.2.4 Scalability and extensibility

GEM terminals may support additional HAVi screen devices and configurations beyond the minimum requirements defined in clause G.1.1, "Device capabilities" of the present document. No particular issues arise where the minimum required HAVi screen devices support additional configurations. GEM applications may discover those configurations and, if they are able to reserve the device concerned, attempt to set the configuration. The conditions under which setting such an additional configuration succeeds is not defined by GEM and may be very specific to a particular GEM terminal.

Where extra HAVi screen devices are supported in addition to the minimum requirements, these extra devices shall only appear behind the minimum required HAVi screen devices of the same type. GEM terminal implementations may support extra graphics or video devices in front of the minimum required HAVi screen devices which are not visible to the GEM environment, e.g. for transient UIs relating to device setup, configuration or operation.

## 13.3 Graphics

### 13.3.1 Modelling of the GEM display stack composition

The following clauses describes the theoretical model of the GEM display stack. Unfortunately, certain real world constraints may apply see clause 13.6, "Approximations". The "Graphics, Video and Subtitle pipeline" is illustrated in figure 22.

Figure 22 shows the conceptual model of the "graphical content" pipelines from an applications point of view. There are four sources of graphical content: subtitles, background, video and application graphics, and these four sources are mapped in a controllable way onto single screen. The figure shows the conceptual processing functions, the content stages, and application control points inside the content pipelines.

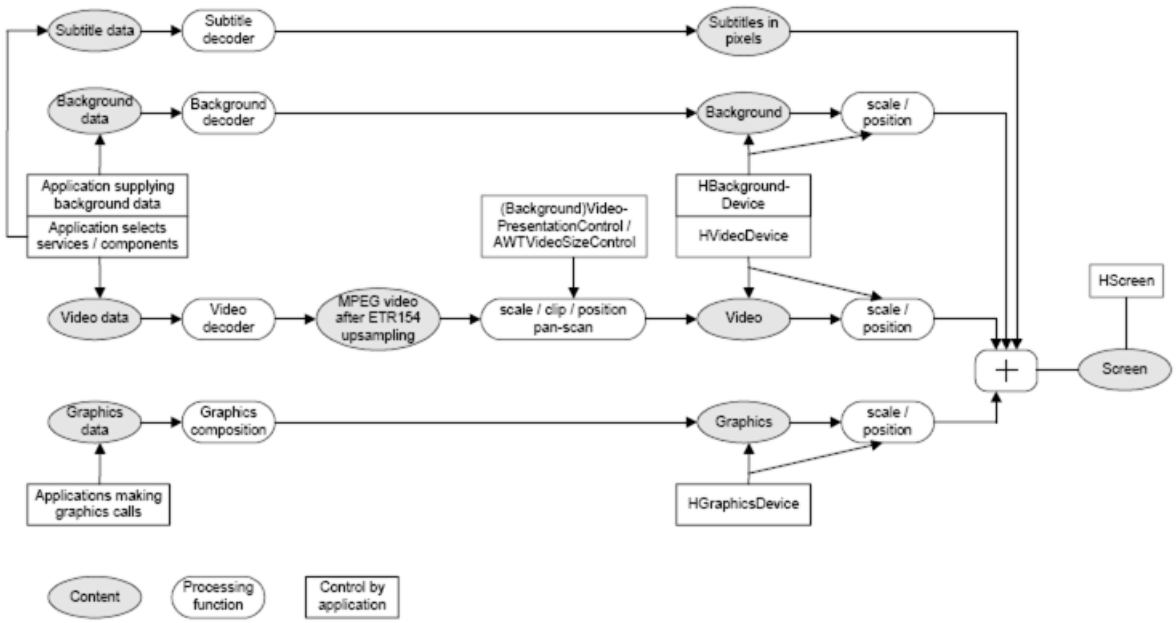



**Figure 22: Graphics, Video and Subtitle pipeline**

The figure shows that applications have full control over the source of the content that enters the different pipelines. Furthermore the application may control clipping, scaling and positioning of the content at different stages in the different pipelines. Of course an application can only apply operations that are supported by the underlying system.

The clipping and positioning of subtitles follows the video. Behaviour of subtitles when video is scaled is platform dependent.

From the application author's point of view the behaviour of the graphics composition process has 3 elements:

- The graphics elements are composed following the traditional graphics model using Porter-Duff rules (see Porter-Duff).The default rule used is the SRC_OVER rule.
- The background and video planes are composed using the Porter-Duff rule SRC_OVER.

NOTE: Only alpha of 0 and 1 is used in the Video planes.

- The results are composed together using the SRC_OVER rule (with the graphics results as the source and the results of the background / video composition as the destination).

This is illustrated in figure 23.

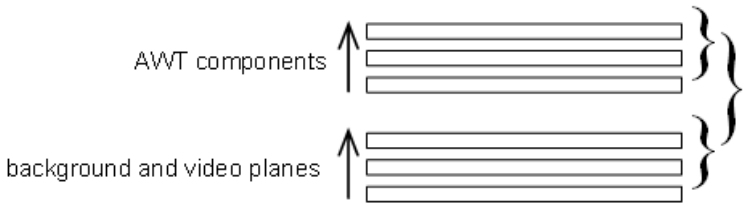



**Figure 23: Overview of AWT / HAVi plane composition**

The HAVi stack has a single full screen `HBackgroundDevice` at the back. In front of this are an ordered set of zero or more `HVideoDevices`. Each of the video devices can occupy the full screen area or part of it. Any area that is not occupied behaves as transparent. The occupied areas (video pixels) are considered to be opaque and will be obscured by any video devices in front of them. Non active video devices are invisible.

Systems that support only a single video device that can display full screen (and possibly other partial screen video devices) should report the full screen capable device as the first (back most) of the video devices.

The composition rules in each group follow the traditional "painter's" algorithm i.e. composing the layers from back to front.

The Xlet AWT Root Component is transparent (as shown in figure 24) so by default the result of the HAVi video and background device composition is the background to any application graphics.

Video already running in the HVideoDevice will keep on playing when an application starts.

The stack is illustrated in figure 24.

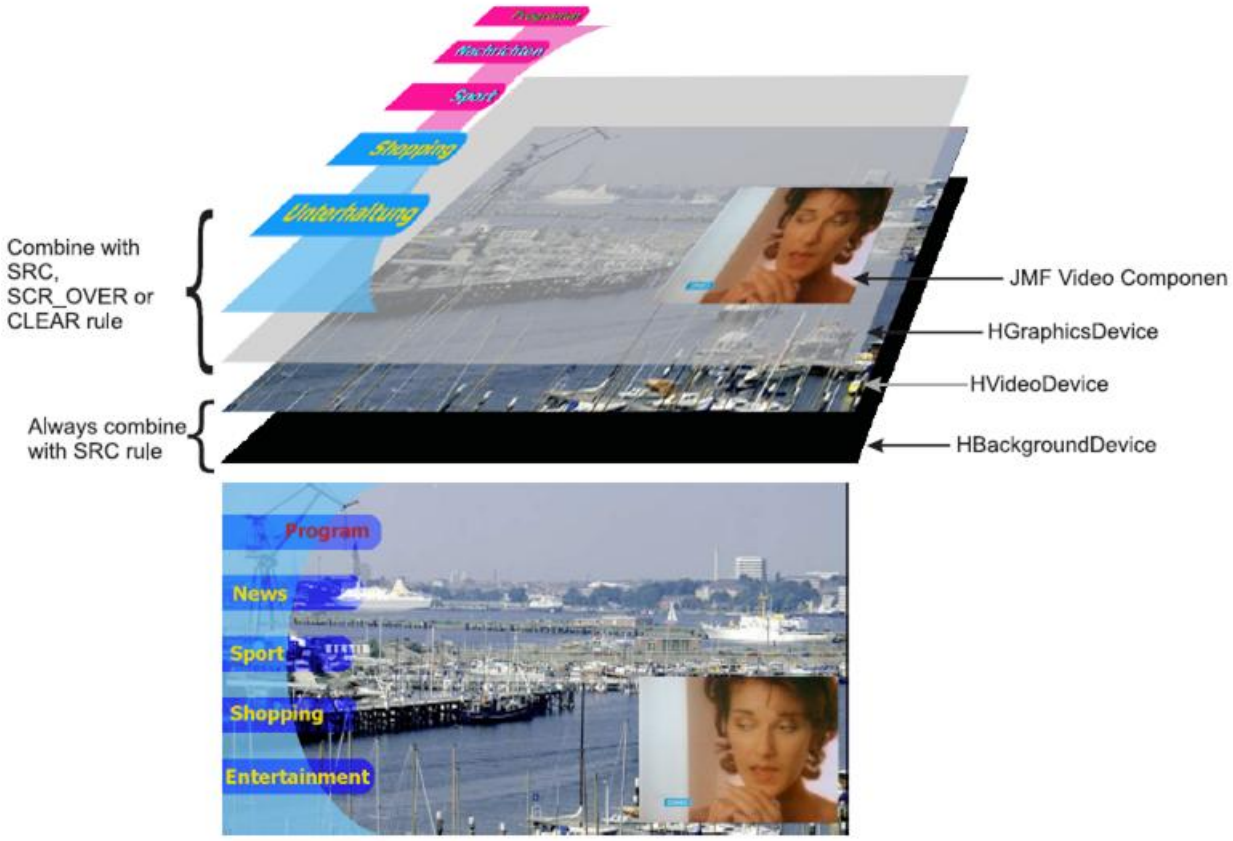



**Figure 24: The GEM display stack illustrated**

## 13.3.2 AWT Reference Model in the GEM

In the AWT all graphics rendering of the lightweight components is done via the `java.awt.Graphics` class. When a component needs to be redrawn it issues a repaint command which gets passed from component to component from top to bottom until a heavyweight component is reached. Although the enhanced and interactive profiles of GEM do not require any heavyweight components to be present, the `HScene` can be thought of being similar to a heavyweight component. Thus in GEM devices the repaint command gets passed until it reaches the root container - the `HScene`. The repaint method of the `HScene` causes a call to this component's `update` method as soon as possible. Before calling the paint method of a child the origin gets translated to the coordinate system of the child and the graphics object passed is clipped to the size of the child (see figure 25).

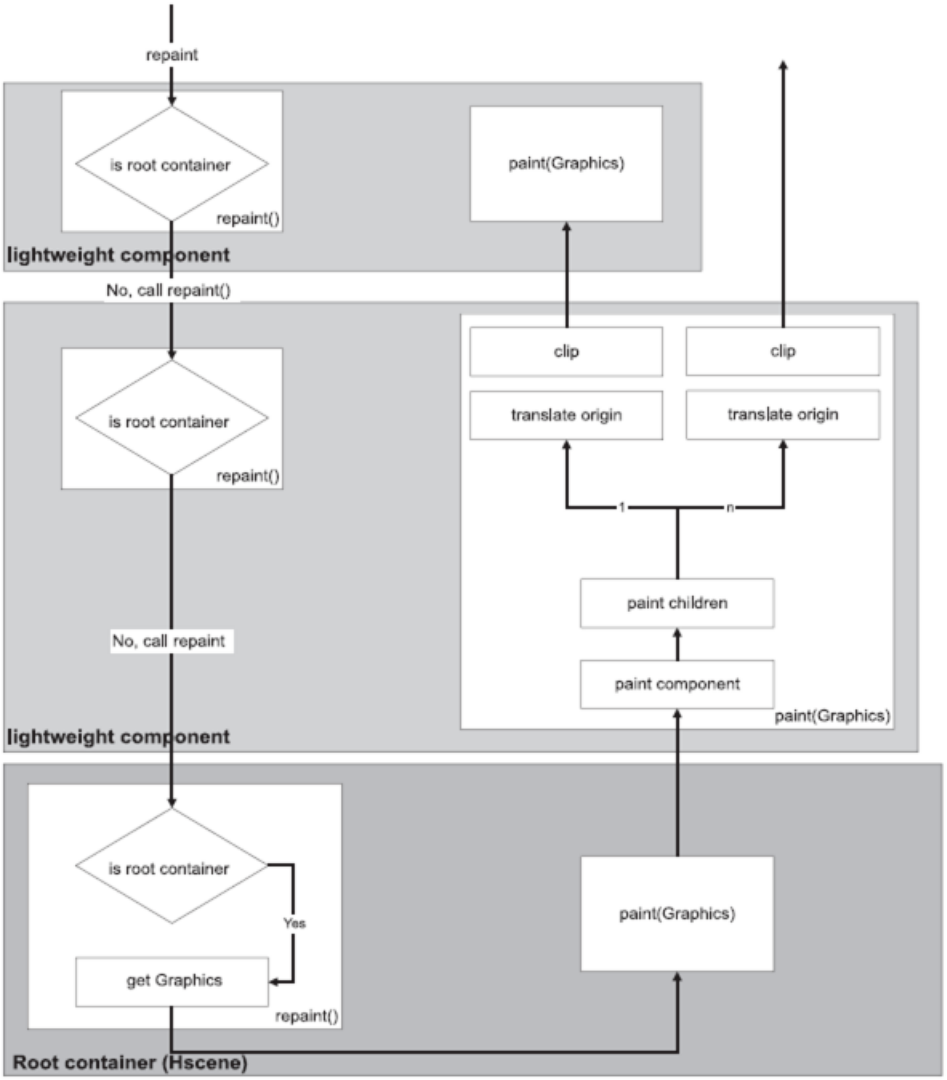



**Figure 25: Repaint model in a GEM application**

## 13.3.3 HAVi devices and AWT components

The top level user interface container for DVB-J applications is represented by the `org.havi.ui.HScene` class. DVB-J applications may obtain an instance of this class from an instance of `org.havi.ui.HSceneFactory`. The `HSceneFactory` class provides a number of methods which may be used to obtain an `HScene` depending on the specific requirements of the DVB-J application. `HScene` is conceptually equivalent to `java.awt.Window` or `java.awt.Frame` however it models the differing graphical environment found in many digital TV receivers. In this environment, there are typically significant constraints on what sizes and positions of top level containers may be possible.

One means for obtaining an `HScene` from an `HSceneFactory` is to populate an `HSceneTemplate` with a set of constraints and then request an `HScene` from the `getBestScene()` method which meets these constraints. Applications wishing to obtain a full screen sized `HScene` may use the `getFullScreenScene()` method specifying a particular graphics device on which the full screen scene should be presented.

GEM terminals are allowed to support only non overlapping `HScenes`. In implementations not supporting overlapping `HScenes` the following behaviour shall be implemented:

- Overlapping at creation or size change through `HSceneFactory`:
  - Returns a null reference if "`REQUIRED`" is used. Returns best effort if "`PREFERRED`" is used.

- Overlapping later when sizes change by awt:
    - Size does not change. Fails silently.

The GEM specification provides a general model for video output devices using the model found in the HAVi specification. A final output video signal is expressed through the `org.havi.ui.HScreen` class and is the result of adding a number of different video components.

- The output of one or more graphics decoders.
- The output of one or more video decoders.
- The output of any special decoders, e.g. a background.

An abstraction of each of these decoders is represented by an instance of a class inheriting from `HScreenDevice` - `HGraphicsDevice`, `HVideoDevice` and `HBackgroundDevice` respectively. Each of these can potentially exist in a range of configurations which are exposed by instances of classes inheriting from `HScreenConfiguration` - `HGraphicsConfiguration`, `HVideoConfiguration` and `HBackgroundConfiguration` respectively. Where devices support multiple configurations, applications may construct and populate templates to define criteria to select between configurations. These templates are classes inheriting from `HScreenConfigTemplate`.

As well as the general features, some of these sub-classes provide support for specific features of the devices concerned. `HGraphicsConfiguration` supports loading of images and listing of fonts specific to that configuration. It also support conversion between various coordinate systems. `HVideoDevice` provides access to the source of the video currently being decoded (a Locator) and provides access to the decoder object for the video currently being decoded (a `javax.media.Player`). The `HBackgroundClass` provides a means to support MPEG I-frames through the `HStillImageBackgroundConfiguration` class.

The method `HScreen.getCoherentScreenConfigurations ( HScreenConfigTemplate[] configs )` allows applications to express a common set of constraints for video, graphics and backgrounds and get back a coherent answer. In `HGraphicsConfigTemplate`, the constant `VIDEO_MIXING` allows applications to request configurations where graphics is super-imposed above video but without any requirement for pixels to be aligned. In `HScreenConfigTemplate`, there are constants to allow applications to ask for configurations as follows:

- `VIDEO_GRAPHICS_PIXEL_ALIGNED` - video and graphics pixels are the same size and aligned.
- `ZERO_VIDEO_IMPACT` - a graphics configuration must not change the existing video configuration.
- `ZERO_GRAPHICS_IMPACT` - a video configuration must not change the existing graphics configuration.

#### 13.3.3.1 Video and graphics pixel aligned

The `VIDEO_GRAPHICS_PIXEL_ALIGNED` relationship between the pixels in the `HGraphicsDevice` and the pixels in the `HVideoDevice` is shown in figure 26.

NOTE: The relationship applies to the pixels in `HVideoDevice` after "Decoder Format Conversion" processing, and the pixels in `HGraphicsDevice`. As a result of this relationship the following constraint holds for the configurations of `HGraphicsDevice` and `HVideoDevice` that have aligned video and graphics pixels:

- The pixel aspect ratio of the pixels in both devices is equal.

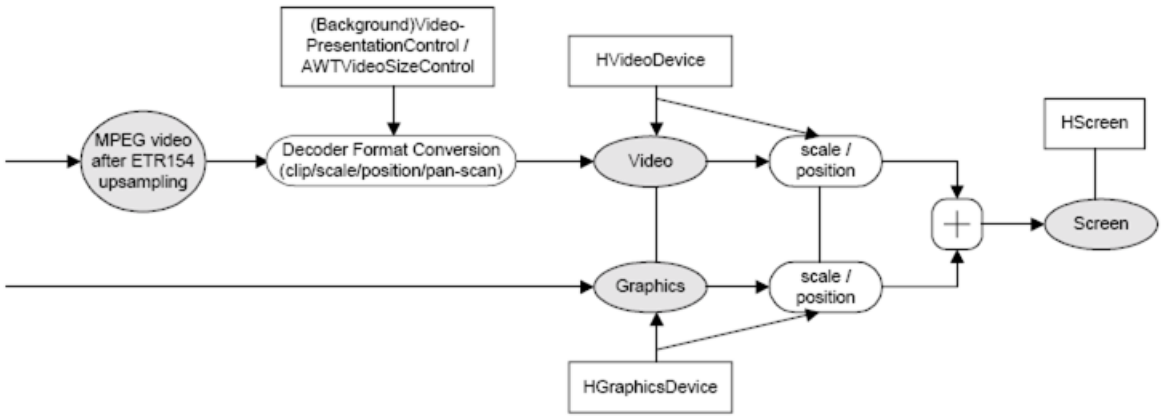



**Figure 26: Video and graphics pixel impact**

#### 13.3.3.2 Zero graphics impact

Constants `ZERO_GRAPHICS_IMPACT` and `ZERO_VIDEO_IMPACT` can be used by applications to prevent changes to the `HGraphicsDevice` or the `HVideoDevice` respectively, in the case that changes are not intended. In general a change of the configuration of the `HGraphicsDevice` may lead to an automatic change of the configuration of, for example, the `HVideoDevice` (if the application is authorized to make this change), because restricted systems may not be able to deal with the two different configurations simultaneously. Therefore an application must specify `ZERO_GRAPHICS_IMPACT` or `ZERO_VIDEO_IMPACT` in the configuration template if it does not want to change the configuration of another device.

### 13.3.4 Composition

#### 13.3.4.1 AWT paint rule

The normal AWT paint rules shall be followed. That is the root container (the `HScene`) is painted and then its components are painted recursively.

The observed *behaviour* shall be such that of each component drawing directly into the root component. Any use of temporary storage or double buffering shall not affect the final result of the AWT composition or its subsequent composition with the result of the video composition.

The process is shown in figure 27. The numbers in the arrows indicate the chronological order of the painting.

NOTE: The root container is painted first.

If a container has multiple components they get painted in the order N, N-1, ,0 where N indicates the order components are added to the parent container. (See Documentation on `java.awt.Container` in PBP 1.1 [4]).

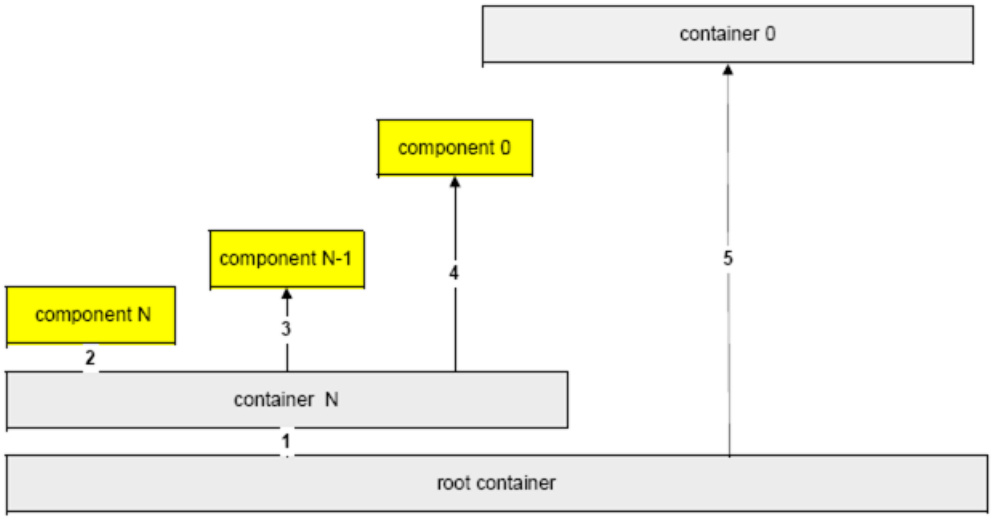



**Figure 27: Chronological order of painting**

## 13.3.5 Composition Rules

### 13.3.5.1 Components generally

By setting an appropriate `DVBAlphaComposite` rule on a `DVBGraphics` and using translucent colours blending may be achieved. By repeated placement of components complex scenes can be described. In figure 28 the background picture (of the harbour) is representative of the video plane. The translucent grey rectangle is drawn into a transparent off-screen buffer using the SRC rule. The red circle is then drawn into this off-screen buffer using one of the 8 most common Porter-Duff operations. When the AWT rendering in the off-screen buffer is complete it is composited with the video using the SRC_OVER rule.

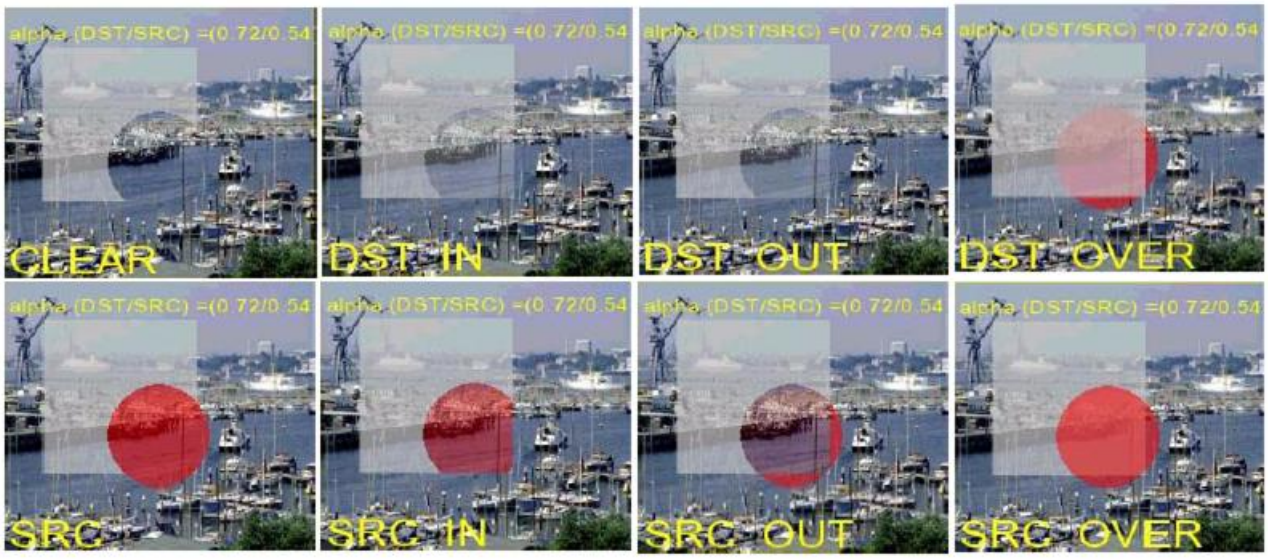



**Figure 28: Summary of the Porter-Duff rules**

## 13.3.6 Extensions to the AWT graphics capabilities

For graphics the GEM specification mainly uses AWT facilities defined in PBP 1.1 [4]. However, certain extensions are made.

In order to allow compositing in GEM applications, the graphics capabilities of PBP 1.1 [4] have been extended by providing the `org.dvb.ui` package.

The package consist of the classes DVBGraphics, DVBAlphaComposite, DVBBufferedImage, DVBColor and one Exception. See annex U, "Extended graphics APIs".

### 13.3.6.1 Graphics Objects in GEM applications

The class org.dvb.ui.DVBGraphics extends the normal java.awt.Graphics class by adding support for alpha compositing.

In a GEM application each platform-created graphics object shall be an instance of org.dvb.ui.DVBGraphics.

DVBGraphics and DVBAlphaComposite has a principle support of 8 different Porter-Duff compositing rules but not all rules have to be supported for all DVBGraphics Objects.

Different DVBGraphics Object could support different compositing rules. E.g. a DVBGraphics Object created using a DVBBufferedImage with an image type of DVBBufferedImage.TYPE_ADVANCED supports all 8 specified Porter-Duff rules, TYPE_BASIC supports only SRC, CLEAR, SRC_OVER. Application can query the available compositing rules using DVBGraphics.getAvailableCompositingRules(). When setting an unsupported compositing rule a org.dvb.ui.UnsupportedDrawingOperationException will be thrown. All images whose getGraphics method returns a non-null object shall support alpha.

The supported compositing rules are defined in annex G, "Minimum platform capabilities".

The default compositing rule used by all graphics objects is SRC_OVER.

### 13.3.6.2 Buffered Image

The class DVBBufferedImage in the package org.dvb.ui adds the support for an accessible, transparent Image which can be used e.g. for off screen buffers. Two different platform dependent sample models are supported by DVBBufferedImage. When doing compositing in the TYPE_BASE sample model approximations may be applied (using SRC instead of SRC_OVER, see figure 36 or by approximating the alpha, see annex G, "Minimum platform capabilities") while in the TYPE_ADVANCED sample model the compositing rules set by the program will be used.

TYPE_ADVANCED is always a direct colour model while TYPE_BASE can be a CLUT based colour model.

### 13.3.6.3 DVBColor

NOTE: The general philosophy of this class has been to imitate the features of JDK 1.2 dealing with colour.

Unless explicitly specified otherwise, the internal implementation of the platform (e.g. java.awt) shall use the org.dvb.ui.DVBColor class instead of the java.awt.Color where technically possible (e.g. not the constructor for java.awt.Color). The class signatures shall not change. For example, where a method is specified to return java.awt.Color it shall return an instance of org.dvb.ui.DVBColor.

#### 13.3.6.3.1 Modified packed colour representation

The most significant byte of the integer representation of an RGB colour is defined to hold an alpha value as is illustrated in figure 29. This data type is used in various of the constructors and methods described in the API documentation. It is referred to in the API documentation as TYPE_INT_ARGB.

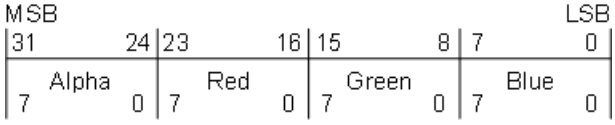



**Figure 29: GEM colour format (TYPE_INT_ARGB)**

### 13.3.7 14:9 Aspect Ratio Support

Support for the 14:9 aspect ratio is optional in GEM terminal specifications.

As well as the real display aspect ratios of 4:3 and 16:9, GEM defines a 14:9 aspect ratio. When this is in effect, all text will be subject to moderate aspect ratio distortion. However, broadcasters will be able to predict text flow without having to author specifically for each display type.

All `HGraphicsDevices` must be `HEmulatedGraphicsDevices` capable of emulating a14:9 aspect ratio on their actual aspect ratio at the time.

NOTE 1: Applications can still set the configuration of an `HEmulatedGraphicsDevice` to an `HGraphicsConfiguration` and have this be supported without emulation.

Where the current configuration of an `HEmulatedGraphicsDevice` has a 14:9 aspect ratio and the device is not reserved by any GEM application, the GEM terminal shall automatically maintain this 14:9 aspect ratio if the real underlying aspect ratio changes between 4:3 and 16:9 or vice-versa.

NOTE 2: The `HGraphicsConfiguration` observed by an application when it starts running may be unpredictable since there may be an already running GEM application which has reserved the `HGraphicsDevice` and set the `HGraphicsConfiguration`.

Setting the configuration of an `HGraphicsDevice` to an `HEmulatedGraphicsConfiguration` supporting a 14:9 display shall only impact the transformation of outline fonts into display pixels. Display pixels shall remain mapped 1:1 onto the actual pixels of the `HGraphicsDevice`.

### 13.3.8 Interaction between graphics from GEM and resident applications

If application graphics are wholly or partly obscured by graphics from resident applications (e.g. the GEM navigator), then either:

a) the resident application shall restrict itself to the use of opaque colours; or

b) the composition between the graphics of the resident application and those of GEM applications shall be done such that it does not result in video pixels that are obscured by opaque graphics pixels from a GEM application becoming visible.

Implementations that cannot meet one of these requirements shall remove the GEM application graphics from the screen when the graphics of resident applications are visible as described in clause 9.10, "Lifecycle interactions between GEM and resident applications".

## 13.4 Video

### 13.4.1 Component-based players and background players

Video is received by a GEM application as an MPEG sequence of compressed frames, each of which contains a large number of picture elements (pels) arranged in rows and columns. The GEM application decodes the MPEG stream to a presented stream which may be placed either in the video plane, or in a component in the graphics plane.

These two different ways of presenting video result in having two different kinds of JMF players, a background JMF player and a component-based JMF player.

A component-based JMF player plays video inside an AWT component, and the video inside that component is positioned and scaled by positioning and resizing the component. The video is always scaled to the full size of the component (or, if the terminal is not capable of scaling and positioning video this accurately, the video may extend outside the bounds of the component by up to one pixel in each direction [above, below, left and right]). This approximation shall not affect the position and size of the AWT component itself and hence shall not be visible to applications via the usual AWT methods).

Support for background players is mandatory for broadcast streaming formats (see clause 7.2, "Media streaming formats").

Background JMF players play video in the video plane, outside and independent of any AWT component hierarchy.

When a component-based player is stopped, the last displayed frame shall continue to be displayed until the underlying video decoder resource is re-used. The behaviour when the underlying video decoder resource is re-used is implementation dependent.

When a background player is in the realized state, the video plane concerned shall become transparent to expose what is behind it such as background plane(s). See figure 16 "Illustration of the different types of display planes".

In GEM terminals exposing through the GEM-defined API the simultaneous decoding of more than one broadcast video stream, only a single background player shall be supported at one time. Realising a second JMF player while an existing background player is in use shall result in the second player replacing the first as the background player. Applications wishing the older player to remain visible must convert it to a component-based player and make that visible.

NOTE: GEM does not mandate a particular broadcast streaming format, e.g. Standard Definition 25Hz MPEG-2 Video is not required by GEM. However, as described in clause 7.2.2 a GEM terminal specification has to include some mechanism for delivering audio and video programming in standard definition. For these formats, background JMF players are created.

## 13.4.2 Modelling MPEG decoding and presentation pipeline

Figure 30 illustrates the underlying format conversion control process in a TS 101 154 [2] compliant SD digital receiver. In this model the video decoder produces "full-screen" video from the MPEG data (i.e. the decoder resolves any sub-sampling in the MPEG broadcast). Subsequent "Decoder Format Conversion" adapts this for the display device taking account of:

- Broadcast meta data (aspect ratio, pan/scan and active format description).
- Display knowledge (4:3 or 16:9, and resolution).
- User preferences (e.g. display wide screen in a letterbox).

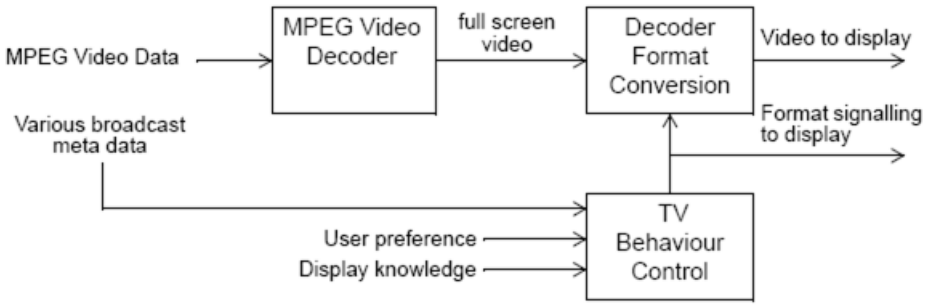



**Figure 30: Format control in TV mode**

If appropriate, the video that is sent to the display is accompanied by the proper WSS or SCART signalling to indicate the format of the video that is being sent.

NOTE: That the display device may do its own Decoder Format Conversion on top of the Decoder Format Conversion that the GEM device does. This is beyond the control of the GEM device.

In this version of the GEM specification, the video decoding process complies with clause 5.1.4 "Luminance resolution" of TS 101 154 [2] and applies up sampling for a defined set of luminance resolutions so that the decoded pictures are displayed at full-screen size. The input video source to JMF shall be the output of this up sampling process as illustrated in figure 31. In addition, there are two alternative sources of control for the "Decoder Format Conversion" process:

- Conventional TV format control behaviour (as in figure 30).
- Application format control behaviour.

The selection between these behaviours is under the control of the application. Before and after the existence of an application the behaviour is that for a conventional TV. During application execution the default behaviour of the JMF player's decoder format conversion shall be the conventional TV behaviour.

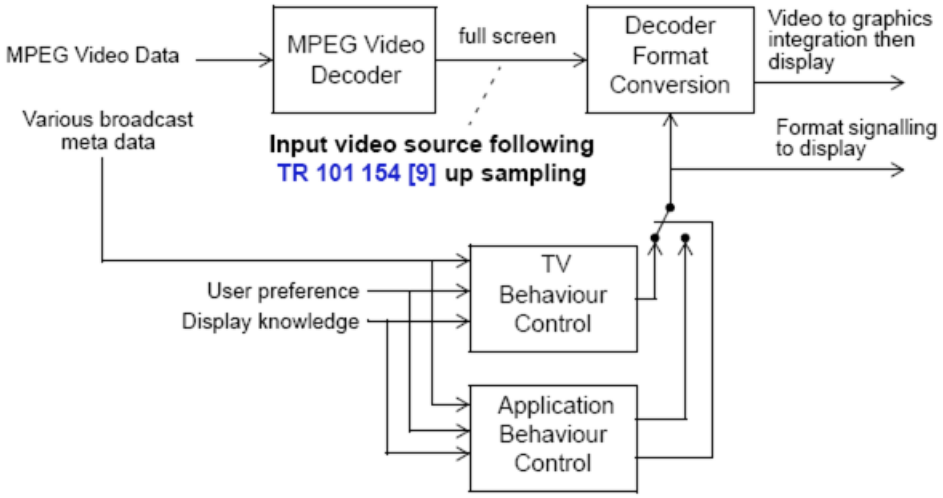



**Figure 31: Format control in the presence of a JMF player**

The Decoder Format Conversion consists of three steps that are performed on the full screen input video, which may have been up-sampled by the MPEG video decoder to become full screen. As illustrated in figure 32, these steps are clipping, scaling and positioning.

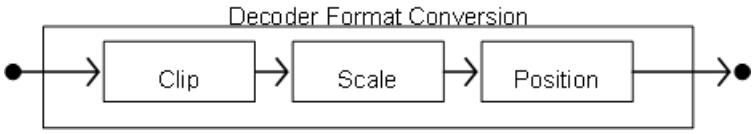



**Figure 32: Reference model for Decoder Format Conversion**

An application can query the implemented capabilities of each step of the Decoder Format Conversion using the org.dvb.media.VideoPresentationControl ("VideoPresentationControl"). For instance, it can query the supported scaling factors.

An application can also set up the Decoder Format Conversion steps atomically by using an org.dvb.media.VideoTransformation object ("VideoTransformation") that encapsulates the clipping, scaling, and positioning parameters. An application can get a number of pre-defined video transformations that correspond with standard Decoder Format Conversions like 16:9 letterboxing in a 4:3 display. It can either use these video transformations directly to set the Decoder Format Conversion, or it can change one or more parameters of the transformation before setting it. The API also offers support for querying the current video transformation.

## 13.4.3 Coordinate Spaces

The input to the Decoder Format Conversion block is always full screen video. If necessary, the MPEG video decoder block performs up-sampling as required by TS 101 154 [2] to get full screen video.

An application expresses the video clipping in terms of the pixel-based coordinate space of the full screen video. For 50 Hz SD video this is always a 720 x 576 raster, and for 60Hz SD video, 720x480 raster.

Positioning of the video is expressed in the normalized coordinate space for background JMF players. For component-based JMF players, the position of the video component is expressed in the pixel-based device coordinate space (i.e. a video component is positioned like any other AWT component).

## 13.4.4 Video components

A scaled portion of a video can be presented as a component within the AWT hierarchy. The controls that influence this presentation are parts of AWT and JMF.

Video components are treated just like any other component. However, it shall be noted that pixels within the video have opaque colours. The mapping to the source video is provided by the video presentation control (see annex N, "Streamed media API extensions") which allows an arbitrary portion of the video to be placed within the AWT hierarchy.

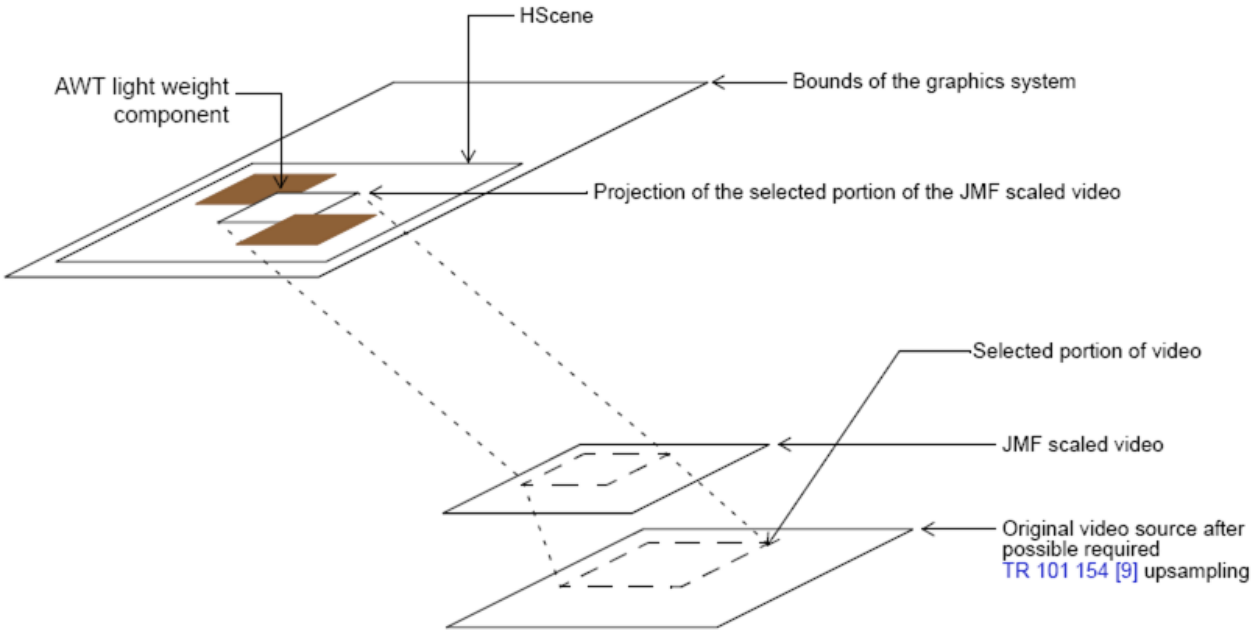



**Figure 33: Introducing video into the AWT component stack**

Graphics drawn over a component-based JMF Player using the SRC or CLEAR rules shall be alpha-blended with the scaled video from the component-based Player, not the background video plane.

Where two or more video components overlap, the z-order of the video shall be determined by the z-order of the video components within the AWT graphics system.

## 13.5 Subtitles

This clause only applies to GEM terminal specifications for which subtitling is available. If subtitling signalling is available, the presentation of subtitles shall follow the model specified here. Further, the introduction to subtitles contained in clause 13.1 and the components related to subtitles in figure 17 clause 13.2.1 apply only to GEM terminal specifications that include support for subtitling.

NOTE 1: US closed captioning is not the same thing as subtitles, thus, in systems where closed captioning is available but subtitling is not, this clause is optional.

NOTE 2: See also clause N.1, "Active Format Definition".

### 13.5.1 Language and presentation setting

The following reference model shows the how the presentation of subtitles is controlled. The selection of the subtitles language depends by default on the user's preferences, the audio language and can be overridden by the application. The end user can set the subtitles on or off where the default depends on the preferences. The application can override all this and switch the subtitles off even if the user has set them on.

Figure 34 illustrates the decision procedure.

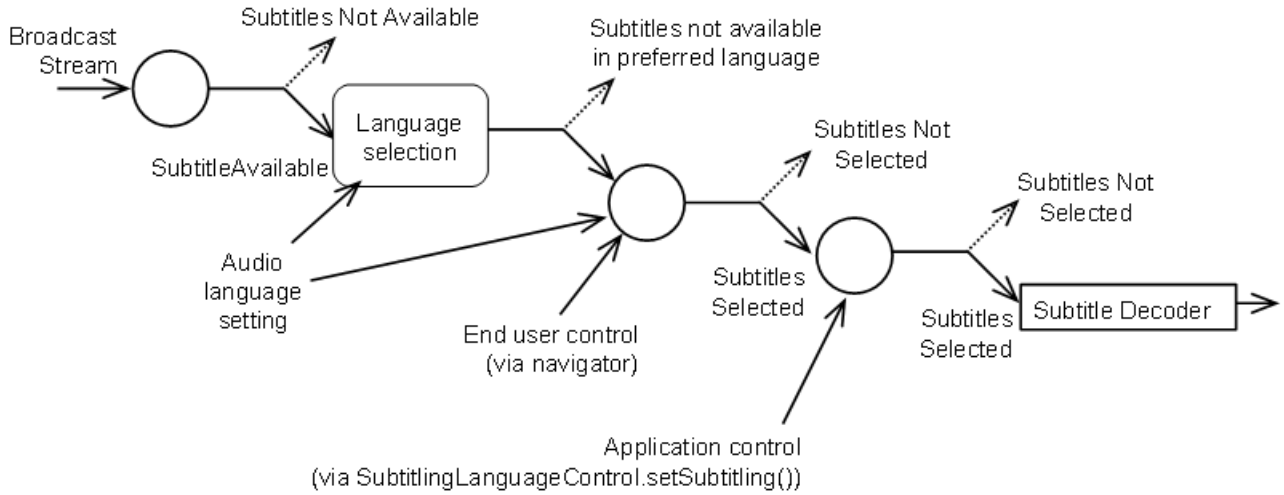



**Figure 34: Determining subtitling language and presentation setting**

The DVB-J API includes a control (see "SubtitlingEventControl") that the application can use to get notified of the state changes in the availability and presentation status of the subtitles. This corresponds to the status of the left hand side input and the right hand side output in the diagram. The language selection and the resulting preferred language are determined using an implementation dependent algorithm. e.g. the platform may support separate language preferences for audio and for subtitles.

The org.davic.media.SubtitlingLanguageControl allows the application to query the currently set subtitling language, override the default language setting and switch the subtitles off or let them be end user controlled. This corresponds to the language selection phase and the application controlled switch in figure 34.

### 13.5.2 Relation to graphics

Ideally, subtitles should be presented on top of the video plane but below the graphics plane(s). However, GEM terminals conforming to the present document are only required to support subtitles in areas where they are not overlapped by application graphics (i.e. by an HScene for a DVB-J application). If any GEM application has an HScene occupying the full area of the screen or has the AWT focus, GEM application graphics should be given preference over subtitles when they overlap otherwise subtitles should be given preference.

See also the description of television viewing mode in clause 11.4.1.4, "Television viewing mode".

The subtitling plane is full screen and allows the subtitles to be positioned anywhere on screen. The positioning of the subtitling texts is part of the subtitling stream content and is fully controlled by the broadcaster.

### 13.5.3 Coordinate Spaces

Subtitles are decoded into a plane with the same coordinate system and position as the `HVideoDevice`.

## 13.6 Approximations

### 13.6.1 Approximations in composition

The GEM specification references the Porter-Duff rules for composition (see Porter-Duff).The minimum set of operations required is defined in (clause G.1.3.1, "Composition rules"), in addition the implementation of the compositions may be approximated as detailed below.

### 13.6.1.1 Implementation of modes

Typical implementations have a hardware blending process that blends the graphics plane over the video using SRC_OVER (see figure 35). This places certain practical limitations on the purity of the display model.

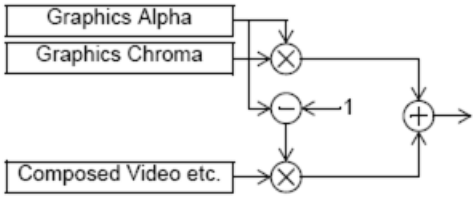



**Figure 35: Typical current technology implementation**

The SRC rule simply requires that the colour and alpha characteristics of the new pixels replace those previously present.

Components obtained from a JMF Player via the `Player.getVisualComponent()` method are treated as having an alpha of 1 so whether SRC or SRC_OVER is used when placing the video component the effect is that the video completely replaces anything previously drawn.

For the purposes of this discussion, MPEG I-frames and video drips shall be considered as video and not as graphics even if particular implementations may implement them using part of the graphics system of their hardware.

#### 13.6.1.1.1 Graphics directly over video

When drawing graphics directly over video:

- The effect of SRC_OVER mode is as expected as the alpha value of the drawn graphics is used to control blending of the graphics with the video.

#### 13.6.1.1.2 Graphics over other graphics

##### 13.6.1.1.2.1 SRC

If graphics are painted over other graphics using SRC mode the result is as expected.

##### 13.6.1.1.2.2 SRC_OVER

If graphics are painted over other graphics using SRC_OVER mode the result will be implementation dependent when $0 < \text{alpha} < 1$. Drawing with colour where alpha = 0 (i.e. fully transparent) shall not cause any effect to the existing pixels. Figure 36 illustrates a variety of implementations of SRC_OVER graphics to graphics blending. One allowed behaviour where the sample model is of `TYPE_BASE` is that defined in PBP 1.1 [4] for those signatures of the `drawImage` methods in `java.awt.Graphics` which do not have a `"bgcolor"` parameter. Specifically, only pixels which are fully opaque are drawn at all. Pixels which are transparent to any extent do not affect whatever pixels are already there.

a) Shows the logically correct result where the green and red areas mix to produce intermediate colours.

This implies a graphics to graphics blending process, it also implies a large gamut in both the chroma and alpha channels. This may not be practical in many early implementations.

The following cases illustrate simplifications of the blending scheme. These should not be considered equally good approximations:

b) Here the graphics alpha is preserved only when it is drawn directly over a video component. When drawn over another graphic the alpha facets of the colours are considered to be 1.

c) Here the source graphics alpha is preserved when it is drawn over a video component even if there is an intermediate semi-transparent graphic. Where the source is over opaque graphics then a graphic to graphic blend is implemented.

d) Here the source graphics alpha becomes the hardware mixing alpha regardless of what has been drawn previously over the MPEG image. This approximation is not permitted for mixing of graphics from the same GEM application. This approximation is permitted for mixing of graphics between GEM applications on GEM terminals that support overlapping HScenes.

e) Here the source graphics alpha becomes the hardware mixing alpha in areas where the alpha is already less than 1. Where the alpha is currently 1 (i.e. over opaque graphics) the alpha remains 1.

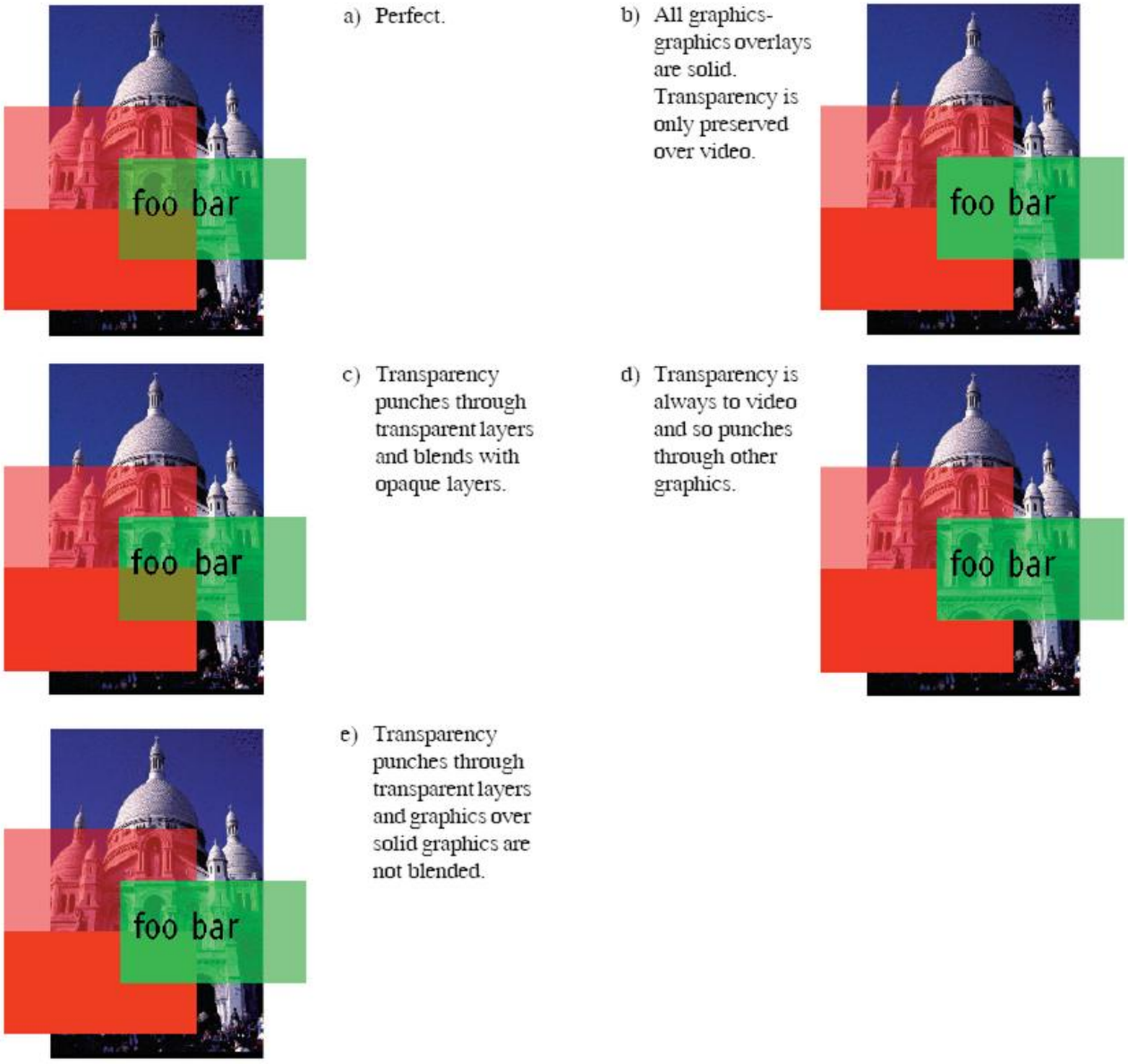



**Figure 36: Implementations of blending**

#### 13.6.1.1.2.3 CLEAR

If graphics are painted over other graphics using the CLEAR mode the result is as expected. This operation is the same as using a source with alpha=0 and the SRC rule.

### 13.6.1.2 Approximation of alpha

The precision of implementation of alpha depends on the `DVBGraphics` object concerned. The minimum requirements are specified in annex G, "Minimum platform capabilities". The actual colour used for a given colour can be queried using `org.dvb.ui.DVBGraphics.getBestColorMatch()`.

#### 13.6.1.3 Approximation of colour

Logically the colour model is a "true colour" one. However, colour approximation is allowed as described in clause G.1.5, "Colour capabilities".

# 14 System integration aspects

## 14.1 Namespace mapping

GEM does not mandate any particular format for locators. Note, however, that clause 14.8, "Locators and content referencing" requires that terminal specifications based on GEM define some text representation for certain entities.

## 14.2 Reserved names

File names starting with the characters "dvb." are reserved for use as "well known" files defined in this or future specifications.

Authors shall not use file names with this form to avoid possible collision with standards defined files.

## 14.3 XML notation

These XML notation rules only apply to XML file formats defined in GEM.

These rules shall apply to the processing and encoding of all the files where XML is used as an encoding format in a GEM application.

Rules for encoding of the XML formatted files:

- The file shall be a well-formed XML document (but not necessarily valid against the DTD specified in this version of GEM). Here 'well-formed' and 'valid' are used as defined in the XML 1.0 (see XML 1.0 [60]) specification.

NOTE 1: The remark on validity is included, because it is possible to be valid only relative to one DTD. Valid documents relative to a DTD specified in a later version of GEM would not be valid relative to the DTD specified in the present document - however, the rules defined here intend to provide this future-proofness and allow terminal implementations compliant with the present document to be able to process files that may be encoded according to a later version of GEM.

- The XML files may contain the XMLDecl item ("<?xml ?>" tag) in the prologue in the beginning of the file.
- All the XML files shall be formatted using the UTF-8 character encoding which is the default used in XML.
- GEM terminal specifications may extend the allowed XML notation by permitting other character encodings, if this extension is explicitly stated in a GEM terminal specification. If no encoding is specified in an XML file, however, the default shall be UTF-8.
- The possible XMLDecl item in the beginning of the file shall not contain an 'encoding' attribute specifying another encoding than UTF-8.
- If the XMLDecl item is included in files conforming to the present document, it shall indicate XML version 1.0.
- The XML file shall contain a document type declaration ("<!DOCTYPE >" tag) where the Name is the same as the name of the root element.

- The document type declaration shall contain an `ExternalID` item with the "PUBLIC" identifier and both a `PublicLiteral` and a `SystemLiteral`. GEM specifies the `PublicLiteral` that shall be used to identify the document types defined in GEM. GEM specifies a `SystemLiteral` that can be used for identifying a location where the DTD can be retrieved. The `SystemLiteral` included in the document type declaration shall point to a location where the DTD can be obtained via the Internet using the HTTP protocol as specified in GEM.

- The `PublicLiteral` is used for identifying the type of the file. For document types specified in GEM, the `PublicLiteral` shall have the following syntax:

```
"-//DVB//DTD " <document type> " " <version_number> "//EN"
```

where:

```
<document type> has the following syntax:

    <document_type> = letter letters
    letters = "" | letter letters
    letter = uppercase_letter | lowercase_letter | space
    uppercase_letter =          "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H"
                                    | "I" | "J" | "K" | "L" | "M" | "N" | "O"
                                    | "P" | "Q" | "R" | "S" | "T" | "U" | "V"
                                    | "W" | "X" | "Y" | "Z"
    lowercase_letter =          "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h"
                                    | "i" | "j" | "k" | "l" | "m" | "n" | "o"
                                    | "p" | "q" | "r" | "s" | "t" | "u" | "v"
                                    | "w" | "x" | "y" | "z"
    space = " "

<version_number> has the following syntax:
    <version_number> = major_version "." minor_version
    major_version = digit digits
    minor_version = digit digits
    digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
    digits = "" | digit digits
```

The `<document_type>` part of the `PublicLiteral` is used as an identifier of the document type. When future versions of GEM specify newer, backwards compatible versions of the document type, the `<document_type>` part shall not be changed and the `<version_number>` part shall be changed to a new version number unused in a previous version of GEM for that document type.

- The XML file shall be valid relative to the DTD identified in the `ExternalId` document type declaration. Here 'valid' is used as defined in the XML 1.0 [60] specification.

- The document type declaration shall not contain a declaration part in square brackets ('[' and ']').

- Where the XML type PCDATA is used in XML elements and it is specified that this contains a string (for example a file name), these strings shall not contain '<', '&' or '>' characters that might be mistaken as XML tags or references of the markup.

NOTE 2: Strings where these characters are allowed should be specified to be encoded as CDATA, leading to a more complex notation but allowing those symbols to be used.

- The file shall include only tags and attributes that are defined in GEM or a later version of GEM.

- The file shall not include XML entity declarations ("<!ENTITY .>" tags).

- The file shall not include XML character or entity references (references starting with '&' character).

- The file shall not include XML processing instructions, except optionally the "<?xml ?>" XMLDecl item.

- The file may include XML comments ("<!-- -->" strings), but not within elements that are specified as PCDATA containing strings to be encoded as defined in GEM.

Rules for processing of the XML formatted files in the GEM terminal:

- The parser shall use the `PublicLiteral` in the document type declaration in the XML file ("<!DOCTYPE…>" tag) for identifying the type of the file.

- The `PublicLiteral` in the document type declaration identifies the version of the DTD that is used for this file. There is no requirement for the parser to try to fetch that DTD file using the URL defined by the `SystemLiteral`. It is an implementation option for the parser to retrieve the DTD from that URL. If the DTD is unavailable from that URL, then the behaviour shall be platform dependent.

- The parser shall accept files that have a different version number in the `PublicLiteral` than the one specified in GEM for the given file type. These are probably files encoded according to a different version of GEM. From those files, the parser shall parse, recognize and handle all those elements and attributes that are part of the DTD included in the present document.

- The parser shall ignore all XML elements (start tag, end tag, and possible string between them) that are not specified in the DTD included in the present document.

NOTE 3: This allows extending the DTDs in the future in a future proof manner where existing terminals ignore all the elements introduced in later versions of GEM

- The parser shall ignore such attributes of XML tags that are not specified in the DTD included in the present document.

- The parser shall ignore XML comments encoded as defined in the XML 1.0 specification.

- The parser must accept empty XML elements specified in GEM both in their start-tag and end-tag form as well as in the empty element tag form (e.g. '<tuning value="true"></tuning>' may be used as well as '<tuning value="true"/>').

- Rules for evolving the specification must ensure that the encoding of the file will always be maintained backwards compatible when these rules are followed (i.e. later versions of GEM may add new XML tags and new attributes to existing XML tags, but may not change the semantics of the existing elements).

- If the encoding of the file violates the rules defined for the encoding above, the behaviour of the parser can be platform dependent, including the possibility that the parser may completely discard such files and the system may behave as if the file is not present at all.

## 14.4 Network signalling (error behaviour)

The behaviour of GEM terminals when receiving incorrectly formatted data, however transmitted or otherwise acquired, is implementation dependent except where a specific error behaviour is required by GEM or referenced specifications. GEM terminals may implement whatever strategy they like for this situation. It is an allowed implementation choice to pass values from the network straight through to applications without checking them for correctness. Hence API calls which are specified as returning a specific piece of information may not return a valid piece of information if the original information in the network is wrong.

GEM terminals should observe the behaviour defined in section 4.1 of TS 101 154 [2].

NOTE: It is highly recommended that the GEM terminal should be designed to allow for future compatible extensions to the DVB SI, DSMCC or other formatted data interpreted by the GEM terminal. All of the fields "reserved" (for ISO), "reserved_future_use" (for ETSI), and "user defined" in theEN 300 468 [14] should be ignored by GEM terminals not designed to make use of them. The "reserved" and "reserved_future_use" fields may be specified in the future by the respective bodies, whereas the "user defined" fields will not be standardized. Where a GEM API provides access to this data, the data should be returned to the GEM application without validation or correction.

## 14.5 Text encoding of application identifiers

Where an `organisation_id` or `application_id` is encoded in textual form it shall be encoded as follows:

- A hexadecimal representation of the value.

- Lower case letters.

- No extra leading zeros (as would be produced by Integer.toHexString).

Where both an `organisation_id` and `application_id` are combined into an application identifier, they will be represented as a single hexadecimal number using the previously described encoding with the `organisation_id` as the most significant bits and the `application_id` as the least significant bits.

# 14.6 Filename requirements

## 14.6.1 Persistent storage

Receivers shall support path and file names as specified by `persistentpath` and `persistentfilename` in the following BNF:

```
lowalpha =      "a" | "b" | "c" | "d" | "e" | "f" | "g" | "h" | "i" |
                "j" | "k" | "l" | "m" | "n" | "o" | "p" | "q" | "r" |
                "s" | "t" | "u" | "v" | "w" | "x" | "y" | "z"

upalpha =       "A" | "B" | "C" | "D" | "E" | "F" | "G" | "H" | "I" |
                "J" | "K" | "L" | "M" | "N" | "O" | "P" | "Q" | "R" |
                "S" | "T" | "U" | "V" | "W" | "X" | "Y" | "Z"

alpha   =       lowalpha | upalpha

digit   =       "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"

punct   =       "_"

persistentfilechar := alpha | digit | punct

persistentfilesubstring := persistentfilechar | persistentfilesubstring persistentfilechar

persistentfilename = persistentfilesubstring
                           | persistentfilesubstring "."
                           | persistentfilesubstring "." persistentfilesuffix
persistentpath = persistentfilesubstring | persistentpath "/" persistentfilesubstring

persistentfilesuffix = persistentfilechar | persistentfilesuffix persistentfilechar
```

Receivers are required to support:

- `persistentfilesubstrings` of length less than or equal to 8 characters.
- `persistentfilesuffixes` of length less than or equal to 3 characters.

Receivers are not required to reject filenames which exceed these requirements but applications using such filenames are not compliant. These restrictions do not apply to stored applications.

Receivers shall have filesystems for persistent storage which are either case sensitive or "case preserving". Applications shall be written to work on both of these. "case preserving" filesystems are case insensitive when opening an existing file but preserve the case which was used when the file was initially created.

## 14.6.2 DSMCC object carousel

Receivers shall support object carousels at least matching these requirements;

- File paths (i.e. the concatenation of directory names, directory separators and a filename) must be ≤ 1 024 bytes long.
- Within the file path requirement, file and directory names must be ≤ 255 bytes.
- The OC file system shall be case sensitive.

NOTE 1: There is no limit on the maximum number of directory nesting levels as long as the limit on file paths is not exceeded.

NOTE 2: Authors should avoid relying on OC being case sensitive since some authoring platforms will get it wrong.

NOTE 3: To ease development of applications on computer platforms using traditional file systems the author is recommended to restrict file names to the character codes 0x21 to 0x7E excluding 0x22, 0x27, 0x3A, 0x3B, 0x5C (double quote, single quote, colon, semicolon, backslash).

### 14.6.3 Stored applications

Receivers shall support file and path names in stored applications as defined for the object carousel in clause 14.6.2, "DSMCC object carousel" except as follows:

- File systems for stored applications shall be either case sensitive or "case preserving". Applications to be stored shall be written to work on both of these. "Case preserving" is defined in clause 14.6.1, "Persistent storage" above.
- The characters 0x22 (double quote), 0x27 (single quote), 0x3A (colon), 0x3B (semicolon), 0x5C (backslash), 0x2F (slash), 0x3F (question-mark), 0x3C (less-than), 0x3E (greater-than), 0x2A (star) or 0x7C (pipe) shall not be used.

## 14.7 Files and file names

The locator format shall be able to reference any valid file name; at least the minima specified in clause B.2.1 shall be supported.

The GEM specification defines how applications can use the names of files in order to access content held in files. It is intentionally silent about the file systems and file system namespaces of GEM terminals except as defined below.

- When a GEM application starts, the filesystem where that application is carried will be mounted into the file system namespace of the GEM terminal concerned. For a DVB-J application clause11.5.1, "Broadcast Transport Protocol Access API" defines that creating a new instance of `java.io.File(".")` will result in a reference to the base directory of the application. The present document does not define the use of "file:" URLs relative to this base directory, only absolute URLs are defined. This base directory may be a sub-directory within this filesystem.
- GEM applications which have requested the right to access persistent storage, and had this right granted, are allowed to access the persistent file namespace. For DVB-J applications, the top level directory of this namespace is obtainable from the system property, `"dvb.persistent.root"`.
- GEM applications may have the ability to mount additional filesystems into the file system namespace of the GEM terminal concerned. DVB-J applications are allowed to use the `attach()` method on the `org.dvb.dsmcc.ServiceDomain` class in order to attach an object carousel as an additional file system. In all other methods, using a locator that refers to a file or directory, as described in clause 14.8, "Locators and content referencing" shall not mount the specified object carousel file system.
- Conformant GEM applications shall not attempt to access files or file systems outside what is allowed by the present document. The consequences should they attempt to do this are undefined and implementation dependent. Platforms are allowed to choose to limit the access rights of DVB-J applications through use of platform security mechanisms, e.g. `java.io.FilePermission`.
- References to content carried in files shall either be done using names of files encoded in text or using "file:" URLs as defined in RFC 1738 [61]. File names encoded in "dvb:" locators shall be transformed to "file:" URLs before use. For DVB-J applications, file names shall be encoded in Java String objects and "file:" URLs shall be encoded in instances of `java.net.URL`. See `ServiceDomain.getURL(Locator)`. When applications construct a `java.net.URL` for the "file:" protocol, the host part shall be empty and the path part shall be as specified for the constructors of `java.io.File` with the platform file separator replaced by "/" and omitting the first separator character.
- Stand-alone interaction channel applications shall see a file system where the current directory is derived from the contents of the AIT File (see clause 10.4.5, "AIT File"). Unless the GEM terminal is certain of the complete contents of a directory in this file system, all attempts to list such contents shall fail as if an I/O error had occurred in the underlying system.

# 14.8 Locators and content referencing

Table 59 lists the types of entity that may be addressed by locators in GEM, and defines the entities for which a text representation is required. In the case of locators, where a text representation is required GEM does not specify what that representation is, however GEM terminal specifications shall supply an unambiguous, concrete syntax for each of these entities.

NOTE 1: Clause 11.7.6 "Content referencing" describes how this text representation can be used in DVB-J applications.

GEM does not require support for addressing any other type of entity in a GEM system by locator or URL.

**Table 59: Addressable entities, locators and their text representation**

| Entity | Text Representation |
|---|---|
| **Required for all targets:** | |
| Service | locator text representation shall be defined |
| Service domain | locator text representation shall be defined |
| | |
| File | "file:" URL as defined in RFC 1738 [61] (note 1)<br>DVB locator including "dvb_abs_path" element (note 2)<br>"http:" URL as defined in RFC 2616 [45].<br>"https:" URL as defined in RFC 2818 [73]<br>locator text representation shall be defined for files located within a Service domain |
| Directory | "file:" URL as defined in RFC 1738 [61] (note 1)<br>DVB locator including "dvb_abs_path" element (note 2)<br>"http:" URL as defined in RFC 2616 [45].<br>"https:" URL as defined in RFC 2818 [73].locator text representation shall be defined for directories located within a Service domain |
| **Required for broadcast targets only:** | |
| Transport stream | locator text representation shall be defined |
| Network | no standardized text representation required |
| Program Event | locator text representation shall be defined |
| Drip feed decoder | "dripfeed://" |
| **Required for targets except IPTV:** | |
| MPEG Elementary Stream | locator text representation shall be defined |

NOTE 2: Even if an optional carousel functional equivalent is not defined, there still is a locator for files carrying the application.

# 14.9 Content referencing for IPTV/ OTT

## 14.9.1 Introduction (informative)

Java TV defines the concepts of transport independent locators and transport dependent locators. Services are equal if their Locators are equal which requires their external forms are equal. GEM permits both of the following;

- transport independent references to a service where there is no requirement for GEM applications to specify whether the service is available via a classical DVB network interface or an IP channel;
- transport dependent references to a service where GEM applications specify the network interface or channel by which the service is to be delivered.

In consequence, implementations may need to support at least 3 different external forms for locators - transport independent, transport dependent via classical DVB network interface and transport dependent via IP channel.

The SD&S broadcast discovery information record syntax permits the transport independent locator to be the basic "dvb:" locator as defined in MHP [1], clause 14.1, "Namespace mapping (DVB Locator)" since it contains enough information to resolve this locator. Hence this permits GEM applications using those locators to be independent of the means of their distribution. Transport dependent locators for content delivered via a classical DVB network interface need to be those defined in earlier versions of GEM for compatibility reasons.

Transport dependent locators for content delivered via IP need to identify the source of the transport stream and the service within that transport stream (for the case of MPTS). To identify the service within the transport stream, the DVB-SI service Id is sufficient since this is the same as the program number in the corresponding program map table.

### 14.9.2 Details

Table 60 shows the different forms of locators which are applicable to GEM.

**Table 60: Locators and URLs**

| Type of reference | Class | External Form |
|---|---|---|
| Transport independent | org.davic.net.dvb.DvbLocator | "dvb:" as in MHP [1], clause 14.1, "Namespace mapping (DVB Locator)" |
| Transport dependent | | |
| Service accessed via classic DVB network interface (see note 1) | org.davic.net.dvb.DvbNetworkBoundLocator | No standardized text representation |
| Service accessed via RTSP implemented in the client | org.dvb.locator.ip.RTSPLocator | "rtsp:" URL defined in clause 3.2 of RFC 2326 [83] |
| Service accessed via unicast IP (including RTSP implemented in a GEM application) | org.dvb.locator.ip.UnicastLocator | No standardized representation |
| Service accessed via multicast IP | org.dvb.locator.ip.MulticastLocator | No standardized representation (see note 2) |
| Service accessed via HTTP | org.dvb.locator.ott.HTTPLocator | “http:” URL defined in clause 3.2.2 of RFC 2616 [45] |
| Service accessed via HTTPS | org.dvb.locator.ott.HTTPLocator | “https:” URL as defined in clause 2.4. of RFC 2818 [73]. |
| NOTE 1: Not applicable in GEM terminals without a classic DVB network interface.<br>NOTE 2: The "rtp:" URI defined in clause 8 of TS 102 539 [81] may be used to obtain a MulticastLocator however this does not imply or restrict the content being accessed to being encapsulated in rtp. There is no requirement for MulticastLocator.toExternalForm to return a "rtp" URI. | | |

Where references to IP content use the "dvb:" URL defined in MHP [1], clause 14.1, "Namespace mapping (DVB Locator)" of the present document, these shall be matched against the DVBTriplet or TextualIdentifier fields in the broadcast discovery records defined by clause 5.2.6.2 of TS 102 034 [80].

A DvbNetworkBoundLocator is the transport dependent locator for services delivered by a classical DVB network interface. Even in GEM terminals without a classical DVB network interface, these shall not be matched against services announced by TS 102 034 [80].

## 14.10 Service identification

Java TV can support two kinds of locators for identifying a service: transport independent locators and transport dependent locators. Both enable global, unique identification of a service.

A transport independent locator has additional properties:

- It can identify two (or more) service instances as being the same service even if they for technical reasons have different transport dependent locators.

  It is up to the service provider to decide whether different service instances are identified as being the same service.

- They can give alternate identifications for a single service.

Terminal specifications based on GEM shall define a textual representation for at least one of the transport dependent or transport independent locators. A standardised format definition of the textual representation is beyond the scope of the present document.

## 14.11 CA system

GEM places no requirements on a CA system, if one is present.

NOTE: GEM terminal specifications may include the MHP requirements in MHP [1], clause 14.11 as described in clause 15.6, "Functional equivalents".

## 14.12 Focus management

NOTE 1: The following is basically aligned with the focus handling rules defined in clause 25.2.2.1.1 of OCAP [5].

The GEM terminal shall maintain an ordered list of those HScenes which are able to receive focus at the current time. HScenes shall be included in this list if and only if they meet all of the following criteria;

- they are visible (as defined by HScene.isVisible);
- they are active (as defined in HScene.setActive);
- they have explicitly requested focus at least one time;
- the HScene's application is in the Active state.

When an HScene not in the list meets all the criteria to be included in the list, it shall automatically be added to the end of the list. When an HScene in the list stops meeting all of these criteria, it shall automatically be removed from the list. Calling dispose() on an HScene in the list shall result in it being removed from the list.

When an HScene gets focus, it shall move to the start of the list pushing other HScenes down the list. The HScene with focus shall remain at the start of the list while it has focus. If the HScene with focus is removed from the list then the focus shall be given to the next HScene in the list.

NOTE 2: The class description of org.havi.ui.HScene defines requirements to report gaining and losing focus by applications using WindowEvents of type WINDOW_ACTIVATED and WINDOW_DEACTIVATED.

# 15 Detailed platform profile definitions

This clause defines the capabilities of platforms as presented to applications. Products that claim to conform to a profile shall provide at least the minimum capabilities identified for the profile. In some cases this implies that specific hardware resources are present in the platform.

All GEM platforms must support the following underlying platform specifications:

- CDC 1.1 [3] or later: the Java ME Connected Device Configuration (CDC) specification for non-graphical Java APIs for small electronic devices.
- PBP 1.1 [4] or later: the Java ME Personal Basis Profile (PBP) specifies a standards-based graphical user interface framework that support resource-constrained devices.

Taken together, the CDC and PBP provide a complete J2ME application environment for consumer products and embedded devices.

The following table is organized by four classes of GEM terminal specification targets: OTT, IPTV, broadcast and packaged media. For all but the OTT and IPTV targets, there are two profiles, the enhanced profile (abbreviated "E.P." in the table) and the interactive profile (abbreviated "I.P." in the table).

NOTE: The packaged media profile is present in the following table, but GEM 1.1 and GEM 1.2 changes have not been analyzed in the context of the packaged media profile.

**Table 61: Platform profile definitions**

| Area | Specification | IPTV Target | OTT Target | Broadcast Targets | | Packaged Media Targets | |
|---|---|---|---|---|---|---|---|
| | | | | E.P. | I.P. | E.P. | I.P. |
| **Static Formats** | | | | | | | |
| Bitmap pictures | 7.1.1.3, "PNG" + 15.1, "PNG - restrictions" | M | M | M | M | M | M |
| | 7.1.1.3, "PNG" without restrictions | - | - | - | - | - | - |
| | 7.1.1.4, "GIF" | - | - | - | - | - | - |
| | 7.1.2, "MPEG-2 I-Frames" | O | - | M | M | M | M |
| | 7.1.1.2, "JPEG" + 15.3, "JPEG - restrictions" | - | - | M | - | M | - |
| | 7.1.1.2, "JPEG" without restrictions | M | M | - | M | - | M |
| Audio clips | 7.1.4, "Monomedia format for audio clips" | M | M | M | M | M | M |
| Video drips | 7.1.3, "MPEG-2 Video "drips"" | O | - | M | M | M | M |
| Text encoding | 7.1.5, "Monomedia format for text" | M | M | M | M | M | M |
| **Media Streaming formats** | | | | | | | |
| Video | 7.1.5, "Monomedia format for text" | M | M | M | M | M | M |
| Audio | 7.2.1, "Audio" | M | M | M | M | M | M |
| Subtitles | 7.2.3, "Subtitles" | - | - | - | - | - | - |
| Container | 7.2.4. “Containers” | O | M | - | - | - | - |
| Manifest | 7.2.5 “Streaming Manifest” | O | O | - | - | - | - |
| **Fonts** | | | | | | | |
| Built in | Character set see annex E, "Character set"<br>Metrics see annex D, "Text presentation"<br>Face: UK RNIB "Tiresias" | O | O | M | M | O | O |
| Downloadable | 7.4.1, "PFR" | O | O | M | M | O | O |
| | 7.4.2, "OpenType" | O | O | O | O | O | O |
| **Broadcast channel protocols** | | | | | | | |
| MPEG-2 | 6.2.2, "MPEG-2 sections" | O | O | M | M | M | M |
| Object Carousel | 6.2.5, "Object carousel" | O | O | M | M | - | - |
| IP Multicast | IP Multicast stack based on:<br>6.2.6, "Protocol for delivery of IP multicast over the broadcast channel"<br>6.2.7, "Internet Protocol (IP)"<br>6.2.8, "User Datagram Protocol (UDP)"<br>6.2.10, "IP signalling" | - | - | O | Ro | - | - |
| **Interaction channel protocols** | | | | | | | |
| TCP/IP | 6.3.3, "Transmission Control Protocol"<br>6.3.2, "Internet Protocol" | M | M | - | M | - | M |
| UDP/IP | 6.3.2, "Internet Protocol"<br>6.3.8, "User Datagram Protocol" | M | M | - | M | - | M |
| IGMP | 6.5.1.5, “Internet Group Management Protocol (IGMP)” | O | O | - | - | - | - |
| RTP/UDP/IP | 6.3.2, "Internet Protocol"<br>6.3.8, "User Datagram Protocol"<br>6.5.1.3, “Real Time Protocol (RTP)” | O | O | - | - | - | - |
| RTSP/UDP/IP | 6.3.2, "Internet Protocol"<br>6.3.8, "User Datagram Protocol"<br>6.5.1.4, “Real Time Streaming Protocol (RTSP)” | O | O | - | - | - | - |
| HTTP | 6.3.7.1, "HTTP 1.1" | O | M | - | O | - | O |
| | 6.3.7.2, "GEM profile of HTTP 1.0" | M | - | - | M | - | M |
| DSMCC-UU RPC | 6.3.4, "UNO-RPC"<br>6.3.5, "UNO-CDR"<br>6.3.6, "DSM-CC User to User" | - | - | - | O | - | - |
| DNS | 6.3.9, "DNS" | M | M | - | M | - | M |
| HTTPS | 6.3.7.3, "HTTPS" | M | M | - | M | - | M |
| Interaction Channel File System | 6.4.1, "File system implemented only by the interaction channel" | O | O | - | M | - | M |
| DSMCC / HTTP hybrid | 6.4.2, "Hybrid between broadcast stream and interaction channel" | O | O | - | M | - | M |
| IPTV | 5, "Basic architecture" | O | O | O | O | O | O |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Application Model** | | | | | | | |
| Application Model | All parts of clause 9, "Application model" except those clauses (and their subclauses) identified below | M | M | M | M | M | M |
| | 9.6.1, "Applications loaded from the interaction channel" | O | O | O | O | O | O |
| | 9.6.2, "Stored services" | O | O | O | O | - | - |
| | 9.7, "Lifecycle of internet access applications" | O | O | - | M | - | O |
| | 9.9, "Stored and cached applications" | O | O | O | O | - | - |
| | 9.13, "Unbound Applications" | O | O | O | O | O | O |
| **Application Signalling** | | | | | | | |
| Application Signalling | 10, "Application signalling" | M | M | M | M | M | M |
| **DVB-J** | | | | | | | |
| DVB-J Platform | All parts of clause 11, "DVB-J platform" except those clauses (and their subclauses) identified below. | M | M | M | M | M | M |
| Presentation APIs | 11.4.1, "Graphical User Interface API" | O | O | O | O | O | O |
| Data Access APIs | 11.5.2, "Support for Multicast IP over the Broadcast Channel" | - | - | O | Ro | - | - |
| | 11.5.3, "Support for IP over the Return Channel" | M | M | - | M | - | M |
| | 11.5.4, "MPEG-2 Section Filter API" | O | O | M | M | O | O |
| | 11.5.5, "Mid-Level Communications API" as modified by 11.5, "Data access APIs" | M | M | O | M | O | M |
| | 11.5.7, "File Storage Device Access" | O | O | O | O | O | O |
| Service Information and Selection APIs | 11.6.3, "Tuning API" | O | O | M | M | - | - |
| | 11.6.4, "Conditional access API" | - | - | - | - | - | - |
| | 11.6.6, "Service discovery and selection for IPTV" | O | O | - | - | - | - |
| | 11.6.7, "Integration between protocol independent SI API and TV-Anytime" | O | O | - | - | - | - |
| Common Infrastructure APIs | 11.7.4, "Basic MPEG concepts" | O | O | M | M | - | - |
| | 11.7.6 "Content referencing" | O | O | O | O | O | O |
| | 11.7.7, "Common error reporting" | O | O | M | M | M | M |
| | 11.7.8, "Plug-in APIs" | O | O | M | M | O | O |
| | 11.7.9, "Provider API" | O | O | - | - | - | - |
| | 11.7.10, "Content referencing for IPTV" | O | O | - | - | - | - |
| | 11.7.11, "TV-Anytime content referencing and metadata" | O | O | - | - | - | - |
| | 11.7.12, “Content referencing for OTT” | O | M | - | - | - | - |
| Security | 11.8.2, "APIs for return channel security" | M | M | - | M | - | M |
| | 11.8.3, "Additional permissions classes" | O | O | M | M | - | - |
| | 11.8.6, "DVB Extensions for Cryptography" | O | O | O | O | O | O |
| Other APIs | 11.9.4, "Non-CA smart card API" | O | O | O | O | O | O |
| | 11.9.5.2, "JDOM" | O | O | - | - | - | - |
| | 11.9.6, "GEM terminal hardware API" | O | O | O | O | O | O |
| | 11.9.7, “Content Download API” | O | O | - | - | - | - |
| | 11.4.2.5.7, “Streaming Monitoring API” | O | M | - | - | - | - |
| | 11.4.2.5.8, “Media Stream Synchronization API” | O | O | - | - | - | - |
| Content Referencing | 11.11.1, "Transport stream" | O | O | M | M | O | O |
| | 11.11.2, "Network" | O | O | M | M | O | O |
| | 11.11.4.4, “Content referencing for IPTV” | M | O | - | - | - | - |
| | 11.11.4.5, “Content referencing for OTT services” | O | M | - | - | - | - |
| | 11.11.5, "Program event" | O | O | M | M | O | O |
| | 11.11.9, "Drip feed decoder" | O | O | M | M | O | O |
| Stand-alone Apps | 11.12.2, "Stored services" | O | O | O | O | O | O |
| OCAP APIs | 11.15, "APIs defined in OCAP" | O | O | O | O | O | O |
| Internet Access | 11.14, "Internet Access" | O | O | - | O | - | O |
| | 17, "Internet access clients" | O | O | - | O | - | O |

Key:

| | |
|---|---|
| - | Not applicable/Not required. |
| O | Optional feature in the GEM terminal. |
| Ro | Recommended optional feature in the GEM terminal. |
| M | Mandatory feature in the GEM terminal |

## Optional APIs:

An optional API means optional functionality, i.e. a GEM terminal is not required to include it.

Terminal implementations, which use eager linking VMs (see section 11.2. of the Java Language Specification [11]) shall provide API signatures with empty implementation (Compilation stubs) for all non-implemented optional APIs. This is required to prevent VM startup failures on these implementations.

For each package of an optional API, where only an empty implementation (stub) exists, there MUST be a system property consisting of the package name with a property value of "STUBBED". Additionally these classes shall contain a static initialization block which throws a NoClassDefFoundError to indicate an attempt to illegally use it.

NOTE: In this version of the specification, for IPTV the functional equivalent for clause 6.2.5, "Object carousel" is optional. This will be revisited in a future version of the present document. A mechanism for mounting a filesystem using `ServiceDomain.attach(Locator)` and a way of sending triggers that can be received using the DSMCCStreamEvent API are important for interoperability, e.g. with packaged media applications. A future version of the present document may make this level of functional equivalent mandatory, and IPTV implementations that wish to support interoperability with these commonly used capabilities have to support these functionalities.

If a GEM terminal specification wishes to include APIs, signalling or behaviours defined in MHP [1] as functional equivalents, it shall do so by referencing the corresponding entry in clause 15.6, "Functional equivalents" and not by directly referencing MHP.

# 15.1 PNG - restrictions

PNG is defined as in PNG Version 1.0 [43]; later versions of PNG introduce additional chunk types, which are not required to be supported. Receivers should ignore gAMA (gamma) and cHRM (chromaticity) chunks in PNG files.

GEM terminals are required to support all of the PNG colour types defined in PNG Specification Version 1.0 [43] (see table 62). GEM terminals are responsible for mapping these colours to those used by the terminal's OSD.

Any combination of PNGs with different colour types may be active at any one time. Similarly, terminals are responsible for mapping RGB16 direct colour specifications to colours that the OSD can support.

**Table 62: PNG formats**

| Colour Type | Allowed Bit Depths | Interpretation |
|---|---|---|
| 0 | 1, 2, 4, 8, 16 | Each pixel is a grayscale sample. |
| 2 | 8, 16 per component | Each pixel is an R,G,B triple. |
| 3 | 1, 2, 4, 8 | Each pixel is a palette index; PLTE chunk must appear. |
| 4 | 8, 16 | Each pixel is a grayscale sample, followed by an alpha sample. |
| 6 | 8, 16 | Each pixel is an R,G,B triple, followed by an alpha sample. |

# 15.2 Minimum media formats supported by DVB-J APIs

Table 63 specifies the minimum set of media types that implementations of the "Enhanced Broadcast Profile" and "Interactive Broadcast Profile" shall support (see clause 4.1.6, "Profiles"). It also identifies the APIs that shall provide this support:

**Table 63: Media type support required in Enhanced and Interactive Broadcast profiles**

| Media type | Reference | API(s) |
|---|---|---|
| Downloadable fonts | 7.4 | `org.dvb.ui.FontFactory` |
| Audio from file | 7.1.4 | `org.havi.ui.HSound`<br>JMF only with references to files |
| MPEG I frame images | 7.1.2 | `org.havi.ui.HBackgroundImage` |
| PNG images | 7.1.1.3 | `java.awt.Image` |
| JPEG images | 7.1.1.2 | `java.awt.Image` |
| DVB service | EN 300 468 [14] | JMF only with references to DVB services for playback direct from the network |
| Video "drips" | 7.1.3 | `org.dvb.media.DripFeedDataSource` |
| Text files | 7.1.5 | All Java APIs supporting reading or writing text files. |
| Subtitles | 7.2.3 | JMF only as part of a DVB service |

Support for subtitles is optional.

Support of MPEG-2 Video "drips" is optional for the IPTV target.

Media formats in table 63, "Media type support required in Enhanced and Interactive Broadcast profiles" may be replaced with functional equivalents as permitted by clause 15.6, "Functional equivalents".

## 15.3 JPEG - restrictions

The restricted JPEG specification is as specified in clause 7.1.1.2, "JPEG" except that the "progressive DCT-based" mode is excluded.

## 15.4 Locale support

Support of resources for the following locales is required:

- One guaranteed Locale java.util.Locale.UK. (“en”. ”GB”).
- Zero (or more) implementation dependent ones.

Further it is guaranteed that the default Locale shall have resources. The default Locale is implementation dependent.

NOTE: Terminal specifications may, of course, guarantee support for locales in addition to the UK locale, however, support for the UK locale is required by the present document.

## 15.5 Video raster format dependencies

This clause addresses the aspects of the present document that vary as a consequence of the video raster format.

### 15.5.1 Standard Definition (PAL/SECAM or NTSC resolution)

#### 15.5.1.1 Logical pixel resolution

The logical pixel resolution shall be 72 dots per inch.

NOTE: This is the a convention adopted by the Java2D API. See the specification of `java.awt.Font.getSize()`. As a result of this convention, the return value of `getSize()` gives the size in pixels for a screen device.

## 15.6 Functional equivalents

Table 64 lists the set of mandatory and optional functional equivalents. Those elements marked mandatory shall be defined in a GEM terminal specification. Those marked optional may be defined in a GEM terminal specification.

Each functional equivalent is identified by a name in the leftmost column of table 64.

If a GEM terminal specification wishes to include technology that is not required by GEM, it may do so only for one or more named functional equivalents listed in column 1 in table 64 and shall be conformant with the clauses for that element referenced by table 64.

For each mandatory named functional equivalent, the GEM terminal specification shall explicitly indicate that it adopts the entire definition as specified in the current document or shall specify a replacement as required by the corresponding GEM clause(s) in table 64.

For each optional named functional equivalent, the GEM terminal specification shall explicitly indicate whether it is supported. If supported, the GEM terminal specification shall indicate that it adopts the entire definition as specified in the current document or shall specify a replacement as required by the corresponding GEM clause(s) in table 64.

Table 64 is organized by the three classes of GEM terminal specification targets, IPTV, broadcast and packaged media. For all but the IPTV target, there are two profiles, the enhanced profile (abbreviated "E.P." in the table) and the interactive profile (abbreviated "I.P." in the table).

**Table 64: Functional equivalents**

| Name | GEM Clause(s) | IPTV Target | OTT Target | Broadcast Targets | | Packaged Media Targets | |
|---|---|---|---|---|---|---|---|
| | | | | E.P. | I.P. | E.P. | I.P. |
| Arch | 5, "Basic architecture" | O | O | O | O | O | O |
| Carousel | 6.2.5, "Object carousel"<br>See also 11.7.2, "Application discovery and launching APIs" | O | O | M | M | M | M |
| IP MPE | 6.2.6, "Protocol for delivery of IP multicast over the broadcast channel" | O | O | O | O | O | O |
| SI | 6.2.9, "Service information"<br>11.6.1, "Signalling-specific service information API"<br>Annex O, "Integration of the JavaTV SI API" | M | M | M | M | M | M |
| Broadcast IP signalling | 6.2.10, "IP signalling" | O | O | O | O | O | O |
| IPTV Protocols | 6.5, "IPTV protocols" | M | - | - | - | - | - |
| OTT Protocols | 6.6, ”OTT Protocols” | - | M | - | - | - | - |
| Audio | 7.2.1, "Audio" | M | M | M | M | M | M |
| Video | 7.2.2, "Video" | M | M | M | M | M | M |
| Subtitles | 7.2.3, "Subtitles"<br>11.4, "Presentation APIs" - classes related to subtitles | O | O | O | O | O | O |
| Audio Clips | 7.1.4, "Monomedia format for audio clips" | M | M | M* | M* | M | M |
| Resident Fonts | 7.3, "Resident fonts" | M | M | M* | M* | O | O |
| Downloadable Fonts | 7.4, "Downloadable fonts" | M* | M* | M* | M* | M | M |
| Application Signalling | 10.2, "Program specific information"<br>10.4, "Application Description"<br>10.5, "DVB-J specific Application Description"<br>11.7.2, "Application discovery and launching APIs" | M | M | M | M | M | M |
| Application Authentication | 12.2, "Authentication of applications" (see notes 1 and 2)<br>12.9.1, "Certificate Revocation Lists" | M | M | M | M | M | M |
| Conditional Access | 11.4, "Presentation APIs," classes related to conditional access<br>11.6.4, "Conditional access API" | O | O | O | O | O | O |

| Name | GEM Clause(s) | IPTV Target | OTT Target | Broadcast Targets | | Packaged Media Targets | |
|---|---|---|---|---|---|---|---|
| | | | | E.P. | I.P. | E.P. | I.P. |
| Content Referencing | 11.7.6, "Content referencing"<br>11.11.11, "Methods working on many locator types"<br>14.1, "Namespace mapping"<br>14.9, ″Content referencing for IPTV/ OTT″ | M | M | M | M | M | M |
| Graphics Resolution | D.3.4.2, "Horizontal resolution"<br>G.1.1, "Device capabilities"<br>G.4, "Resident fonts and text rendering" | M | M | M | M | M | M |
| Text Wrapping | D.3.7.2, "Text wrapping setting is true" | M | M | M | M | M | M |
| RCMM | 12.9.2, "Root certificate management" | M | M | M | M | M | M |
| Active Format Descriptor | N.1, "Active Format Definition" | O | O | O | O | O | O |
| NOTE 1: See also text in clause 12.6.2.6, "Credentials" and text in clause 12.9, "Certificate management".<br>NOTE 2: See also text in clause 12.9.2, "Root certificate management". | | | | | | | |
| Key:<br>M Functional equivalent required<br>M* MHP definition of the functional equivalent required.<br>O Optional feature, no functional equivalent required.<br>- Not required/Not applicable. | | | | | | | |

NOTE: Features are labelled "O" if they could reasonably be applied to the given target and/or profile. This implies that if this feature is implemented, it is done according to the specification provided by GEM.

## 15.6.1 Modifications to MHP Definitions of Functional equivalents

In certain cases, it may be necessary to slightly modify a functional equivalent. A GEM terminal specification that adopts a functional equivalent may modify it only as described in this clause.

### 15.6.1.1 Carousel

#### 15.6.1.1.1 NSAP Address

The DVB Carousel NSAP Address structure may be modified by replacing the definition of `specifierType`, `specifierData`, and `dvb_service_location` structures with structures specified in the GEM terminal specification.

NOTE: This may be necessary because the `dvb_service_location` is based on `transport_stream_id`, `original_network_id` and `service_id` values that come from DVB SI.

#### 15.6.1.1.2 Content type descriptor

If a content type descriptor is not present in a file object, GEM terminal specifications may define another mechanism to determine the MIME type of a file that takes precedence over the mapping based on the extension portion of the file name, provided that a correctly built MHP object carousel will be processed in a way compatible with MHP.

#### 15.6.1.1.3 Application Icons Descriptor

GEM terminal specifications using the functional equivalent of application signalling may define new application types. If they do, they may define icon locator semantics for the application types they introduce in the application icons descriptor.

### 15.6.1.2 Application Signalling

#### 15.6.1.2.1 Transport protocol descriptor

GEM terminal specifications using the functional equivalent named "application signalling" may define new `protocol_id`s and corresponding selector bytes provided that new `protocol_id` values shall be registered with the DVB.

#### 15.6.1.2.2 AIT descriptor tag values

GEM terminal specifications that include the functional equivalent named "Application Signalling" and that define additional descriptors in the AIT shall register the descriptor tag values with the DVB.

NOTE: For information purposes current registrations are shown in clause 10.6, "Constant Values".

### 15.6.1.3 Application Name Descriptor

GEM terminal specifications using the functional equivalent named "application signalling" may modify the encoding of the `application_name_char` field of the application name descriptor provided that all strings that start with the value 0x20 are interpreted as ASCII strings, in accordance with annex A of EN 300 468 [14].

# 16 Registry of constants

## 16.1 System constants

**Table 65: Registry of constants**

| Entity | Value | Description |
|---|---|---|
| PTimerMinRepeatInterval | 40 ms | This (or optionally a smaller) value shall be returned by javax.tv.util.TVTimer.getMinRepeatInterval(). See clause 11.9.1, "Timer support". |
| PTimerGranularity | 10ms | This (or optionally a smaller) value shall be returned by javax.tv.util.TVTimer.getGranularity(). See clause 11.9.1, "Timer support". |

## 16.2 DVB-J constants

Where this clause lists a constant for a class that is not required by the present document (e.g. a class in the package `org.dvb.si`), that constant is not required.

This clause to be populated with the values of public final static symbols from the various Java APIs.

### 16.2.1 Public and Protected final static primitive fields from DVB packages

The following is a list of the values assigned for public and protected final static primitive fields defined in the DVB defined DVB-J packages:

```
public final static int org.dvb.application.AppAttributes.DVB_J_application = 1;
public final static int org.dvb.application.AppAttributes.DVB_HTML_application = 2;
public final static int org.dvb.application.AppProxy.STARTED = 0;
public final static int org.dvb.application.AppProxy.DESTROYED = 1;
public final static int org.dvb.application.AppProxy.NOT_LOADED = 2;
public final static int org.dvb.application.AppProxy.PAUSED = 3;
public final static int org.dvb.application.AppsDatabaseEvent.NEW_DATABASE = 0;
public final static int org.dvb.application.AppsDatabaseEvent.APP_CHANGED = 1;
public final static int org.dvb.application.AppsDatabaseEvent.APP_ADDED = 2;
public final static int org.dvb.application.AppsDatabaseEvent.APP_DELETED = 3;
public final static int org.dvb.application.DVBJProxy.LOADED = 5;

public static final int org.dvb.application.DVBHTMLProxy.LOADING=6;
public static final int org.dvb.application.DVBHTMLProxy.KILLED=7;

public final static int org.dvb.dsmcc.DSMCCObject.FROM_CACHE = 1;
public final static int org.dvb.dsmcc.DSMCCObject.FROM_CACHE_OR_STREAM = 2;
public final static int org.dvb.dsmcc.DSMCCObject.FROM_STREAM_ONLY = 3;
public final static int org.dvb.dsmcc.DSMCCObject.FORCED_STATIC_CACHING=4;

public final static int org.dvb.event.UserEvent.UEF_KEY_EVENT = 1;
public final static int org.dvb.io.ixcIxcRegistry.SERVICE=1;
public final static int org.dvb.io.ixcIxcRegistry.PAGE=2;
public final static int org.dvb.io.ixcIxcRegistry.GLOBAL=3;

public final static int org.dvb.io.persistent.FileAttributes.PRIORITY_LOW = 1;
public final static int org.dvb.io.persistent.FileAttributes.PRIORITY_MEDIUM = 2;
public final static int org.dvb.io.persistent.FileAttributes.PRIORITY_HIGH = 3;

public final static int org.dvb.media.PresentationChangedEvent.STREAM_UNAVAILABLE = 0;
public final static int org.dvb.media.PresentationChangedEvent.CA_FAILURE = 1;
public final static int org.dvb.media.PresentationChangedEvent.CA_RETURNED = 2;
public final static int org.dvb.media.VideoFormatControl.ASPECT_RATIO_UNKNOWN = -1;
public final static int org.dvb.media.VideoFormatControl.AFD_NOT_PRESENT = -1;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_UNKNOWN = -1;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_NONE = 0;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_FULL = 1;
public final static int org.dvb.media.VideoFormatControl.DAR_4_3 = 1;
public final static int org.dvb.media.VideoFormatControl.ASPECT_RATIO_4_3 = 2;
```

```
public final static int org.dvb.media.VideoFormatControl.AFD_16_9_TOP = 2;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_LB_16_9 = 2;
public final static int org.dvb.media.VideoFormatControl.DAR_16_9 = 2;
public final static int org.dvb.media.VideoFormatControl.ASPECT_RATIO_16_9 = 3;
public final static int org.dvb.media.VideoFormatControl.AFD_14_9_TOP = 3;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_LB_14_9 = 3;
public final static int org.dvb.media.VideoFormatControl.ASPECT_RATIO_2_21_1 = 4;
public final static int org.dvb.media.VideoFormatControl.AFD_GT_16_9 = 4;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_CCO = 4;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_PAN_SCAN = 5;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_LB_2_21_1_ON_4_3 = 6;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_LB_2_21_1_ON_16_9 = 7;
public final static int org.dvb.media.VideoFormatControl.AFD_SAME = 8;
public final static int org.dvb.media.VideoFormatControl.DFC_PLATFORM = 8;
public final static int org.dvb.media.VideoFormatControl.AFD_4_3 = 9;
public final static int org.dvb.media.VideoFormatControl.AFD_16_9 = 10;
public final static int org.dvb.media.VideoFormatControl.AFD_14_9 = 11;
public final static int org.dvb.media.VideoFormatControl.AFD_4_3_SP_14_9 = 13;
public final static int org.dvb.media.VideoFormatControl.AFD_16_9_SP_14_9 = 14;
public final static int org.dvb.media.VideoFormatControl.AFD_16_9_SP_4_3 = 15;
public final static int org.dvb.media.VideoFormatControl.DFC_PROCESSING_16_9_ZOOM = 9;
public final static byte org.dvb.media.VideoPresentationControl.POS_CAP_OTHER = -1;
public final static byte org.dvb.media.VideoPresentationControl.POS_CAP_FULL = 0;
public final static byte
org.dvb.media.VideoPresentationControl.POS_CAP_FULL_IF_ENTIRE_VIDEO_ON_SCREEN = 1;
public final static byte org.dvb.media.VideoPresentationControl.POS_CAP_FULL_EVEN_LINES = 3;
public final static byte
org.dvb.media.VideoPresentationControl.POS_CAP_FULL_EVEN_LINES_IF_ENTIRE_VIDEO_ON_SCREEN = 4;

public final static int org.dvb.net.rc.RCInterface.TYPE_PSTN = 1;
public final static int org.dvb.net.rc.RCInterface.TYPE_ISDN = 2;
public final static int org.dvb.net.rc.RCInterface.TYPE_DECT = 3;
public final static int org.dvb.net.rc.RCInterface.TYPE_CATV = 4;
public final static int org.dvb.net.rc.RCInterface.TYPE_LMDS = 5;
public final static int org.dvb.net.rc.RCInterface.TYPE_MATV = 6;
public final static int org.dvb.net.rc.RCInterface.TYPE_RCS = 7;
public final static int org.dvb.net.rc.RCInterface.TYPE_UNKNOWN = 8;
public final static int org.dvb.net.rc.RCInterface.TYPE_OTHER = 9;

public final static short org.dvb.si.DescriptorTag.NETWORK_NAME = 64;
public final static short org.dvb.si.DescriptorTag.SERVICE_LIST = 65;
public final static short org.dvb.si.DescriptorTag.STUFFING = 66;
public final static short org.dvb.si.DescriptorTag.SATELLITE_DELIVERY_SYSTEM = 67;
public final static short org.dvb.si.DescriptorTag.CABLE_DELIVERY_SYSTEM = 68;
public final static short org.dvb.si.DescriptorTag.BOUQUET_NAME = 71;
public final static short org.dvb.si.DescriptorTag.SERVICE = 72;
public final static short org.dvb.si.DescriptorTag.COUNTRY_AVAILABILITY = 73;
public final static short org.dvb.si.DescriptorTag.LINKAGE = 74;
public final static short org.dvb.si.DescriptorTag.NVOD_REFERENCE = 75;
public final static short org.dvb.si.DescriptorTag.TIME_SHIFTED_SERVICE = 76;
public final static short org.dvb.si.DescriptorTag.SHORT_EVENT = 77;
public final static short org.dvb.si.DescriptorTag.EXTENDED_EVENT = 78;
public final static short org.dvb.si.DescriptorTag.TIME_SHIFTED_EVENT = 79;
public final static short org.dvb.si.DescriptorTag.COMPONENT = 80;
public final static short org.dvb.si.DescriptorTag.MOSAIC = 81;
public final static short org.dvb.si.DescriptorTag.STREAM_IDENTIFIER = 82;
public final static short org.dvb.si.DescriptorTag.CA_IDENTIFIER = 83;
public final static short org.dvb.si.DescriptorTag.CONTENT = 84;
public final static short org.dvb.si.DescriptorTag.PARENTAL_RATING = 85;
public final static short org.dvb.si.DescriptorTag.TELETEXT = 86;
public final static short org.dvb.si.DescriptorTag.TELEPHONE = 87;
public final static short org.dvb.si.DescriptorTag.LOCAL_TIME_OFFSET = 88;
public final static short org.dvb.si.DescriptorTag.SUBTITLING = 89;
public final static short org.dvb.si.DescriptorTag.TERRESTRIAL_DELIVERY_SYSTEM = 90;
public final static short org.dvb.si.DescriptorTag.MULTILINGUAL_NETWORK_NAME = 91;
public final static short org.dvb.si.DescriptorTag.MULTILINGUAL_BOUQUET_NAME = 92;
public final static short org.dvb.si.DescriptorTag.MULTILINGUAL_SERVICE_NAME = 93;
public final static short org.dvb.si.DescriptorTag.MULTILINGUAL_COMPONENT = 94;
public final static short org.dvb.si.DescriptorTag.PRIVATE_DATA_SPECIFIER = 95;
public final static short org.dvb.si.DescriptorTag.SERVICE_MOVE = 96;
public final static short org.dvb.si.DescriptorTag.SHORT_SMOOTHING_BUFFER = 97;
public final static short org.dvb.si.DescriptorTag.FREQUENCY_LIST = 98;
public final static short org.dvb.si.DescriptorTag.PARTIAL_TRANSPORT_STREAM = 99;
public final static short org.dvb.si.DescriptorTag.DATA_BROADCAST = 100;
public final static byte org.dvb.si.PMTStreamType.MPEG1_VIDEO = 1;
public final static byte org.dvb.si.PMTStreamType.MPEG2_VIDEO = 2;
public final static byte org.dvb.si.PMTStreamType.MPEG1_AUDIO = 3;
public final static byte org.dvb.si.PMTStreamType.MPEG2_AUDIO = 4;
```

```
public final static short org.dvb.si.SIInformation.FROM_CACHE_ONLY = 0;
public final static short org.dvb.si.SIInformation.FROM_CACHE_OR_STREAM = 1;
public final static short org.dvb.si.SIInformation.FROM_STREAM_ONLY = 2;
public final static byte org.dvb.si.SIMonitoringType.NETWORK = 1;
public final static byte org.dvb.si.SIMonitoringType.BOUQUET = 2;
public final static byte org.dvb.si.SIMonitoringType.SERVICE = 3;
public final static byte org.dvb.si.SIMonitoringType.PMT_SERVICE = 4;
public final static byte org.dvb.si.SIMonitoringType.PRESENT_FOLLOWING_EVENT = 5;
public final static byte org.dvb.si.SIMonitoringType.SCHEDULED_EVENT = 6;
public final static byte org.dvb.si.SIRunningStatus.UNDEFINED = 0;
public final static byte org.dvb.si.SIRunningStatus.NOT_RUNNING = 1;
public final static byte org.dvb.si.SIRunningStatus.STARTS_IN_A_FEW_SECONDS = 2;
public final static byte org.dvb.si.SIRunningStatus.PAUSING = 3;
public final static byte org.dvb.si.SIRunningStatus.RUNNING = 4;
public final static short org.dvb.si.SIServiceType.UNKNOWN = -1;
public final static short org.dvb.si.SIServiceType.DIGITAL_TELEVISION = 1;
public final static short org.dvb.si.SIServiceType.DIGITAL_RADIO_SOUND = 2;
public final static short org.dvb.si.SIServiceType.TELETEXT = 3;
public final static short org.dvb.si.SIServiceType.NVOD_REFERENCE = 4;
public final static short org.dvb.si.SIServiceType.NVOD_TIME_SHIFTED = 5;
public final static short org.dvb.si.SIServiceType.MOSAIC = 6;
public final static short org.dvb.si.SIServiceType.PAL = 7;
public final static short org.dvb.si.SIServiceType.SECAM = 8;
public final static short org.dvb.si.SIServiceType.D_D2_MAC = 9;
public final static short org.dvb.si.SIServiceType.FM_RADIO = 10;
public final static short org.dvb.si.SIServiceType.NTSC = 11;
public final static short org.dvb.si.SIServiceType.DATA_BROADCAST = 12;
public final static short org.dvb.si.SIServiceType.MHP_APPLICATION = 16;

public final static int org.dvb.test.DVBTest.UNTESTED = -5;
public final static int org.dvb.test.DVBTest.UNRESOLVED = -4;
public final static int org.dvb.test.DVBTest.HUMAN_INTERVENTION = -3;
public final static int org.dvb.test.DVBTest.OPTION_UNSUPPORTED = -2;
public final static int org.dvb.test.DVBTest.FAIL = -1;
public final static int org.dvb.test.DVBTest.PASS = 0;

public final static int org.dvb.ui.DVBAlphaComposite.CLEAR = 1;
public final static int org.dvb.ui.DVBAlphaComposite.SRC = 2;
public final static int org.dvb.ui.DVBAlphaComposite.SRC_OVER = 3;
public final static int org.dvb.ui.DVBAlphaComposite.DST_OVER = 4;
public final static int org.dvb.ui.DVBAlphaComposite.SRC_IN = 5;
public final static int org.dvb.ui.DVBAlphaComposite.DST_IN = 6;
public final static int org.dvb.ui.DVBAlphaComposite.SRC_OUT = 7;
public final static int org.dvb.ui.DVBAlphaComposite.DST_OUT = 8;
public final static int org.dvb.ui.DVBBufferedImage.TYPE_ADVANCED = 20;
public final static int org.dvb.ui.DVBBufferedImage.TYPE_BASE = 21;
public final static int org.dvb.ui.DVBTextLayoutManager.HORIZONTAL_START_ALIGN = 1;
public final static int org.dvb.ui.DVBTextLayoutManager.HORIZONTAL_END_ALIGN = 2;
public final static int org.dvb.ui.DVBTextLayoutManager.HORIZONTAL_CENTER = 3;
public final static int org.dvb.ui.DVBTextLayoutManager.VERTICAL_START_ALIGN = 4;
public final static int org.dvb.ui.DVBTextLayoutManager.VERTICAL_END_ALIGN = 5;
public final static int org.dvb.ui.DVBTextLayoutManager.VERTICAL_CENTER = 6;
public final static int org.dvb.ui.DVBTextLayoutManager.LINE_ORIENTATION_HORIZONTAL = 10;
public final static int org.dvb.ui.DVBTextLayoutManager.LINE_ORIENTATION_VERTICAL = 11;
public final static int org.dvb.ui.DVBTextLayoutManager.START_CORNER_UPPER_LEFT = 20;
public final static int org.dvb.ui.DVBTextLayoutManager.START_CORNER_UPPER_RIGHT = 21;
public final static int org.dvb.ui.DVBTextLayoutManager.START_CORNER_LOWER_LEFT = 22;
public final static int org.dvb.ui.DVBTextLayoutManager.START_CORNER_LOWER_RIGHT = 23;

public static final int org.dvb.internet.InternetServiceFilter.EMAIL_CLIENT = 1;
public static final int org.dvb.internet.InternetServiceFilter.WWW_CLIENT = 2;
public static final int org.dvb.internet.InternetServiceFilter.NEWS_CLIENT = 3;
public static final int org.dvb.application.AppProxy.INVALID = 8;

public static final int org.dvb.spi.ict.ResourceTransportObject.LOAD_TYPE_FILE = 1;
public static final int org.dvb.spi.ict.ResourceTransportObject.LOAD_TYPE_MEMORY= 0;
public static final int org.dvb.smartcard.SmartCardReaderEvent.SMART_CARD_ERROR = 3;
public static final int org.dvb.smartcard.SmartCardReaderEvent.SMART_CARD_IN = 0;
public static final int org.dvb.smartcard.SmartCardReaderEvent.SMART_CARD_MUTED = 2;
public static final int org.dvb.smartcard.SmartCardReaderEvent.SMART_CARD_OUT = 1;
```

# 17 Internet access clients

This feature is optional in all profiles of GEM.

## 17.1 Referencing DVB services and content within WWW content

GEM internet access clients shall support the following mechanisms of referencing DVB services and content from within WWW content.

- Use of a reference to a DVB service in an HTML "href" shall launch that DVB service in the same way as is done by GEM applications using the GEM service selection API or the end user of the GEM terminal using the navigator to select DVB services. GEM internet access clients are not covered by the GEM security model and shall be able to perform service selection and tuning without any consideration of permissions. The normal GEM resource management mechanisms shall be applied if tuning is required and the tuner concerned is reserved.

NOTE: Only GEM applications with `SelectPermission` can start an internet access client. Such applications can in any case use the service selection API to also select broadcast services. Hence this does not weaken the GEM security model.

GEM applications shall only be started in response to selection of a reference to a DVB service. All GEM applications run in the scope of a DVB service and cannot run on their own outside of a service of some type.

## 17.2 Minimum requirements for internet clients

Table 66 defines some minimum network protocols which shall be supported by those internet client applications which form part of the GEM internet access profile.

**Table 66**

| Internet Client Application | Minimum required protocols |
|---|---|
| WWW browser | HTTP (see clause 6.3.7.2, "GEM profile of HTTP 1.0")<br>HTTPS (see clause 6.3.7.3, "HTTPS") |
| Email client | SMTP for sending email (see RFC 821 [71]) or HTTP to a WebMail server.<br>Protocols for receiving email are implementation dependent. |
| Usenet news | NNTP (RFC 977 [72]) or HTTP to a WebNews server. |

WWW browser internet clients shall support use of the "mailto" URL as defined in RFC 2818 [73] and RFC 2368 [74].

Email internet clients shall support use of the "http" and "https" URLs as defined in section 3.2.2 of RFC 2616 [45].

If a Usenet news internet client is present, then the WWW browser and E-mail internet clients shall also support the use of the "news" URL as defined by RFC 1738 [61]. Usenet news internet clients, if present, shall support the use of the "http", "https", and "mailto" URLs.

## 17.3 Internet streamed media

There is no requirement for GEM terminals to support internet streaming media content however where these are both supported and visible to DVB-J applications, it shall be done using the Java Media Framework (Java TV [16]).

## 17.4 Internet connection management

Internet client applications shall always use the default ISP configured into the GEM application. For example, in a GEM terminal with only one `RCInterface` that is a `ConnectionRCInterface`, this is the same default as defined by

`ConnectionRCInterface.setTargetToDefault`. If this default ISP is reached via a `ConnectionRCInterface` that is currently connected to another target, the normal GEM resource management mechanism shall apply with the internet client application having the resource priority of the GEM application that launched it. If the internet client application fails to reserve such a `ConnectionRCInterface`, the service selection shall fail with a `SelectionFailedEvent` with reason code `INSUFFICIENT_RESOURCES`.

# Annex A (normative): External references; errata, clarifications and exemptions

## A.1 Void

## A.2 Errata to DAVIC

The following errata to DAVIC [17] shall apply.

### A.2.1 org.davic.media.MediaTimeEventControl - deregistering listeners

The following shall be considered to be added to the description of the main interface description of `org.davic.media.MediaTimeEventControl`:

- It is an implementation's responsibility to deregister any registered instances of `MediaTimeEventListener` at an appropriate time, e.g. when the Xlet is destroyed.

The following shall be considered to be added to the description of the `notifyWhen(org.davic.media.MediaTimeEventListener i, long mediaTime, int id)` method:

- When this method is called with a `listener`, an `id` and a negative `mediaTime` as arguments, the listener is deregistered for the negated value of the corresponding `mediaTime` parameter and matching id.
- The availability of this deregistration feature on the platform may be indicated via the existence of a system property.
- Calling this method with a value that does not match a previously registered positive media time shall have no effect.

NOTE 1: When this method is called with a `mediaTime` value of 0 the result is implementation dependent.

NOTE 2: When an application calls `notifyWhen` more than once with the same `mediaTime` and `id`, it is implementation dependent if more than one event is generated and whether multiple deregistrations will be required.

The following shall be considered to be added to the description of the `notifyWhen(org.davic.media.MediaTimeEventListener i, long mediaTime)` method:

- When this method is called with a `listener` and a negative `mediaTime` as arguments, the `listener` is deregistered for the negated value of the corresponding `mediaTime` parameter.
- The availability of this deregistration feature on the platform may be indicated via the existence of a system property.

NOTE 3: A deregistration via this method is equivalent to calling `notifyWhen(org.davic.media.MediaTimeEventListener i, long mediaTime, int id)` with an `id` value of 0.

NOTE 4: Although this class isn't required by any profile of GEM, GEM terminal specifications may include it as a mandatory or optional element. For example, BD-J is known to require this class.

## A.2.2 org.davic.resources.ResourceClient.requestRelease

The following text in the description of the `requestRelease()` method:

- A call to this operation informs the `ResourceClient` that another application has requested the resource accessed via the proxy parameter.

Shall be considered to be replaced with:

- A call to this operation informs the `ResourceClient` that another `ResourceClient` instance has requested the resource accessed via the proxy parameter.

## A.2.3 org.davic.mpeg

### A.2.3.1 General

The following classes are considered to have a no-argument protected constructor:

- `ElementaryStream.`
- `Service.`
- `TransportStream.`

### A.2.3.2 NotAuthorizedException

The specification is considered to include `org.davic.mpeg.NotAuthorizedException` as specified below:

# org.davic.mpeg

# NotAuthorizedException

#### Syntax

```
public class NotAuthorizedException extends java.lang.Exception implements
org.davic.mpeg.NotAuthorizedInterface
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.davic.mpeg.NotAuthorizedException
```

#### All Implemented Interfaces:

```
NotAuthorizedInterface, java.io.Serializable
```

#### Description

This class is thrown by MPEG related APIs when access is requested to information which is scrambled and to which access is not permitted by the security system.

Constructors:

#### NotAuthorizedException()

```
public NotAuthorizedException()
```

Constructs a NotAuthorizedException with no detail message.

### NotAuthorizedException(String)

```
public NotAuthorizedException(java.lang.String s)
```

Constructs a NotAuthorizedException with the specified detail message.

Parameters:

`s` - the detail message.

Methods:

### getElementaryStreams()

```
public ElementaryStream[] getElementaryStreams()
```

If getType() returns ELEMENTARY_STREAM, then this method returns the set of ElementaryStreams that could not be descrambled. Otherwise it returns null.

Specified by:

`NotAuthorizedInterface.getElementaryStreams()` in interface `NotAuthorizedInterface`

Returns:

either the set of ElementaryStreams that could not be descrambled or null.

### getReason(int)

```
public int[] getReason(int index)
```

Returns the reason(s) why descrambling was not possible.

Specified by:

`NotAuthorizedInterface.getReason(int)` in interface `NotAuthorizedInterface`

Parameters:

`index` - If the component to which access failed is a Service, index shall be 0. Otherwise index shall refer to one stream in the set returnedby getElementaryStreams().

Returns:

an array of length 2 where the first element of the array is the major reason and the second element of the array is the minor reason.

Throws:

`IndexOutOfBoundsException` - If the component to which access failed is a Service, this exception will be thrown if index is non zero. If the component(s) to which access failed was a (set of) elementary streams then this exception will be thrown where index is beyond the size of the array returned by getElementaryStreams.

See also:

`NotAuthorizedInterface.getElementaryStreams()`

### getService()

```
public Service getService()
```

If getType() returns SERVICE, then this method returns the Service that could not be descrambled. Otherwise it returns null.

Specified by:

`NotAuthorizedInterface.getService()` in interface `NotAuthorizedInterface`

Returns:

either the Service that could not be descrambled or null.

getType()

```
public int getType()
```

Specified by:

`NotAuthorizedInterface.getType()` in interface `NotAuthorizedInterface`

Returns:

SERVICE or ELEMENTARY_STREAM to indicate that either a service (MPEG program) or one or more elementary streams could not be descrambled.

# A.2.4 Chapter 9, "Application Format"

## A.2.4.1 Section 9.4.7, "The MPEG-2 Section Filter API"

In the description of this class there are 2 instances of a cross reference to DAVIC part 10, section 115.3. In each case this shall be considered as a reference to DAVIC part 10, section 12.5.3.

# A.2.5 org.davic.mpeg.sections

## A.2.5.1 RingSectionFilter

Is considered to have the following text appended to its description:

- All sections in a ring section filter are initialized to empty when the ring section filter is first created. Clearing them to empty any time after this is the responsibility of the application. Starting a ring section filter shall not clear any of the sections to empty.

## A.2.5.2 Section

### A.2.5.2.1 clone()

Section is considered to have the method clone() with the following behaviour.

- A cloned Section object is a new and separate object. It is unaffected by changes in the state of the original Section object or restarting of the SectionFilter the source Section object originated from. The clone method must be implemented without declaring exceptions.

### A.2.5.2.2 getData()

Remove the following text from the methods of `org.davic.mpeg.sections.Section`:

- (everything after the length field, not including a CRC check).

### A.2.5.2.3 getFullStatus()

Is considered to have the following text appended to its description:

- Returns true when the `Section` object contains valid data.

#### A.2.5.2.4 Class Description

The following text shall be considered to be present at the end of the class description:

- In this class, the term "section data" shall be considered to include the section header, any possible CRC and all bytes in between.

### A.2.5.3 SectionFilter

#### A.2.5.3.1 Cross reference error

In the description of this class there are 12 instances of a cross reference to H7 of DAVIC [17]. In each case this shall be considered as a reference to E.8.1.

#### A.2.5.3.2 startFiltering(all signatures)

In these methods, the description of when the `FilterResourceException` shall be thrown is considered to have the following text appended to its description:

- This shall be applied whether the parent section filter group is connected to a TS or not, or whether this SectionFilter object is already started or not.

#### A.2.5.3.3 startFiltering(java.lang.Object, int, int, int, byte[], byte[])

Is considered to have the following text appended to its description:

- `IllegalFilterDefinitionException` is thrown if offset is too small.

#### A.2.5.3.4 startFiltering (appData, pid, tableId) exceptions

Like other `startFiltering` methods `org.davic.mpeg.sections.SectionFilter.startFiltering (appData, pid, tableId)` shall throw an `IllegalFilterDefinitionException` where:

- the Java integer is negative;
- the Java integer is larger than what is allowed for PID or table_id according to the relevant MPEG specification.

#### A.2.5.3.5 Started Section Filters

The class description is considered to have the following text added at its end.

- When a SectionFilterGroup is detached, either by the client or through resource withdrawal, started SectionFilters shall remain started. Hence if the SectionFilterGroup is re-attached, those filters shall re-activate.

### A.2.5.4 SectionFilterGroup

#### A.2.5.4.1 attach

A `NotAuthorizedException` is added to the definition of this method.

#### A.2.5.4.2 Constructors

The constructors are considered to have.

- Throws `IllegalArgumentException` if `numberOfFilters` <1.

#### A.2.5.4.3 sectionSize

All methods with a `sectionSize` parameter are considered to have the following text appended to their descriptions:

- Throws `IllegalArgumentException` if `sectionSize` <1.

#### A.2.5.4.4 newRingSectionFilter

Is considered to have the following text appended to its description:

- Throws `IllegalArgumentException` if `ringSize` <1.

#### A.2.5.4.5 Constructor(int, boolean)

Is considered to have the following text appended to its description:

- The scope of the `resourcePriority` shall be a single application only.

#### A.2.5.4.6 General

In the methods of this class, all occurrences of "(successful startFiltering)" shall be considered to read "(successful call to the startFiltering method of the newly created instance)".

### A.2.5.5 TimeOutEvent

The specification is considered to include `org.davic.mpeg.sections.TimeOutEvent` as specified below:

# org.davic.mpeg.sections
# TimeOutEvent

Declaration

```
public class TimeOutEvent extends org.davic.mpeg.sections.EndOfFilteringEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.davic.mpeg.sections.org.davic.mpeg.sections.SectionFilterEvent
              |
              +--org.davic.mpeg.sections.org.davic.mpeg.sections.EndOfFilteringEvent
                    |
                    +--org.davic.mpeg.sections.TimeOutEvent
```

All Implemented Interfaces:

`java.io.`Serializable

Description

This event is generated if section filter operations time out within the period specified by the setTimeOut() method. For a SimpleSectionFilter it will be generated if no sections arrive within the specified period. For a TableSectionFilter, it will be generated if the complete table does not arrive within the specified time. For a RingSectionFilter, it will be generated if the specified time has elapsed since the arrival of the last section being successfully filtered.

Constructors:

TimeOutEvent(SectionFilter, Object)

```
public TimeOutEvent(SectionFilter f, java.lang.Object appData)
```

This constructs an TimeOutEvent event for the specified SectionFilter object.

Parameters:

`f` - the SectionFilter object which timed out.

`appData` - application data that was passed to the startFiltering method.

Methods:

getSource()

```
public java.lang.Object getSource()
```

This returns the SectionFilter object which timed out.

Overrides:

`EndOfFilteringEvent.getSource()` in class `EndOfFilteringEvent`

# A.2.6 Simple Section Filter

The description of the getSection method shall be considered to have the following text added:

- This method shall return null if no section has been successfully filtered for this instance and the owning SectionFilterGroup is not attached.

# A.2.7 org.davic.media

## A.2.7.1 FreezeControl.resume()

Add the following to the description of the semantics for this method:

- If the player is started and if decoding of the media stream is not frozen then calls to this method shall have no effect.
- If the player is not started then the exception shall be thrown.

## A.2.7.2 MediaTimePositionChangedEvent

Add the following constructor:

```
MediaTimePositionChangedEvent (
    Controller from,
    int previous,
    int current,
    int target,
    Time mediaTime)
```

With the following definition of parameters:

Parameters:

- from - the controller whose media position was changed;
- previous - the state the controller was in before this event;
- current - the state the controller was in at the time the event was generated;
- target - the state that the controller is heading to;
- mediaTime - the media time after the change.

## A.2.7.3 NotAuthorizedMediaException

The following constructors are considered to be removed from the specification:

```
NotAuthorizedMediaException()
NotAuthorizedMediaException(java.lang.String reason)
```

The following constructors are considered to be a normative part of the specification:

```
NotAuthorizedMediaException( org.davic.mpeg.Service, int reason )
NotAuthorizedMediaException( org.davic.mpeg.ElementaryStream[], int reason[] )
NotAuthorizedMediaException( org.davic.mpeg.ElementaryStream[], int[], int[] )
NotAuthorizedMediaException(Service, int major_reason,int minor_reason)
```

### NotAuthorizedMediaException(ElementaryStream[], int[])

```
public NotAuthorizedMediaException(org.davic.mpeg.ElementaryStream[] e, int[] reason)
```

Constructor for exception due to failure accessing one or more MPEG elementary streams The caller of this constructor is responsible for ensuring the two arrays provided as parameters are the same size. The implementation is not expected to check this. The exception has no detail message.

Parameters:

`e` - the elementary streams which could not be accessed

`reason` - the reason why the exception was thrown for each elementary stream.

Use of the constructor NotAuthorizedMediaException(ElementaryStream[] e, int[] reason) will result in the major reason for each elementary stream being the one specified in the reason parameter to the method and the minor reason being OTHER as defined in NotAuthorizedInterface.

### NotAuthorizedMediaException(Service, int)

```
public NotAuthorizedMediaException(org.davic.mpeg.Service s, int reason)
```

Constructor for exception due to failure accessing an MPEG service. The exception has no detail message.

Parameters:

`s` - the service which could not be accessed.

`reason` - the reason why the service could not be accessed.

Use of the constructor NotAuthorizedMediaException(Service, int reason) will result in the major reason for the service being the one specified in the reason parameter to the method and the minor reason being OTHER as defined in NotAuthorizedInterface.

NotAuthorizedMediaException(ElementaryStream[], int[], int[])

```
public NotAuthorizedMediaException(ElementaryStream[] e, int[] major_reason,int[]
minor_reason)
```

Constructor for exception due to failure accessing one or more MPEG elementary streams The caller of this constructor is responsible for ensuring the three arrays provided as parameters are the same size. The implementation is not expected to check this.

Parameters:

`e` - the elementary streams which could not be accessed.

`major_reason` - the major reason why the exception was thrown for each elementary stream.

`minor_reason` - the minor reason why the exception was thrown for each elementary stream.

### NotAuthorizedMediaException(Service, int, int)

```
public NotAuthorizedMediaException(org.davic.mpeg.Service s, int major_reason,
int minor_reason)
```

Constructor for exception due to failure accessing an MPEG service.

Parameters:

`s` - the service which could not be accessed.

`major_reason` - the major reason why the service could not be accessed.

`minor_reason` - the minor reason why the service could not be accessed.

Since:

MHP 1.0.2

## A.2.7.4 LanguageControl

### A.2.7.4.1 Class description

The following description of semantics is considered to be present:

- If more than one stream with the same language exists, the behaviour of `selectLanguage(String)` is to select the first listed in the network signalling.

NOTE: This is equivalent to item b under clause 11.4.2.3, "Default media player behaviour".

In the methods `getCurrentLanguage` and `selectDefaultLanguage`, the following additional text shall be considered to be present:

- If no content of the appropriate media type for the control is present, a String of length zero is returned.

In the method `selectDefaultLanguage`, the following shall be considered to be appended to the method description:

- If this information is not available, a String of length zero is returned.

### A.2.7.4.2 selectLanguage(String)

The following additional text shall be considered to form part of the description of this method:

- When this method returns normally, the language will have been synchronously selected.

# A.2.8 org.davic.net

## A.2.8.1 InvalidLocatorException

The following class definition is considered to be a normative part of the specification:

# org.davic.net

# InvalidLocatorException

Syntax

```
public class InvalidLocatorException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.davic.net.InvalidLocatorException
```

All Implemented Interfaces:

java.io.Serializable

Description

This exception is thrown when one or more parameters to construct a Locator are invalid.

Constructors:

InvalidLocatorException()

```
public InvalidLocatorException()
```

Constructor without reason.

InvalidLocatorException(String)

```
public InvalidLocatorException(java.lang.String reason)
```

Constructor for the exception with a specified reason.

Parameters:

`reason` - the reason why the exception was raised.

## A.2.8.2 Locator

### A.2.8.2.1 Locator()

The no-argument constructor for `org.davic.net.Locator` is considered to not be present. The absence of any description on the method indicates that this was an editing error in the DAVIC specification.

### A.2.8.2.2 toExternalForm()

Is considered to have the following text appended to its description:

- If the instance of `Locator` has been created using `Locator(java.lang.String url)` and the URL is a non-null invalid URL the behaviour is implementation dependent.

## A.2.8.3 tuning

### A.2.8.3.1 NetworkInterfaceController

#### A.2.8.3.1.1 reserve()

The semantic of the following throws clause is corrected as follows:

```
Throws:     NoFreeInterfaceException
            raised if the requested network interface can not be reserved
```

The following from the semantic of this method:

- If this NetworkInterfaceController has already reserved another NetworkInterface, then it will either release that NetworkInterface and reserve the specified one, or throw an exception. If the specified NetworkInterface has already been reserved by this NetworkInterfaceController, then this method does nothing.

is replaced with the following:

- If this NetworkInterfaceController has currently reserved another NetworkInterface, then it will either release that NetworkInterface and reserve an appropriate one, or throw an exception. If a NetworkInterface that is able to tune to the specified transport stream is currently reserved by this NetworkInterfaceController, then this method does nothing.

#### A.2.8.3.1.2 reserveFor()

The following from the semantic of this method:

- If this NetworkInterfaceController has already reserved another NetworkInterface, then it will either release that NetworkInterface and reserve an appropriate one, or throw an exception. If NetworkInterfaceController has already reserved a NetworkInterface that is able to tune to the specified transport stream, then this method does nothing.

is replaced with the following:

- If this NetworkInterfaceController has currently reserved another NetworkInterface, then it will either release that NetworkInterface and reserve an appropriate one, or throw an exception. If a NetworkInterface that is able to tune to the specified transport stream is currently reserved by this NetworkInterfaceController, then this method does nothing.

#### A.2.8.3.1.3 tune()

Replace "this NetworkInterface" with "the NetworkInterface reserved by this NetworkInterfaceController" in section H.5.4.3 of DAVIC specification.

### A.2.8.3.2 NetworkInterface

#### A.2.8.3.2.1 Protected constructor

This class is considered to have a no argument protected constructor with the following statement attached to it:

- This constructor is provided for the use of implementations and specifications which extend this specification. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

## A.2.9 org.davic.net.tuning

### A.2.9.1 Figure H-1

In figure H-1, "Tuning API object model: Base classes", in the class `NetworkInterface`, the method `getURL()` shall be considered to be "`getLocator()`".

### A.2.9.2 Figure H-2

In:

- figure H-2: "Tuning API object model: Exceptions".

The exception `IncorrectURLException` shall be considered to be absent.

### A.2.9.3 Network Interface

The following additional methods are considered to be present:

```
/**
 * Tests for equality where two NetworkInterface objects are equal if and only if
 * they control the same physical tuner.
 * @param other the reference object with which to compare.
 * @return true if and only if other is an instance of NetworkInterface and calls to other
 * control the same physical tuner as this.
 */
public boolean equals(Object other)

/**
 * Returns a hash code value for the object. which obeys the contract of Object.hashCode().
 * @return a hash code value for the object.
```

```
 */
public int hashCode()
```

### A.2.10 Extensibility and Over-Riding

The DAVIC specification DAVIC 1.4.1p9 [17] is considered to include the following;

- The addition of public or protected constructors, methods or fields to the `org.davic` packages is allowed except where this would cause a compliant MHP application to fail. Examples of the latter include the adding of abstract methods to classes or adding of methods to interfaces which application classes will implement.
- Within the `org.davic` package and its sub packages, overriding inherited public and protected methods not specified as being overridden is an allowable implementation option.

NOTE: This language does not apply to constructors or to fields, because inherited fields and constructors cannot be overridden in the Java language.

## A.3 Additional GEM requirements on Java TV

### A.3.1 javax.tv.util.TVTimerSpec

NOTE: Java TV 1.1 [16] requires that the argument to deschedule(TVTimerSpec) be the instance passed as argument to the method TVTimer.schedule. Implementations of GEM terminal specifications based on Java TV 1.1 are required to obey this JavaTV-defined behavior. This may be implemented in a way consistent with the GEM requirements by implementing TVTimer.schedule such that it always returns the instance passed in as argument. Note that Java TV's specification of TVTimer.schedule allows the implementation to modify the instance passed in as argument, e.g. to account for timer granularity.

### A.3.2 javax.media.Manager

#### A.3.2.1 getDataSourceList()

In the JavaDoc for javax.media.Manager.getDataSourceList() the following text:

> The first name in the list will always be:
>
> `media.protocol.<protocol>DataSource`

shall be considered to be replaced with:

> The first name in the list will always be:
>
> `media.protocol.<protocol>.DataSource`

#### A.3.2.2 getHandlerClassList()

In the JavaDoc for javax.media.Manager.getHandlerClassList() the following text:

```
public static java.util.Vector getHandlerClassList(java.lang.String contentName)
```

> Build a list of `Handler/CODE>` classes from the `content-prefix-list` and a `content` name.
>
> The first name in the list will always be:
>
> `media.content.<contentName>.Handler`
>
> Each additional name looks like:
>
> `<content-prefix>.media.content.<contentName>.Player`

for every <content-prefix> in the content-prefix-list.

Parameters:

`contentName` - The content type to use in the class name.

Returns:

A vector of strings where each one is a `Player` class-name.

shall be considered to be replaced with:

```
public static java.util.Vector getHandlerClassList(java.lang.String contentName)
```

Build a list of `Handler` classes from the `content-prefix-list` and a `content` name.

The first name in the list will always be:

```
media.content.<contentName>.Handler
```

Each additional name looks like:

```
<content-prefix>.media.content.<contentName>.Handler
```

for every `<content-prefix>` in the content-prefix-list.

Parameters:

`contentName` - The content type to use in the class name.

Returns:

A vector of strings where each one is a `Player` class-name.

# A.4 HAVi

With reference to HAVi [50].

## A.4.1 Drafting conventions

The HAVi specification does not follow the ETSI drafting rules as specified in SR 001 262 [i.10]. In particular, the term "should" shall be interpreted in terms of the ETSI drafting rules as being closer to "shall" than "should".

## A.4.2 General

### A.4.2.1 Thread-safety

As with the `java.awt package`, the `org.havi.ui` package is not required to be thread safe.

### A.4.2.2 javadoc errors

The javadoc for the HAVi specification appears to be compiled against a version of the "java.*" packages more recent than that required for the MHP specification. All methods, fields and inner classes shown as inherited from classes outside the "org.havi" package namespace which are not present in the MHP specification for that class shall be considered to be absent from the HAVi specification.

## A.4.3 Event mechanism

This clause presents a clarification for the HAVi event mechanism.

### A.4.3.1 Introduction

One of the reasons for the HAVi event mechanism is to provide a mechanism for widgets to request a type of input that may not be available on the input device supported by a platform. A HAVi widget can indicate its input preference by implementing one or more `HxxxInputPreferred` interfaces. This is important to enable application provided widgets to have access to the same capabilities as have always been available to platform provided ones.

An `HxxxInputPreferred` interface indicates the widget that implements the interface expects a certain type of input events. E.g. on systems with limited input devices the platform can supply a virtual keyboard to generate the interface-specific events.

The `Hxxxable` and `HxxxValue` interfaces group methods common to various `HVisible` subclasses; they do not imply additional support from the platform.

The `HxxxInputPreferred` interfaces all imply possible support by a (virtual) keyboard. The platform shall provide a way for the user to generate the input expected by the focused component. This could mean automatically presenting some kind of virtual keyboard when a focused component requires input that could not be generated in another way. The platform can determine which virtual keys a widget requires by looking at the `HxxxInputPreferred` interfaces it implements.

For `keyCode` input, `HNavigationInputPreferred.getNavigationkeys()` and `HKeyboardInputPreferred.getValidInput()` allow the platform to determine which `keyCodes` the particular widgets expect. Platform implementations shall provide the user with a means to generate the `keyCodes` expected by all `HxxxInputPreferred` interfaces implemented by the widget having input focus (with the exception of `keyCodes` not supported according to `org.havi.ui.event.HKeyCapabilities`).

HAVi events generated by the platform and sent to a `HComponent` are `HActionEvent`, `HAdjustmentEvent`, `HItemEvent`, `HTextEvent`, `HRcEvent`, `HFocusEvent` and `HKeyEvent`.

### A.4.3.2 Overview of HAVi events

`HComponents` indicate they want to receive `HFocusEvents` by implementing the `HNavigationInputPreferred` interface. For `HNavigationInputPreferred`, the `keyCodes` for navigation keystrokes generated on the `HNavigationInputPreferred` will be passed to the `HNavigationInputPreferred` as an `HFocusEvent` `transferId`. The `HNavigationInputPreferred` will process an `HFocusEvent` of id `FOCUS_TRANSFER` in its `processHFocusEvent` method by calling `requestFocus` on the `HNavigable` associated with the `keyCode` returned by `HFocusEvent.getTransferId()`, if there is one.

Implementations of `HNavigationInputPreferred` will not send this `HFocusEvent` to `FocusListeners` or `HFocusListeners`, it is only used by the platform to communicate the transferId `to` the `HNavigationInputPreferred`.

The manner in which the keystrokes are generated is implementation-specific, as some platforms may supply a more extensive hardware keyboard, while other platforms may choose to provide a virtual keyboard to generate these keystrokes.

Also, HAVi allows generation of the HAVi events on top of AWT (in `HComponent.processEvent`) from other events as an implementation option, to allow for platforms unable to generate HAVi event directly. As a result of this, it is platform specific whether `processEvent` is ever called with a HAVi event as argument, or that the event is generated in this method from other events. In such an implementation, a widget may receive the original, unexpected, platform-specific, non-HAVi `java.awt` events as well as the translated HAVi events. To avoid confusion, a HAVi widget is supposed to process only the HAVi events and applications should ignore other events (i.e. not add listeners for `java.awt` events and not use them through `processXxxEvent` methods).

For the handling of navigation keys, an implementation of the HNavigationInputPreferred interface receives the keycodes for navigation as the transferId of org.havi.ui.event.HFocusEvent, and not as a java.awt.event.KeyEvent or org.havi.ui.event.HKeyEvent. (On platforms which generate HAVi events from other events, it is possible that the component does receive a java.awt.event.KeyEvent, however this is platform-specific and applications should not rely on this).

Hcomponent.processEvent is responsible to dispatch HFocusEvents to the processHFocusEvent method; this method will call requestFocus on the target of a transferId for a FOCUS_TRANSFER event, and send HFocusEvents of type FOCUS_GAINED and FOCUS_LOST to any registered HFocusListeners for classes that support adding these listeners. Depending on which other HAVi interfaces the component implements and on whether it extends HVisible, this method will also change the state of the widget, play appropriate sounds, and notify any listeners specified in these other interfaces for a focus change.

An HComponent implementing HKeyboardInputPreferred will receive input in the form of HKeyEvents. On platforms with limited input devices, the platform can present a virtual keyboard for the user to generate the requested keyCodes. The platform can use the input type of the HKeyboardInputPreferred to determine which keys to present on this virtual keyboard, however, the platform is not required to prevent the user from generating more keyCodes than those requested by the HKeyboardInputPreferred.

HAVi allows to generate the HKeyEvents from other events in HComponent.processEvent, so any KeyEvents received by the component which are not HKeyEvents should be ignored as platform-specific.

Hcomponent.processEvent is responsible for dispatching HKeyEvents to the processHKeyEvent method; this method will send the HKeyEvent to any registered HKeyListeners for classes that support adding these listeners.

If an HComponent implements both HNavigationInputPreferred and HKeyboardInputPreferred, the platform will provide a way for the user to generate the keyCodes for both HFocusEvents and HKeyEvents in an unambiguous way (e.g. while in edit mode, keyCodes are converted into HKeyEvents, otherwise they generate HFocusEvents). The getEditMode method can be called by the platform to determine the current input mode of the widget. The platform will provide a platform specific way to switch between any different input modes as required by any limitations of its input devices, and notify the widget of this change by sending it an appropriate START_CHANGE or END_CHANGE event.

If the same keyCode is expected for different input types (e.g. VK_ENTER is set as navigation code while the widget also expects it as an HKeyEvent), then the platform will present the user with means to separately generate each of the HAVi events for that keyCode (in the case of the example, an HFocusEvent with transferId VK_ENTER and an HKeyEvent). Application authors should realize that to use keyCodes which are likely to have multiple functions may force some platforms to present a less than perfectly user-friendly display.

For platforms that generate HAVi events in HComponent.processEvent, extra events may be received in addition to the HAVi events. For example, for HFocusEvent FOCUS_GAINED and FOCUS_LOST, if the HFocusEvent is generated directly, processEvent should send this HFocusEvent to both processHFocusEvent and to processFocusEvent (the second one will happen in the superclass method). If the platform generates HAVi events in processEvent, when a FocusEvent is received by HComponent.processEvent, it will be sent to processFocusEvent and a new HFocusEvent will be created to send to processHFocusEvent.

For the other HAVi events, like HActionEvent, HAdjustmentEvent, HItemEvent and HTextEvent events, the behaviour is analogous. The platform will provide a way for the user to generate the events, either directly or from other events (translated in HComponent.processEvent), on HComponents that implement the HActionInputPreferred, HAdjustmentInputPreferred, HSelectionInputPreferred, and HKeyboardInputPreferred interfaces, respectively.

Implementations of the interfaces do not have to override processEvent so as to send these events to the appropriate processHxxEvent method, this will be the responsibility of Hcomponent.processEvent. The processHxxEvent method will call any registered listeners and perform any other specified actions, such as changing states, playing sounds, etc.

HAdjustmentInputPreferred and HSelectionInputPreferred, which extend HOrientable, allow the platform to select a way for the user to generate HAdjustmentEvents and HItemEvents which best matches the component's representation.

### A.4.3.3 Relation between HAVi events and AWT events

Applications which extend from `java.awt.Component` (and not `org.havi.ui.HComponent`) and which register event listeners from the `java.awt.event` package shall receive the behaviour defined for the `java.awt.event` package. If events from the `org.havi.ui.event` package are generated on such a `Component`, these shall be sent to any registered listeners according to the behaviour defined for their parent class in `java.awt.event`.

HAVi allows implementations that generate `HxxxEvents` from other events, so an `HComponent` may receive the `HxxxEvent` in addition to those other events. For example, you could get both `havi.ui.event.HKeyEvents` sent to `HKeyListeners` and `awt.event.KeyEvents` sent to AWT KeyListeners.

For subclasses that extend from `Component` and not from `HComponent`, it is implementation dependent whether they receive the events defined in `awt.event` or a subclass of these events defined in havi (e.g. `org.havi.ui.event.HKeyEvent` instead of `java.awt.event.KeyEvent`). However, in the latter case the behaviour will be as defined for the parent class, e.g. an `org.havi.ui.event.HKeyEvent` will go to `KeyListeners` added with `Component.addKeyListener`. Also, applications cannot assume that such `Components` will be able to receive `KeyEvents` with `keyCodes` other than those specified in the MHP minimum profile.

### A.4.3.4 Application guidelines

Any implementation of a `HxxxInputPreferred` interface shall implement the processing of the `HxxxEvent` and calling of listeners in the `processHxxEvent` method (where this event processing is not inherited from a parent class). It is the platforms' responsibility to provide a way in which the user can generate the events requested by the widget. Since the HAVi events may be generated in the `processEvent` method on some platforms, interface implementers should not assume that the `processEvent` method will receive the HAVi events on every platform. On platforms where the HAVi events are generated in `processEvent`, it is the platform's responsibility to call the `processHxxEvent` method with the HAVi event.

## A.4.4 org.havi.ui

### A.4.4.1 HActionable

The class `HListGroup` is considered to be removed from the list of "Platform Classes".

#### A.4.4.1.1 getActionSound

The description of this method shall be considered to be replaced with the following

- Return the last action sound set by the `setActionSound` method or null if no action sound has been set.

#### A.4.4.1.2 Class description

The following sentence immediately before the heading "Interaction States":

- HAVi action events are discussed in detail in the HActionInputPreferred interface description.

Shall be modified to read:

- HAVi action events are discussed in detail in the HActionEvent interface description.

#### A.4.4.1.3 setActionCommand

The parameters clause for the "command" parameter shall be considered to be extended with the following sentence:

- To remove the command specify a null command parameter.

## A.4.4.2 HActionInputPreferred

The description of this method shall be considered to have the following text added:

- If this `HActionInputPreferred` has no action command then an empty string shall be returned.

## A.4.4.3 HAdjustmentInputPreferred

### A.4.4.3.1 Interface description

In the interface description of `HAdjustmentInputPreferred` after the sentence:

- For platforms with a restricted number of physical keys this may involve a "virtual keyboard" or similar mechanism.

is considered to be extended by the following text:

- The system might use the information returned by the method `getOrientation()` of the super interface to select appropriate key mappings for this event. The mechanisms to generate this event shall not be effective while the component is disabled (see `HComponent.setEnabled()`).

Also, the following text:

- All interoperable implementations of the `HAdjustmentInputPreferred` interface must extend `HComponent`.

Is considered to be replaced with:

- Widgets of HAVi compliant applications implementing the `HAdjustmentInputPreferred` interface must have `HComponent` in their inheritance tree.

Also, in the following text:

- Note that the `java.awt.Component` method `isFocusTraversable` should always return true for a `java.awt.Component` implementing this interface.

The word "should" is considered to be replaced with "shall".

### A.4.4.3.2 getAdjustMode

The last sentence of the description of `getAdjustMode()` is considered to be extended with:

- Note that these events are ignored, if the component is disabled. See also: `HComponent.setEnabled()`.

### A.4.4.3.3 proccessHAdjustmentEvent

The description of `proccessHAdjustmentEvent()` is considered to be extended with:

- Widgets implementing this interface shall ignore `HAdjustmentEvents`, while the component is disabled. See also: `HComponent.setEnabled()`.

### A.4.4.3.4 setAdjustMode

The description of `setAdjustMode()` is considered to be extended with:

- Calls to this method shall be ignored, if the component is disabled. See also: `HComponent.setEnabled()`.

## A.4.4.4 HAdjustmentValue

### A.4.4.4.1 Class description

The class `HListGroup` is considered to be removed from the list of "Platform Classes".

### A.4.4.4.2 getAdjustmentSound

This method is considered to have the following text added to the method description.

- null shall be returned if this method is called before its corresponding set method.

### A.4.4.4.3 getBlockIncrement

This method is considered to have the following text added to the method description.

- 1 shall be returned if this method is called before its corresponding set method.

### A.4.4.4.4 getUnitIncrement

This method is considered to have the following text added to the method description.

- 1 shall be returned if this method is called before its corresponding set method.

## A.4.4.5 HAnimateEffect

### A.4.4.5.1 getRepeatCount

This method shall be considered to have the following text added to the method description.

- Except for `HAnimateEffect` implementations that specify a different default, `getRepeatCount()` returns `REPEAT_INFINITE` if no call to `setRepeatCount()` has previously been made.

## A.4.4.6 HBackgroundDevice

### A.4.4.6.1 setBackgroundConfiguration

In the `setBackgroundConfiguration` method, the sentence below shall be considered to be added to the description of this method:

- Applications can prevent or limit changes to configurations of other, not intended, devices by using constants ZERO_GRAPHICS_IMPACT, ZERO_VIDEO_IMPACT and ZERO_BACKGROUND_IMPACT in their configuration templates.

The following shall be added as the first item in the bulleted list:

- If an application tries to select a configuration which is not valid for that device at that time or when the device is in a particular mode then an HConfigurationException shall be thrown.

### A.4.4.6.2 getConfigurations

The following shall be added to the end of the method description.

- The set of configurations returned may include ones which are only valid for the device at particular times or when the device is in a particular mode.

#### A.4.4.6.3 Constructor

In the constructor of this class, the following sentence:

- It is not intended that applications should directly construct `HBackgroundDevice` objects.

Shall be considered to be replaced with the following:

- An interoperable application shall not subclass the `HBackgroundDevice` class.

### A.4.4.7 HBackgroundImage

#### A.4.4.7.1 Constructors - HBackgroundImage(String), HBackgroundImage(URL)

The sentence:

- Loading of the data for the object is not required at this time.

shall be considered to be replaced with:

- Loading of the data for the object shall not happen at this time.

#### A.4.4.7.2 load

The following text is considered to be added to the description of this method:

- Multiple calls to load shall each add an extra listener, all of which are informed when the loading is completed. If load is called with the same listener more than once, the listener shall then receive multiple copies of a single event.

#### A.4.4.7.3 Constructor(byte[])

In the following sentence:

- If the byte array does not contain a valid image then this constructor shall throw a java.lang.IllegalArgumentException.

"shall" shall be considered to say "may".

### A.4.4.8 HComponent

#### A.4.4.8.1 processEvent

The following text is a clarification of the HAVi specification:

The implementation of the method `HComponent.processEvent()` shall ensure that key events which are translated to HAVi events shall not be reported to `processKeyEvent()` or reported to `KeyListeners`. Key events which are not translated to HAVi events shall be reported to `processKeyEvent()` and `KeyListeners` as defined in the Java specification.

NOTE: If applications override `processEvent` they may terminally disturb these processes. Applications should not do this without extreme care, as the results may be very implementation dependent.

### A.4.4.9 HComponentOrdering

#### A.4.4.9.1 addAfter, addBefore

In the returns clause, the text:

- is successfully added.

shall be considered to read:

- is successfully added or was already present.

## A.4.4.10 HEventMulticaster

### A.4.4.10.1 Class description

The following text is considered to be added to the description of this class:

- The `HEventMulticaster` class is intended to assist platform or subclass implementers with the handling of HAVi events. Implementations are not required to use this class to dispatch HAVi events. Applications should not extend the `HEventMulticaster` class and implementations are not required to behave correctly if an application does extend this class. If an extended multicaster is desired, `AWTEventMulticaster` should be used rather than `HEventMulticaster`.

Also, extend the following paragraph in the class description:

- It is an implementation option for this class to insert other classes in the inheritance tree (for example `java.awt.AWTEventMulticaster`). It is allowed that this may result in `HEventMulticaster` inheriting additional methods beyond those specified here.

with:

- If this class does extend `java.awt.AWTEventMulticaster`, it is allowed for the fields defined in this class to be inherited from that parent class.

### A.4.4.10.2 Constructor

The following text is considered to form part of the constructor description:

- The parameters a and b passed to the constructor shall be used to populate the fields a and b of the instance.

### A.4.4.10.3 remove

In the description of this method, replace:

- returns the resulting multicast listener;

with:

- returns the result.

## A.4.4.11 HFontCapabilities

### A.4.4.11.1 getSupportedCharacterRanges

The returns clause of this method shall be considered to be extended with the following text:

- including where the capabilities of the font are unknown.

The following paragraph in the method description:

- Support for a character range does not imply that ALL characters within that range are available in the specified font.

Shall be considered to be extended with the following extra sentence:

- Support does require that at lease one character from the range is included in the font.

# A.4.4.12 HGraphicsDevice

## A.4.4.12.1 getBestConfiguration

The following text shall be considered to be added to the end of the second paragraph of the method description:

- If there are such equally best configurations, the one which is returned by this method is an implementation dependent selection from among those which are equally best.

## A.4.4.12.2 setGraphicsConfiguration

In the `setGraphicsConfiguration` method, the sentence below:

- Applications can prevent or limit changes to configurations of other, not intended, devices by using constants `ZERO_GRAPHICS_IMPACT` and `ZERO_VIDEO_IMPACT` in their configuration templates.

Shall be considered to be extended as follows:

- Applications can prevent or limit changes to configurations of other, not intended, devices by using constants `ZERO_GRAPHICS_IMPACT`, `ZERO_VIDEO_IMPACT` and `ZERO_BACKGROUND_IMPACT` in their configuration templates.

The following shall be added as the first item in the bulleted list:

- If an application tries to select a configuration which is not valid for that device at that time or when the device is in a particular mode then an HConfigurationException shall be thrown.

## A.4.4.12.3 getConfigurations

The following shall be added to the end of the method description.

- The set of configurations returned may include ones which are only valid for the device at particular times or when the device is in a particular mode.

## A.4.4.12.4 Constructor

In the constructor of this class, the following sentence:

- It is not intended that applications should directly construct `HGraphicsDevice` objects.

Shall be considered to be replaced with the following:

- An interoperable application shall not subclass the `HGraphicsDevice` class.

# A.4.4.13 HGraphicsConfigTemplate

## A.4.4.13.1 setPreference

The following text shall be added to the third paragraph of the method description:

- Calling this method with null for the object parameter shall have no effect if the preference is not currently set in the template.

# A.4.4.14 HListElement

## A.4.4.14.1 Class description

The class description is considered to be extended by the following:

- The methods `setIcon()` and `setLabel()` of `HListElement` shall not be used for elements, which are part of `HListGroup`. If an application requires to alter the content, it shall either replace the entire element, or remove it temporarily and re-add it after the content was changed.

# A.4.4.15 HListGroup

## A.4.4.15.1 Class description

In the following text:

- The `HListGroup` is a user interface component representing a list of selectable items (`HListElements`) which contain static read-only graphical and / or textual content.

The phrase "static read-only" is considered to be removed.

Also, extend the following text:

- Interoperable HAVi applications shall not add `HListElement` more than once. If an application requires items with identical contents (label and/or icon), then additional items shall be created. The behaviour of the `HListGroup` if duplicates are added is implementation specific.

With:

- The methods `setIcon()` and `setLabel()` of `HListElement` shall not be used for elements, which are part of `HListGroup`. If an application requires to alter the content, it shall either replace the entire element, or remove it temporarily and re-add it after the content was changed.

Also, in the text:

- the minimum size is the size to present one element or an implementation specific minimum (32 x 32 for example) if no elements are present.

The phrase "if no elements are present" is considered to be replaced with:

- if label and icon size are not set (see `HListGroupLook.getMinimumSize()`).

Also, under the parameters table under heading "Default parameter values exposed in the constructors" the description of the parameter "items" missing is considered to read:

- The initial list of elements for this `HListGroup` or `null` for an empty list.

## A.4.4.15.2 Fields

### A.4.4.15.2.1 ITEM_NOT_FOUND

In the description:

- A constant which may be returned from `getIndex` if the requested element is not found in the content.

The word "may" is considered to be replaced by "shall".

#### A.4.4.15.2.2 ADD_INDEX_END

In the description:

- A constant for use with `addItem` and `addItems` which specifies that the new items should be appended to the end of the list.

The word "should" is considered to be replaced by "shall".

### A.4.4.15.3 Use of "action" and "actioning"

The use of the words "action" and "actioning" in the following methods in `HListGroup` does not imply any connection between `HListGroup` and `HActionable`. There is no such connection in the API:

- `getCurrentIndex()`
- `getCurrentItem()`
- `setCurrentItem()`
- `getSelectionIndices()`
- `getSelection()`

### A.4.4.15.4 addItem(HListItem item, int index)

In the description of parameter "index" all occurrences of the word "items" is considered to be replaced by "item".

### A.4.4.15.5 addItem(HListItem items[], int index)

In the parameter description:

- item - the item to add.

is considered to be replaced by:

- items - the items to add.

### A.4.4.15.6 getIconSize

The description of this method is considered to be extended by:

- If label and icon size do not match the size per element, the associated `HListGroupLook` is allowed to use other sizes during the rendering process. This size shall be used by `HListGroupLook` to calculate the size per element.

### A.4.4.15.7 getLabelSize

The description of this method is considered to be extended by:

- If label and icon size do not match the size per element, the associated `HListGroupLook` is allowed to use other sizes during the rendering process. This size shall be used by `HListGroupLook` to calculate the size per element.

### A.4.4.15.8 getOrientation

The following text:

- The orientation controls how an associated `HLook` lays out the component and affects the visual behaviour of the `HAdjustmentEvent` and `HItemEvent` events. `HListGroups` do not receive `HAdjustmentEvents`.

Is considered to be replaced by:

- The orientation controls how an associated `HLook` lays out the component and affects the visual behaviour of `HItemEvent` events.

### A.4.4.15.9 setFocusTraversal

In all cases the phrase "then null should be specified" is considered to be replaced by "then null shall be specified".

### A.4.4.15.10 setItemSelected

The following text:

- If a successful call to this method causes the selection to change an HItemEvent shall be sent to any registered listeners. If the selection does not change then no HItemEvent shall be sent.

Shall be considered to be replaced with:

- If a call to this method causes an item to become deselected an HItemEvent with an ID of ITEM_CLEARED shall be sent to all registered listeners. This can happen because either sel is false or this HListGroup is not in multi selection mode.
- If a call to this method causes a non selected item to become a selected item then an HItemEvent with an ID of ITEM_SELECTED shall be sent to all registered listeners.

### A.4.4.15.11 setIconSize

The description of this method is considered to be extended by:

- If label and icon size do not match the size per element, the associated `HListGroupLook` is allowed to use other sizes during the rendering process. This size shall be used by `HListGroupLook` to calculate the size per element.

In the description of the size parameter, the following text:

- Dimension(DEFAULT_ICON_SIZE, DEFAULT_ICON_SIZE).

is considered to be replaced by

- Dimension(DEFAULT_ICON_WIDTH, DEFAULT_ICON_HEIGHT).

### A.4.4.15.12 setLabelSize

The description of this method is considered to be extended by:

- If label and icon size do not match the size per element, the associated `HListGroupLook` is allowed to use other sizes during the rendering process. This size shall be used by `HListGroupLook` to calculate the size per element.

### A.4.4.15.13 setListContent

The sentence:

- Any existing selection is discarded (which may cause an HItemEvent to be generated).

Is considered to be replaced by

- If any elements are selected, then the selection is discarded and an HItemEvent is generated.

The following text:

- Set the list content for this HListGroup. If any elements are selected, then the selection is discarded and an HItemEvent is generated.

Shall be considered to be replaced with:

- Set the list content for this HListGroup. If any items are selected, then the selection shall be discarded and an HItemEvent with an ID of ITEM_SELECTION_CLEARED shall be generated and sent to all registered listeners.
- If elements is null then the current active item index shall be set to ITEM_NOT_FOUND. If elements is non null then the current active item index shall be set to 0. An HItemEvent with an ID of ITEM_SET_CURRENT shall be sent to all registered listeners.

### A.4.4.15.14 setMove

In all cases the phrase "then null should be specified" is considered to be replaced by "then null shall be specified".

### A.4.4.15.15 setScrollPosition

This method is considered to have the following text added:

Parameters:

- scroll - the scroll position.

### A.4.4.15.16 setSelectionMode

The parameter in the method signature called `adjust` shall be considered to be called `edit` as used in the parameter list.

### A.4.4.15.17 removeItem

The following text:

- Remove the HListElement at the specified index. The item is also removed from the selection, if any is set. If this was the last item in the selection the entire selection is destroyed and calls to getSelection shall return null until new content and selections are created.
- If the act of removing an item causes the current active item index to change, an HItemEvent shall be sent.
- If the act of removing an item causes the selection to change, an HItemEvent shall be sent.

Shall be considered to be replaced with the following text:

- Removes the HListElement at the specified index. All following items are shifted.
- If the item is the only HListElement in this HListGroup then the current active item index shall be set to ITEM_NOT_FOUND. If the removal of the item causes a change in the current active item index then an HItemEvent with an ID of ITEM_SET_CURRENT shall be generated and sent to all registered listeners.
- If the item is selected then it shall be removed from the selection and an HItemEvent with an ID of ITEM_CLEARED shall be generated and sent to all registered listeners.

### A.4.4.15.18 removeAllItems

The following text:

- Remove all the content. The selection is also destroyed and calls to getSelection shall return null until new content and selections are created.

Shall be considered to be replaced with:

- Removes all the content. If any items are selected, then the selection shall be discarded and an HItemEvent with an ID of ITEM_SELECTION_CLEARED shall be generated and sent to all registered listeners.

- The current active item index shall be set to ITEM_NOT_FOUND and an HItemEvent with an ID of ITEM_SET_CURRENT shall be generated and sent to all registered listeners.

#### A.4.4.15.19 setCurrentItem

The following paragraph shall be considered to form part of the description of this method.

- If index is valid for this HListGroup then the current active item index shall be set to index. If this causes a change in the current active item index then an HItemEvent with an ID of ITEM_SET_CURRENT shall be generated and sent to all registered listeners.

Under the Returns subsection, the following paragraph:

- true if the current item was changed, false if index was not a valid index for this HListGroup or the current item was not changed because it is already selected. No exception is thrown if index is not valid.

Shall be considered to be replaced with:

- true if the current item was changed, false if index was not a valid index for this HListGroup or the current item was not changed because it is already the current item. No exception is thrown if index is not valid.

#### A.4.4.15.20 setMultiSelection

The following text:

- Note that if the HListGroup is switched out of multiple selection mode and more than one item is selected, the selection shall change so that the first of the items is selected and the others are deselected. This will cause an HItemEvent to be sent to any registered listeners.

Shall be considered to be replaced with:

- Note that if the HListGroup is switched out of multiple selection mode and more than one item is selected, the selection shall change so that the first of the items is selected and the others are deselected. An HItemEvent with an ID of ITEM_CLEARED shall be sent to all registered listeners for each deselected item.

### A.4.4.16 HListGroupLook

#### A.4.4.16.1 getMaximumSize

The description of `getMaximumSize()` is considered to be replaced by the following:

- Returns the size to present all elements of the specified `HVisible` plus any additional dimensions that the `HListGroupLook` requires for border decoration etc. If no elements are present, a dimension object is returned with width and height set to `java.lang.Short.MAX_VALUE`.

- The extra space required for border decoration can be determined from the `getInsets()` and `getElementInsets()` methods. The behaviour is not defined for the case, when a subclass overrides these methods. Application developers shall not assume any influence on the returned dimensions.

- The size per element shall be determined by calls to `getIconSize()` and `getLabelSize()` of `HListGroup`. If any of the values requests a default as specified by `DEFAULT_ICON_WIDTH`, `DEFAULT_ICON_HEIGHT`, `DEFAULT_LABEL_WIDTH` and `DEFAULT_LABEL_HEIGHT`, then an implementation specific default is used for the corresponding value(s).

Parameters:

- visible HVisible to which this HLook is attached.

  Returns:

  - A dimension object indicating this HListGroupLook's maximum size. See also:

    ```
    HListGroup.setIconSize()
    HListGroup.setLabelSize()
    HVisible.getMaximumSize()
    ```

### A.4.4.16.2 getMinimumSize

The description of getMinimumSize() is considered to be replaced by the following:

- Returns the size to present one element of the specified HVisible plus any additional dimensions that the HListGroupLook requires for border decoration, etc.
- The extra space required for border decoration can be determined from the getInsets() and getElementInsets() methods. The behaviour is not defined for the case, when a subclass overrides these methods. Application developers shall not assume any influence on the returned dimensions.
- The size per element shall be determined by calls to getIconSize() and getLabelSize() of HListGroup. If any of the dimensions requests a default as specified by DEFAULT_ICON_WIDTH, DEFAULT_ICON_HEIGHT, DEFAULT_LABEL_WIDTH and DEFAULT_LABEL_HEIGHT, then an implementation specific default is used for the corresponding value(s).

Parameters:

- visible HVisible to which this HLook is attached.

  Returns:

  - A dimension object indicating this HListGroupLook's minimum size. See also:

    ```
    HListGroup.setIconSize()
    HListGroup.setLabelSize()
    HVisible.getMinimumSize()
    ```

### A.4.4.16.3 getPreferredSize

The description of getPreferredSize() is considered to be replaced by the following:

- Gets the preferred size of the HVisible component when drawn with this HListGroupLook.
- If a default size for width and height was set with HVisible.setDefaultSize(), then the dimensions are rounded down to the nearest element (minimum of one) according to the orientation of the associated HListGroup, and any dimensions for border decorations etc. are added.
- If no default size was set or only for one dimension (i.e. height is NO_DEFAULT_HEIGHT or width is NO_DEFAULT_WIDTH), then the unset dimension(s) shall be sufficiently large to present five elements according to the HListGroup's orientation. Any dimensions for border decoration etc. are added.
- The extra space required for border decoration can be determined from the getInsets() and getElementInsets() methods. The behaviour is not defined for the case, when a subclass overrides these methods. Application developers shall not assume any influence on the returned dimensions.
- The size per element shall be determined by calls to getIconSize() and getLabelSize() of HListGroup. If any of the values requests a default as specified by DEFAULT_ICON_WIDTH, DEFAULT_ICON_HEIGHT, DEFAULT_LABEL_WIDTH and DEFAULT_LABEL_HEIGHT, then an implementation specific default is used for the corresponding value(s).

Parameters:

- visible `HVisible` to which this `HListGroupLook` is attached.

  Returns:

  - A dimension object indicating the preferred size of the `HVisible` when drawn with this `HListGroupLook`. See also:

```
HListGroup.setIconSize()
HListGroup.setLabelSize()
HVisible.getPreferredSize()
HVisible.setDefaultSize()
```

### A.4.4.16.4 getValue

The first paragraph of `getValue()` is considered to be extended by the following:

- A non-null value represents the scroll position of the associated `HListGroup`. The value shall never be less than zero.

And the description of returns is considered to be replaced by the following:

- the non-negative scroll position associated with the specified pointer position or null

### A.4.4.16.5 hitTest

In the description of `hitTest()` after

- then this method will return the index of that element.

is considered to be extended by the following:

- The `HListGroup` shall interpret this index as current item. If the value is `ADJUST_THUMB`, then the caller shall use `getValue()` to retrieve the actual scroll position corresponding to the specified pointer position.

And the returns description considered to be extended by the following:

- or a non-negative element index.

### A.4.4.16.6 showLook

The description of `showLook()` is considered to be replaced by the following:

- This method is responsible for repainting the entire visible including the list content set on the `HListGroup`, and the visible background, subject to the clipping rectangle of the graphics object passed to it.
- If the method modifies the clipping rectangle of the graphics object, it shall restore the original rectangle upon return.
- `showLook()` paints the visible with its current background colour according to the `getBackgroundMode()` method of `HVisible` and draws any (implementation-specific) borders, regardless whether content is available or not. Note that by default the background mode is set to not paint a background. Furthermore on platforms which support transparent colours the background colour may be partially or completely transparent.
- Any resources explicitly associated with this look shall be loaded by it during its creation. Note that the "standard" looks do not load content by default.
- This method is called from the `paint()` method of `HVisible` and must never be called from elsewhere. Components wishing to redraw themselves shall call their repaint method in the usual way or call `widgetChanged()` under certain circumstances.

- The labels of the associated HListElements shall be rendered by using the current text layout manager of the HListGroup. For each visible label is the render() method of HTextLayoutManager called. The position and size per label are specified as insets relatively to the bounds of HListGroup. Note that the bounds are independent of any borders of the HListGroup, but the insets have to include the borders per element, if any. The look shall use the method getLabelSize() of HListGroup to determine the size for each label. If the returned dimension object has DEFAULT_LABEL_WIDTH for the width and/or DEFAULT_LABEL_HEIGHT for the height as values, then this method shall use implementation specific value(s) as default(s) for the missing dimension(s) instead. If getTextLayoutManager() returns null, then labels shall not be rendered.

- If supported, scaling of icons shall reflect the resize mode of the visible within the area of the respective list element. The look shall use the method getIconSize() of HListGroup to determine the size for each icon. If the returned dimension object has DEFAULT_ICON_WIDTH for the width and/or DEFAULT_ICON_HEIGHT for the height as values, then this method shall use implementation specific value(s) as default(s) for the missing dimension(s) instead.

- Except for the alignment of labels and sizes of labels and icons, it is explicitly not defined, how this look arranges icons and labels within the elements' areas. Additionally, it is an implementation option to render labels and icons in other sizes than specified, if the available size per element is smaller or larger than label and icon size. It is also not defined, how the look presents the current item and selected items, or the current selection mode. The elements shall be layed out as specified by getOrientation() of the associated HListGroup.

- When the associated HListGroup contains more elements than presentable, the look shall make the user aware of that condition, e.g. by displaying an additional scrollbar reflecting the current scroll position. Again, the visible means by which this information is conveyed is not defined and implementation dependent. It is an implementation option for HListGroupLook to draw elements before the scroll position, in order to fill the available space.

- The behaviour of this method, when a subclass overrides the methods getInsets() or getElementInsets(), is not defined. Application developers shall not assume that the corresponding borders will appear as specified.

  Parameters:

  - g the graphics context.
  - visible the visible.
  - state the state parameter indicates the state of the visible, allowing the look to render the appropriate content for that state. Note that the default behaviour of HListGroupLook is to ignore this parameter, since the content of HListGroup is not state based. See also:

```
java.awt.Component.getBounds()
java.awt.Graphics.getClipBounds()
HListGroup.getCurrentItem()
HListGroup.getListContent()
HListGroup.getOrientation()
HListGroup.getScrollPosition()
HListGroup.setIconSize()
HListGroup.setLabelSize()
HListGroup.isItemSelected()
HTextLayoutManager.render()
```

### A.4.4.16.7 Class description

The following sentence shall be considered to be removed.

- Borders are not drawn around the content.

### A.4.4.16.8 showLook and renderVisible

The five paragraphs in showLook starting with "The labels of the associated HListElements shall be rendered..." and ending "...Application developers shall not assume that the corresponding borders will appear as specified." shall be considered to form part of the renderVisible method instead.

#### A.4.4.16.9 renderVisible

The following shall be considered to be added to the end of this method description:

- The org.havi.ui.HExtendedLook.renderVisible method is responsible for painting any implementation specific borders for each HListElement as well as drawing of an additional scrollbar if required.

## A.4.4.17 HLook

### A.4.4.17.1 General

In the description of `HLook` and all implementing classes, the term `clipRect` shall be interpreted to mean "clipping rectangle".

### A.4.4.17.2 Class description

Only the first paragraph of the "Invocation Mechanism" section shall be considered to be present.

### A.4.4.17.3 getPreferredSize

In the description of this method, the text:

- "... then the return value is the size of the largest piece of content..."

shall be considered to be replaced with:

- "...then the return value is a size that is sufficiently large to hold each piece of content..."

The equivalent replacement shall apply for `getMinimumSize()` and `getMaximumSize()`.

In `HLook` and all classes implementing this interface, the following text shall be considered to form part of the method description of getPreferredSize():

- If a default preferred size has been set for this `HVisible` (using `setDefaultSize(Dimension)` ) and the default preferred size has an `NO_DEFAULT_WIDTH` then the return value is a `Dimension` with this height (obtained with `getDefaultSize()` ) and the preferred width for the content plus any additional dimensions that the `HLook` requires for border decoration etc.

- If a default preferred size has been set for this `HVisible` (using `setDefaultSize(Dimension)` ) and the default preferred size has an `NO_DEFAULT_HEIGHT` then the return value is a `Dimension` with this width (obtained with `getDefaultSize()` ) and the preferred height for the content plus any additional dimensions that the `HLook` requires for border decoration etc.

### A.4.4.17.4 showLook

In `org.havi.ui.HLook.showLook()`, the text:

- The `showLook` method should not modify the `clipRect` of the `Graphics` object that is passed to it.

Shall be clarified as follows:

- The `showLook` method shall not modify the `clipRect` (clipping rectangle) of the `Graphics` object passed to it in a way which includes any area not part of that original `clipRect`. If any modifications are made, the original `clipRect` shall be restored.

## A.4.4.18 HMultilineEntry

The following text shall be considered to be added to the class description:

- A call to the inherited method `setDefaultLook(HSinglelineEntry)` shall behave the same as a call to `HSinglelineEntry.setDefaultLook(HSinglelineEntry)`.

## A.4.4.19 HMultilineEntryLook

### A.4.4.19.1 getCaretCharPositionForLine

The following text:

- If an invalid line is specified an `IllegalArgumentException` is thrown If it cannot be moved the nearest position should be returned.

shall be considered to be replaced by:

- If an invalid line is specified an `IllegalArgumentException` is thrown. If the caret cannot be moved to the same column position on this line, the nearest position should be returned.

### A.4.4.19.2 getSoftLineBreakPositions

The following text shall be considered to be added to the returns clause of this method:

- If there is no text content within the `HVisible`, a zero length array shall be returned.

### A.4.4.19.3 getVisibleSoftLineBreakPositions

The following text is considered be added to the returns clause of this method:

- If there is no text content within the `HVisible`, a zero length array shall be returned.

## A.4.4.20 HNavigable

### A.4.4.20.1 Class description

In the following sentence:

- Applications should assume that classes which implement `HNavigable` can only generate events of the type `HFocusEvent` in response to other types of input event.

The word "only" shall be considered to be not present.

### A.4.4.20.2 setMove

All occurrences of "then null should be specified" shall be considered to be replaced by "then null shall be specified".

### A.4.4.20.3 set FocusTraversal

All occurrences of "then null should be specified" shall be considered to be replaced by "then null shall be specified".

## A.4.4.21 HOrientable

### A.4.4.21.1 Interface description

The following text:

- All interoperable implementations of the `HOrientable` interface must extend `HComponent`.

Is considered to be replaced with:

- Widgets of HAVi compliant applications implementing the `HOrientable` interface must have `HComponent` in their inheritance tree.

#### A.4.4.21.2 getOrientation

The following text:

- The orientation controls how an associated `HLook` lays out the component and affects the visual behaviour of the `HAdjustmentEvent` and `HItemEvent` events. For example, the system might use this information to select appropriate key mappings for these events.

Is considered to be replaced with:

- The orientation controls the layout of the component.

#### A.4.4.21.3 setOrientation

The following text:

- The orientation controls how the associated `HLook` lays out the component.

Is considered to be replaced with:

- The orientation controls the layout of the component.

##### A.4.4.21.3.1 Orientation constants

In the description of the orientation constants: `ORIENT_LEFT_TO_RIGHT`, `ORIENT_RIGHT_TO_LEFT`, `ORIENT_TOP_TO_BOTTOM` and `ORIENT_BOTTOM_TO_TOP` the word "shall" is considered to replace "should' in each instance of the phrase "should be rendered".

### A.4.4.22 HScene

#### A.4.4.22.1 addAfter, addBefore

In the returns clause, the text:

- is successfully added;

shall be considered to read:

- is successfully added or was already present.

#### A.4.4.22.2 getFocusOwner

The following text shall be considered to be added to the end of the method description:

- if and only if this HScene is active.

#### A.4.4.22.3 Rendering behavior

The following text shall be considered to form an additional paragraph under this heading.

- An application which sets both of setBackgroundMode(NO_BACKGROUND_FILL) and setImageMode(IMAGE_NONE) shall be responsible for ensuring that all pixels in the Hscene are filled with a value which is either opaque or transparent only to video. Such an application cannot make any assumptions about the previous contents of the graphics objects into which it is drawing. For implementations which double-buffer the display of graphics, these existing contents are implementation dependent.

#### A.4.4.22.4 show

The following text shall be considered to be appended to the description of this method:

- This method does not cause the HScene to request the input focus.

#### A.4.4.22.5 setVisible

The following text shall be considered to be present on the end of the method description;

- If this HScene is already visible, then this method brings it to the front. The semantics of show() and setVisible(true) are identical.

## A.4.4.23 HScreenConfigurationListener

The parameter "gce" for the report method shall be considered to have the following description.

- gce - The event notifying the listener of the modification.

## A.4.4.24 HSceneFactory

#### A.4.4.24.1 getDefaultHScene

The text:

- identical to calling org.havi.ui.HScene.getDefaultHscene(

shall be considered to read:

- identical to calling org.havi.ui.HSceneFactory.getDefaultHscene(

#### A.4.4.24.2 Class Description

The following text:

- Calling resizeScene for an HScene shall apply the same policies as described above for newly created HScenes when deciding whether the resizing operation requested is possible.

Shall be considered to read:

- Calling resizeScene for an HScene shall apply the same policies as described above for newly created HScenes when deciding whether the method call is possible.

## A.4.4.25 HSelectionInputPreferred

#### A.4.4.25.1 Interface description

The following text is considered to be added to the interface description:

- The system must provide a means of generating `HItemEvent` events as necessary. For platforms with a restricted number of physical keys this may involve a "virtual keyboard" or similar mechanism.The system might use the information returned by the method `getOrientation()` of the super interface to select appropriate key mappings for this event. The mechanisms to generate this event shall not be effective while the component is disabled (see `HComponent.setEnabled()`).

Also, the text:

- All interoperable implementations of the `HSelectionInputPreferred` interface must extend `HComponent`.

Is considered to be replaced with:

- Widgets of HAVi compliant applications implementing the `HSelectionInputPreferred` interface must have `HComponent` in their inheritance tree.

Also, in the text:

- Note that the `java.awt.Component` method `isFocusTraversable` should always return true for a `java.awt.Component` implementing this interface.

The word "should" is considered to be replaced with "shall".

#### A.4.4.25.2 getSelectionMode

The description of this method is considered to be extended by the following text:

- Note that these events are ignored, if the component is disabled. See also: `HComponent.setEnabled()`.

#### A.4.4.25.3 processHItemEvent

The description of this method is considered to be extended by the following text:

- Widgets implementing this interface shall ignore HItemEvents, while the component is disabled. See also: `HComponent.setEnabled()`.

#### A.4.4.25.4 setSelectionMode

The description of this method is considered to be extended by the following text:

- Calls to this method shall be ignored, if the component is disabled. See also: `HComponent.setEnabled()`.

### A.4.4.26 HSingleLineEntry

#### A.4.4.26.1 Class description

The term:

- `TEXT_START_CHANGE` event

is considered to be replaced with:

- an `HTextEvent` event with an id of `TEXT_START_CHANGE`.

#### A.4.4.26.2 Default parameter values not exposed in the constructors

In the row "Password protection (the echo character)", in the "default value" column, replace

- Entry is "clear", i.e. not password protected.

with:

- Zero (ASCII NUL), i.e. not password protected.

#### A.4.4.26.3 setType(int)

The following text:

- Set the type of permitted keyboard entry.

  Parameters:

  - type - one of INPUT_ANY, INPUT_ALPHA, INPUT_NUMERIC or INPUT_CUSTOMIZED

Shall be considered to be replaced with:

- Set to indicate to the system which input keys are required by this component. The input type constants can be added to define the union of the character sets corresponding to the respective constants.

Parameters:

- type - sum of one or several of INPUT_ANY, INPUT_NUMERIC, INPUT_ALPHA, or INPUT_CUSTOMIZED.

#### A.4.4.26.4 getValidInput

The following text:

- Retrieve the customized input character range. The return value of this method should reflect the range of input keys which the component wishes to see, should getType return a value with the INPUT_CUSTOMIZED bit set. This method may return null if customized input is not requested.

Shall be considered to be replaced with:

- Retrieve the customized input character range. If getType returns a value with the INPUT_CUSTOMIZED bit set then this method shall return an array containing the range of customized input keys. If the range of customized input keys has not been set then this method shall return a zero length char array. This method shall return null if getType returns a value without the INPUT_CUSTOMIZED bit set.

## A.4.4.27 HStaticAnimation

The following text is considered to be added to the description of this class:

- Calling `setVisible(false)` on `HStaticAnimation` shall not automatically stop the animation hence applications are not required to call the `play()` method again when the animation again becomes visible. It is implementation dependent whether the animation continues from the last visible position or from where it would be if it kept running.

## A.4.4.28 HStaticRange

### A.4.4.28.1 Fields

#### A.4.4.28.1.1 SCROLLBAR_BEHAVIOR

The description of this field is considered to be replaced by the following text:

- The `HStaticRange` shall behave as a scrollbar, i.e. the allowable values that may be set / returned for the `HStaticRange` shall be affected by the "thumb" offsets, and hence its value shall be able to vary between `[minimum + minThumbOffset, maximum - maxThumbOffset]`.

#### A.4.4.28.1.2 SLIDER_BEHAVIOR

The description of this field is considered to be replaced by the following text:

- The `HStaticRange` shall behave as a slider, i.e. the allowable values that may be set / returned for the `HStaticRange` shall not be affected by the "thumb" offsets, and hence its value shall be able to vary between `[minimum, maximum]`.

### A.4.4.28.2 getOrientation

The following text:

- The orientation controls how an associated `HLook` lays out the component and affects the visual behaviour of the `HAdjustmentEvent` and `HItemEvent` events. For example, the system might use this information to select appropriate key mappings for these events.

Is considered to be replaced by:

- The orientation controls how the associated `HLook` lays out the component.

#### A.4.4.28.3 setBehavior

The following text is considered to be appended to the method description:

- If the new behaviour is `SCROLLBAR_BEHAVIOR` and the control's settings for range and thumb offsets are illegal, i.e. `minimum` + `thumbMinOffset` is equal or greater than `maximum` - `thumbMaxOffset`, then an `IllegalArgumentException` shall be thrown. If the control's value is not valid for the offsets, then the value shall be changed to the closest valid value.

#### A.4.4.28.4 setRange

The following text is considered to be appended to the method description:

- If the maximum is greater than the minimum and the value of the control is outside the new range (subject to the control's current behaviour), then the value is changed to the closest valid value.

#### A.4.4.28.5 setThumbOffsets

The following text is considered to be appended to the method description:

- If this control's behaviour is `SCROLLBAR_BEHAVIOR`, then the following rules apply: If the thumb offsets are illegal, i.e. `minimum` + `thumbMinOffset` is equal or greater than maximum - `thumbMaxOffset`, then an `IllegalArgumentException` shall be thrown. If the control's value is not valid for the specified offsets, then the value shall be changed to the closest valid value.

#### A.4.4.28.6 setValue

The method's parameter description is considered to be replaced by:

- value - the value for this HStaticRange.

Also, the following text is considered to be appended to the method description:

- If the specified value is not valid, then the method shall round it to the closest valid value. An application can retrieve the corrected value by means of method `getValue()`.

### A.4.4.29 HVideoDevice

#### A.4.4.29.1 getBestConfiguration

The following text is considered to be added to the end of the second paragraph of the descriptions of both methods of this name.

- If there are such equally best configurations, the one which is returned by this method is an implementation dependent selection from among those which are equally best.

#### A.4.4.29.2 setVideoConfiguration

In the `setVideoConfiguration` method, the sentence below:

- Applications can prevent or limit changes to configurations of other, not intended, devices by using constants ZERO_GRAPHICS_IMPACT and ZERO_VIDEO_IMPACT in their configuration templates.

shall be considered to be extended as follows:

- Applications can prevent or limit changes to configurations of other, not intended, devices by using constants ZERO_GRAPHICS_IMPACT, ZERO_VIDEO_IMPACT and ZERO_BACKGROUND_IMPACT in their configuration templates.

The following additional text is considered to be present.

NOTE: If a configuration which includes ZERO_GRAPHICS_IMPACT or ZERO_BACKGROUND_IMPACT and this would require changes to already running devices then this will not be possible to apply successfully and hence this method will return False.

#### A.4.4.29.3 getVideoController

The following text:

- HPermissionDeniedException - (HPermissionDeniedException) if the application does not currently have the right to get the VideoPlayer object.

shall be considered to be replaced by:

- HPermissionDeniedException - (HPermissionDeniedException) this exception shall never be thrown.

#### A.4.4.29.4 setVideoConfiguration

The following shall be added as the first item in the bulleted list:

- If an application tries to select a configuration which is not valid for that device at that time or when the device is in a particular mode then an HConfigurationException shall be thrown.

#### A.4.4.29.5 getConfigurations

The following shall be added to the end of the method description.

- The set of configurations returned may include ones which are only valid for the device at particular times or when the device is in a particular mode.

#### A.4.4.29.6 getVideoSource

The following text:

- HPermissionDeniedException - if the application does not currently have the right to get the `VideoSource` object.

Shall be considered to be replaced by:

- HPermissionDeniedException - this exception shall never be thrown.

#### A.4.4.29.7 Constructor

In the constructor of this class, the following sentence:

- It is not intended that applications should directly construct `HVideoDevice` objects.

Shall be considered to be replaced with the following:

- An interoperable application shall not subclass the `HVideoConfiguration` class.

### A.4.4.30 HVisible

#### A.4.4.30.1 Class description

In the class description of `HVisible`, the sentence:

- Most content is stored by reference, but text content is copied as `java.lang.String` objects are immutable.

Shall be considered to not be present.

In the table "Default parameter values not exposed in the constructors", row "getDefaultSize", column "Default value" the following text:

- The default preferred size not set (i.e. null) unless specified by width and height parameters.

Shall be considered to be replaced with:

- The default preferred size not set (i.e. NO_DEFAULT_SIZE) unless specified by width and height parameters.

### A.4.4.30.2 Constructors

The following text shall be considered to form part of those constructor descriptions which have an `HLook` as a parameter:

- Applications shall not use `HLooks` with this constructor unless those `HLooks` are specified as working with `HVisible`. If an `HLook` is used which is specified as only working with specific sub-classes of `HVisible` then the failure mode is implementation dependent.

### A.4.4.30.3 setDefaultSize

The following text:

- If this parameter or specifies a size smaller than an implementation-defined minimum size a `java.lang.IllegalArgumentException` will be thrown.

Shall be considered to be replaced by:

- If this parameter specifies a size smaller than an implementation-defined minimum size, the preferred size of this component shall be set to that implementation-defined minimum size.

Also, the following text:

- If the parent Container into which the `HVisible` is placed has no layout manager this method has no effect.

Is considered to be removed.

### A.4.4.30.4 setLookData

The method description shall be considered to have the following text added.

- If for this specified key a data object has been set, the old data object shall be replaced with the new one.

### A.4.4.30.5 setResizeMode

The following text added shall be considered to be replaced:

- Set the scaling mode for scaling any state-based content rendered by an associated HLook. If content is not used in the rendering of this HVisible calls to this method shall change the current alignment mode, but this will not affect the rendered representation.

by:

- Set the resize mode of this HVisible. If the associated HLook does not render content or if scaling is not supported, changing the mode may have no visible effect.

### A.4.4.30.6 setTextContent

In the description of this method, the following sentence is considered to not be present:

- Note that unlike `setGraphicContent`, `setAnimateContent` and `setContent`, the content is copied as it is not possible to store a reference to a `java.lang.String`.

#### A.4.4.30.7 NO_DEFAULT_SIZE

The following text shall be considered to be appended to the end of the field description.

- The contents of the Dimension object cannot be relied upon. Comparisons must always be done using object identity, i.e. using the "==" operator.

### A.4.4.31 HVideoConfigTemplate

#### A.4.4.31.1 setPreference

The paragraph:

- An application which wishes to remove a preference from an existing template (e.g. one generated by the platform) may call this method with null for the object parameter.

Shall be considered to be extended with the following sentence:

- Specifying null as the object parameter shall have no effect if the preference is not in the template.

### A.4.4.32 HVideoComponent

#### A.4.4.32.1 HAVi Specification

In section 8.3.3.6 "Integrating HAVi Video Support into Platforms" the following two sentences shall be removed:

- "The class HVideoComponent is intended to be returned by a platform specific controller for video. In platforms based on the Java Media Framework, the Player.getVisualComponent method shall return objects of this class".

Note that the `HVideoComponent` class must be present, and MHP terminals may choose to use it or not. Interoperable applications should not use `HVideoComponent`.

#### A.4.4.32.2 removeOnScreenLocationModifiedListener

In this method, the parameters clause shall be changed from:

- slml - listener to be notified when the on-screen location of the component is modified.

to:

- slml - listener to be removed

### A.4.4.33 HBackgroundConfiguration

#### A.4.4.33.1 setColor

The "throws" clause for `HPermissionDeniedException` shall be considered to read as follows

- If the application has not currently reserved the HBackgroundDevice associated with this configuration or this configuration is not the current configuration of that HBackgroundDevice.

#### A.4.4.33.2 getConfigTemplate

The following text shall be considered to be added to the description of this method:

- Preferences that are not filled in by the platform will return DONT_CARE priority.

#### A.4.4.33.3 Constructor

In the description of the constructor, the following sentence:

- It is not intended that applications should directly construct `HBackgroundConfiguration` objects.

shall be considered to read;

- An interoperable application shall not subclass the `HBackgroundConfiguration` class.

## A.4.4.34 HScreenDevice

### A.4.4.34.1 reserveDevice

The following paragraph:

- Once the right to control this device has been granted and not removed in the intervening period further calls to this method shall have no effect and return true.

Shall be considered to be replaced by:

- Once the right to control this device has been granted and not removed in the intervening period further calls to this method re-using the current resource client shall have no effect and return true.

## A.4.4.35 HImageMatte

### A.4.4.35.1 setMatteData

The method `setMatteData` shall be considered to have the following text added:

- Note that if the size of the image is smaller than the size of the component to which the matte is applied, the empty space behaves as if it were an opaque flat matte of value 1.0. By default images are aligned at the top left corner of the component. This can be changed with the setOffset method.

## A.4.4.36 HVersion

### A.4.4.36.1 General

The following text:

- Note that it is a valid implementation to return empty strings for the implementation, vendor and name strings.

Shall be considered to be replaced with:

- Note that it is a valid implementation to return empty strings for HAVI_IMPLEMENTATION_NAME, HAVI_IMPLEMENTATION_VENDOR and HAVI_IMPLEMENTATION_VERSION strings.

### A.4.4.36.2 MHP Specific Clarification

In MHP, a call to `getProperty()` when referencing the constants listed in column Constant in the table below shall return a string as listed in column Value.

**Table 67: getProperty**

| Constant | Value |
|---|---|
| HAVI_SPECIFICATION_VENDOR | "DVB" |
| HAVI_SPECIFICATION_NAME | "MHP" |
| HAVI_SPECIFICATION_VERSION | "<version>" |

<version>

This string shall represent the MHP version to which this HAVi implementation is conformant. The encoding shall be, in order, the major, minor and micro values presented as the textual representation of decimal integers with no leading zeros and separated by a single "." character, e.g. "1.0.3".

#### A.4.4.36.3 Field descriptions

- Consider all occurrences of: java.lang.System.getProperty(havi.specification.version) to be replaced with: java.lang.System.getProperty(HVersion.HAVI_SPECIFICATION_VERSION); and
- all occurrences of: java.lang.System.getProperty(havi.specification.vendor) to be replaced with: java.lang.System.getProperty(HVersion.HAVI_SPECIFICATION_VENDOR); and
- all occurrences of: java.lang.System.getProperty(havi.specification.name to be replaced with: java.lang.System.getProperty(HVersion.HAVI_SPECIFICATION_NAME);
- all occurrences of: java.lang.System.getProperty(havi.implementation.version) to be replaced with: java.lang.System.getProperty(HVersion.HAVI_IMPLEMENTATION_VERSION); and
- all occurrences of: java.lang.System.getProperty(havi.implementation.vendor) to be replaced with: java.lang.System.getProperty(HVersion.HAVI_IMPLEMENTATION_VENDOR); and
- all occurrences of: java.lang.System.getProperty(havi.implementation.name) to be replaced with: java.lang.System.getProperty(HVersion.HAVI_IMPLEMENTATION_NAME).

### A.4.4.37 HKeyboardInputPreferred

#### A.4.4.37.1 INPUT_NUMERIC

The following text:

- This constant indicates that the component only requires alphanumeric input, as determined by the java.lang.Character isDigit method.

shall be considered to be replaced by:

- This constant indicates that the component requires numeric input, as determined by the java.lang.Character isDigit method.

#### A.4.4.37.2 INPUT_ALPHA

The following text:

- This constant indicates that the component only requires alphanumeric input, as determined by the java.lang.Character isLetter method.

Shall be considered to be replaced by:

- This constant indicates that the component requires alphabetic input, as determined by the java.lang.Character isLetter method.

#### A.4.4.37.3 getValidInput

The following text:

- Retrieve the customized input character range. The return value of this method should reflect the range of input keys which the component wishes to see, should getType return a value with the INPUT_CUSTOMIZED bit set. This method may return null if customized input is not requested.

Shall be considered to be replaced with:

- Retrieve the customized input character range. If getType returns a value with the INPUT_CUSTOMIZED bit set then this method shall return an array containing the range of customized input keys. If the range of customized input keys has not been set then this method shall return a zero length char array. This method shall return null if getType returns a value without the INPUT_CUSTOMIZED bit set.

## A.4.4.38 HScreenConfigTemplate

### A.4.4.38.1 Class description

The following text shall be considered to be added to the end of the class description:

- Several preferences in this class are required to be not filled in by the platform in templates generated by the platform. This shall mean:
  - the object for this preference (if there is one) is set to null;
  - the priority for this preference is set to DONT_CARE.

### A.4.4.38.2 VIDEO_GRAPHICS_PIXEL_ALIGNED

In the description of this field, the following text shall be considered to be absent:

- Where a video device is moving the video relative to the screen in real time (e.g. implementing pan and scan), graphics configurations shall only support this feature where the implementation of the graphics device can track the position changes in the video device automatically.

### A.4.4.38.3 DISABLED

Add the following to the class description of HScreenConfigTemplate

```
*
* A special template indicating that an HScreenDevice is disabled.
* Disabled HScreenDevice instances are not being composited as part of
* the video output signal of the HScreen.
* The preferences, properties and attributes of this HScreenConfigTemplate
* are implementation dependent.
*
public static HScreenConfigTemplate DISABLED;
```

### A.4.4.38.4 SCREEN_ASPECT_RATIO

The following additional field shall be considered to be present.

```
/**
 * A value for use in the preference field of the setPreference,
 * getPreferenceObject and getPreferencePriority methods in the
 * HScreenConfigTemplate that indicates that the device configuration
 * supports the screen aspect ratio, as specified in a Dimension object
 * which indicates the (relative) x, y pixel aspect ratio. Typical
 * values are 4:3 and 16:9. <p>
 *
 * Instances of HScreenConfigTemplate generated by the platform and
 * returned to applications (e.g. from getConfigTemplate shall have this
 * preference set to a platform specific value with the REQUIRED
 * priority.
 */
public static final int SCREEN_ASPECT_RATIO=16;
```

# A.4.4.39 HGraphicsConfiguration

## A.4.4.39.1 getConfigTemplate

The following text shall be considered to be added to the description of this method:

- Preferences that are not filled in by the platform will return DONT_CARE priority.

## A.4.4.39.2 Constructor

In the constructor of this class, the following sentence:

- It is not intended that applications should directly construct `HGraphicsConfiguration` objects.

Shall be considered to be replaced with the following:

- An interoperable application shall not subclass the `HGraphicsConfiguration` class.

# A.4.4.40 HVideoConfiguration

## A.4.4.40.1 getConfigTemplate

The following text shall be considered to be added to the description of this method:

- Preferences that are not filled in by the platform will return DONT_CARE priority.

## A.4.4.40.2 Constructor

In the constructor of this class, the following sentence:

- It is not intended that applications should directly construct `HVideoConfiguration` objects.

shall be considered to be replaced with the following:

- An interoperable application shall not subclass the `HVideoConfiguration` class.

# A.4.4.41 HToggleButton

## A.4.4.41.1 Default parameter values exposed in the constructors

The following descriptions will be used to replace the ones already in HToggleButton's "Default parameter values exposed in the constructors" table:

- imageFocused - The image to be used as the content for the HState.FOCUSED_STATE and HState.DISABLED_FOCUSED_STATE states of this component.
- imageActioned - The image to be used as the content for the HState.ACTIONED_FOCUSED_STATE and HState.DISABLED_ACTIONED_FOCUSED_STATE states of this component.
- imageNormalActioned - The image to be used as the content for the HState.ACTIONED_STATE and HState.DISABLED_ACTIONED_STATE states of this component.

# A.4.4.42 HGraphicButton

## A.4.4.42.1 Default parameter values exposed in the constructors

The following descriptions will be used to replace the ones already in HGraphicButton's "Default parameter values exposed in the constructors" table:

- imageFocused - The image to be used as the content for the HState.FOCUSED_STATE and HState.DISABLED_FOCUSED_STATE states of this component.
- imageActioned - The image to be used as the content for the HState.ACTIONED_STATE, HState.ACTIONED_FOCUSED_STATE states of this component.

# A.4.4.43 HTextButton

## A.4.4.43.1 Default parameter values exposed in the constructors

The rows for textFocus and textAction in HTextButton's "Default parameter values exposed in the constructors" table shall be considered to be removed.

# A.4.4.44 HLook and classes implementing HLook

In the getMinimumSize method of HLook and all these classes, the following paragraph shall be considered to be replaced:

- If the HLook supports the scaling of its content (e.g. an HGraphicLook) and content is set then the return value is a size sufficiently large to hold the minimum size of each piece of content plus any additional dimensions that the HLook requires for border decoration etc.

with:

- If the HLook supports the scaling of its content (e.g. an HGraphicLook) and scaling is requested and content is set then the return value is a size containing the width of the narrowest content and the height of the shortest content plus any additional dimensions that the HLook requires for border decoration etc.

# A.4.4.45 HTextLook

## A.4.4.45.1 General

In getMaximumSize(), getMinimumSize() and getPreferredSize(), the following sentence:

- If the HDefaultTextLayoutManager returns a zero size, then proceed with the following steps.

shall be considered to be replaced with the following:

- If HVisible.getTextLayoutManager() does not return an instance of HDefaultTextLayoutManager, or HDefaultTextLayoutManager returns a zero size, then proceed with the following steps as if content for this HVisible had not been set.

## A.4.4.46 HExtendedLook

# org.havi.ui

# HExtendedLook

### Syntax

```
public interface HExtendedLook extends HLook
```

All Superinterfaces:

java.lang.Cloneable, `HLook`

### Description

The HExtendedLook interface is derived from the `HLook` interface and abstracts out the drawing of the look into separate background, border and visible data components. The interface allows the programmer to derive new looks from the default looks and have control over which component parts are drawn. This aids interoperability between platforms since no two manufacturer's `HLook` implementation look the same.

See Also:

setLook(HLook), setLookData(Object, Object), paint(Graphics), setBackgroundMode(int), setHorizontalAlignment(int), setVerticalAlignment(int), setResizeMode(int), HTextLook, HGraphicLook, HAnimateLook, HRangeLook, HSinglelineEntryLook, HMultilineEntryLook

Methods

### fillBackground(Graphics, HVisible, int)

```
public void fillBackground(java.awt.Graphics g, HVisible visible, int state)
```

The fillBackground(Graphics, HVisible, int) method paints the component with its current background `Color` according to the `setBackgroundMode(int)` method of `HVisible`.

This method is only called from `showLook` within this `HExtendedLook` implementation. It's not the intention to call this method directly from outside of the `HExtendedLook`.

Regardless of the background mode, it is an implementation option for this method to render added decorations which may affect the look's transparency. This method shall not be used to render any HVisible related data or content associated with the HVisible. It is purely for background and decoration only.

The `fillBackground` method should not modify the clipRect (clipping rectangle) of the `Graphics` object that is passed to it in a way which includes any area not part of that original clipRect. If any modifications are made, the original clipRect shall be restored.

Parameters:

`g` - the graphics context.

`visible` - the visible.

`state` - the state parameter indicates the state of the visible.

See Also:

`isOpaque(HVisible)`

### renderBorders(Graphics, HVisible, int)

```
public void renderBorders(java.awt.Graphics g, HVisible visible, int state)
```

The renderBorders(Graphics, HVisible, int) method paints any implementation specific borders according to the setBordersEnabled(boolean) method of HVisible . If borders are drawn, the border decoration shall be rendered within the border area as returned by getInsets. The behavior of this method, when a subclass overrides the method getInsets is undefined, except that it will never draw outside the border area as returned by getInsets.

This method is only called from showLook within this HExtendedLook implementation. It's not the intention to call this method directly from outside of the HExtendedLook.

The renderBorders(Graphics, HVisible, int) method should not modify the clipRect (clipping rectangle) of the Graphics object that is passed to it in a way which includes any area not part of that original clipRect. If any modifications are made, the original clipRect shall be restored.

Parameters:

g - the graphics context.

visible - the visible.

state - the state parameter indicates the state of the visible.

### renderVisible(Graphics, HVisible, int)

```
public void renderVisible(java.awt.Graphics g, HVisible visible, int state)
```

The renderVisible(Graphics, HVisible, int) method paints any content or other data associated with the look's HVisible. This method shall not be used to render a background nor any other decoration. Its purpose is purely to render content or other value data stored on the HVisible. This may be set via HVisible methods such as setTextContent and setGraphicContent. Rendering shall take place within the bounds returned by the getInsets method.

This method is only called from showLook within this HExtendedLook implementation. It's not the intention to call this method directly from outside of the HExtendedLook.

For looks which draw content (e.g. HTextLook , HGraphicLook and HAnimateLook ), if no content is associated with the component, this method does nothing.

The renderVisible(Graphics, HVisible, int) method should not modify the clipRect (clipping rectangle) of the Graphics object that is passed to it in a way which includes any area not part of that original clipRect. If any modifications are made, the original clipRect shall be restored.

Parameters:

g - the graphics context.

visible - the visible.

state - the state parameter indicates the state of the visible.

### showLook(Graphics, HVisible, int)

```
public void showLook(java.awt.Graphics g, HVisible visible, int state)
```

The showLook(Graphics, HVisible, int) method is the entry point for repainting the entire HVisible component. This method delegates the responsibility of drawing the component background, borders and any HVisible related data or content to the fillBackground, renderVisible and renderBorders methods respectively, subject to the clipping rectangle of the Graphics object passed to it. This method shall call the methods fillBackground, renderVisible, and renderBorders in that order and shall not do any rendering itself.

The showLook(Graphics, HVisible, int) method should not modify the clipRect (clipping rectangle) of the Graphics object that is passed to it in a way which includes any area not part of that original clipRect. If any modifications are made, the original clipRect shall be restored.

Any resources explicitly associated with an HExtendedLook should be loaded by the HExtendedLook during its creation, etc. Note that the "standard" looks don't load content by default.

This method is called from the paint(Graphics) method of HVisible and must never be called from elsewhere. Components wishing to redraw themselves should call their repaint method in the usual way.

Overrides:

showLook(Graphics, HVisible, int) in interface HExtendedLook

Parameters:

`g` - the graphics context.

`visible` - the visible.

`state` - the state parameter indicates the state of the visible, allowing the look to render the appropriate content for that state. Note that some components (e.g. HStaticRange, HRange, HRangeValue) do not use state-based content.

## A.4.4.47 HAnimateLook, HGraphicLook, HListGroupLook, HMultilineEntryLook, HRangeLook, HSinglelineEntryLook and HTextLook

The following changes shall be considered to apply to all these implementing looks.

a) The method descriptions in HExtendedLook for fillBackground(), renderBorders() and renderVisible() shall be considered to form part of all the above looks.

b) The description of showLook() in each of the above looks shall be considered to be replaced with the one in HExtendedLook.

c) HAnimateLook, HGraphicLook, HSinglelineEntryLook and HTextLook shall be considered to implement HExtendedLook instead of HLook.

d) HListGroupLook and HRangeLook shall be considered to implement HExtendedLook additionally to HAdjustableLook. The method showLook() shall be implemented as specified by HExtendedLook.

## A.4.4.48 HSwitchable

The following sentence:

- Actioned states (i.e. those with the ACTIONED_STATE_BIT bit set may persist after any registered HActionListener listeners have been called, until a further HActionEvent is received.

Shall be considered to be modified as follows:

- Actioned states (i.e. those with the ACTIONED_STATE_BIT bit set) may persist after any registered HActionListener listeners have been called, until a further HActionEvent is received.

## A.4.4.49 HStillImageBackgroundConfiguration

### A.4.4.49.1 Constructor

In the constructor of this class, the following sentence:

- It is not intended that applications should directly construct `HStillImageBackgroundConfiguration` objects.

Shall be considered to be replaced with the following:

- An interoperable application shall not subclass the `HStillImageBackgroundConfiguration` class.

## A.4.4.50 HTextValue

### A.4.4.50.1 Class description

In the description of this interface, the following text:

- HAVi text events are discussed in detail in the HKeyboardInputPreferred interface description.

shall be considered to be replaced by:

- HAVi text events are discussed in detail in the HTextEvent description.

## A.4.4.51 HDefaultTextLayoutManager

### A.4.4.51.1 getMinimumSize

The following parameters and returns clause shall be considered to be present:

Parameters:

- hvisible - a component within which text will be rendered.

Returns:

- the minimum size of text rendered in that component.

### A.4.4.51.2 getMaximumSize

The following parameters and returns clause shall be considered to be present.

Parameters:

- hvisible - a component within which text will be rendered.

Returns:

- the maximum size of text rendered in that component.

### A.4.4.51.3 getPreferredSize

The following parameters and returns clause shall be considered to be present.

Parameters:

- hvisible - a component within which text will be rendered.

Returns:

- the preferred size for text rendered in that component.

## A.4.4.52 Hscreen

### A.4.4.52.1 getHGraphicsDevices, getHVideoDevices, getHBackgroundDevices

Add the following to the method body of each of the following methods - HScreen.getGraphicsDevices, HScreen.getVideoDevices, HScreen.getBackgroundDevices:

```
*
* The devices returned shall be in z-order with each device being in
* front of those devices with a higher index in the array of results.
*
```

### A.4.4.52.2 getCoherentScreenConfigurations

Add the following to the method description of HScreen.getCoherentScreenConfigurations(..)

```
*
* Both the input to this method and the output from this method shall be
* sorted in z-order (from front to back) within each type of HscreenConfiguration.
* The ordering of different sub-classes of HScreenConfigTemplate shall be ignored.
* Where more than one template of the same type is
* provided, the returned configurations shall be in the same z-order as
* the templates in the input. e.g. if the template at index 0 in the input
* requests a resolution of 960x540 and the template at index 1 in the
* input requests a resolution of 720x576 then the configuration at index 0
* in the output shall have a resolution of 960x540 and the configuration
* at index 1 in the output shall have a resolution of 720x576. If the
* z-order requirement cannot be met, this function shall return null.
* The special HScreenConfigTemplate DISABLED should not normally be requested
* in this method since more capable MHP terminals may be able to support
* more than what is requested.
*
```

### A.4.4.52.3 setCoherentScreenConfigurations

In the method, HScreen.setCoherentScreenConfigurations():

Add the following to the class description:

- The HScreenConfigurations shall be in z-order as defined for HScreen.getCoherentScreenConfigurations.

Change the @throws clause for HConfigurationException to read as follows:

```
*
* @throws HConfigurationException - if the specified
* HScreenConfiguration[] array is not valid for any of the devices
* or if the z-order of the configurations in the array does not match
* the z-order of the devices.
*
```

### A.4.4.52.4 Additional methods

The following additional methods shall be considered to be present:

```
/**
 * Return the aspect ratio of the video output signal represented
 * by this HScreen.
 * @return a Dimension object specifying the aspect ratio of the video
 * output signal
 */
public Dimension getScreenAspectRatio()

/**
 * Return the aspect ratio of the the physical display that is
 * connected to this HScreen as far as that is known. The extent to
 * which this is known depends on the connector and protocol used
 * to connect the display. Some connectors may support a protocol
 * signalling the display aspect ratio. Otherwise the display aspect
 * ratio may be a user setting entered as part of the configuration
 * process of the MHP terminal.
 *
 * @return a Dimension object specifying the aspect ratio of the display
 * @see VideoFormatControl#getDisplayAspectRatio
 */
public Dimension getDisplayAspectRatio()
```

# A.4.5 org.havi.ui.event

## A.4.5.1 HActionEvent

### A.4.5.1.1 getModifiers

The text:

- Modifiers are not used with the HAVi platform. Interoperable HAVi applications shall not use the return value of this method.

Shall be considered to be replaced by:

- Modifiers are not used for `HActionEvents` with the HAVi platform. Interoperable HAVi applications shall not use the return value of this method.

## A.4.5.2 HItemEvent

### A.4.5.2.1 Constructor

The following text:

- item - The item which caused the change, or null if this information is not available. This information shall be provided if the event id is one of ITEM_SELECTED, ITEM_CLEARED, ITEM_SET_CURRENT, ITEM_SET_NEXT or ITEM_SET_PREVIOUS.

Shall be considered to be replaced with the following:

- item - The item which caused the change, or null if this information is not available.

  If the event is sent to listeners, this information shall be provided if the event id is one of ITEM_SELECTED, ITEM_CLEARED, ITEM_SET_NEXT or ITEM_SET_PREVIOUS".

### A.4.5.2.2 ITEM_SET_CURRENT

The following paragraph:

- An item event with this id is sent from the component whenever the current item of an HItemValue component changes.

Shall be considered to be replaced with:

- An item event with this id is sent to or from the component whenever the current item of an HItemValue component changes.

## A.4.5.3 HKeyEvent

The following paragraph shall not apply in the present document:

- Note that the HAVi system should only generate KEY_PRESSED events. Neither KEY_TYPED nor KEY_RELEASED events should be generated. Furthermore, the system should collapse combined events. For example, a usual Java Virtual Machine generates for the letter A three events: KEY_PRESSED for modifier key Shift, KEY_PRESSED for letter "A" and KEY_TYPED for "A". This should be collapsed into one single KEY_PRESSED event with the letter "A" and the Shift modifier set. This is to simplify the key event handling of applications.

#### A.4.5.3.1 Constructors

The description of the "id" parameter shall be considered to have the following text added:

- This is the value that will be returned by the event object's getID method.

### A.4.5.4 HRcEvent

The following changes shall be considered to apply in this class:

- Replace all occurrences of "action id" in the descriptions of the VK constants by "key code".
- Replace all occurrences of "event ids" in the descriptions of RC_FIRST and RC_LAST by "key codes".Tag both constants as being deprecated. Add for RC_FIRST a specific deprecated text:

  * @deprecated The value of this field is useless, since it mixes event ids and key codes. It does not reflect any of the remote control key codes listed in this class.
- Remove the following sentence from the class description: "Note that it is an implementation constraint that the HrcEvent event range should not intersect with the Java AWT key event range".
- Add to the parameter descriptions of "id" for both constructors: "in the range KEY_FIRST to KEY_LAST".
- In the definition of the field, VK_COLORED_KEY_0, the paragraph starting with "Up to six colored soft keys can be included on a remote control." shall be considered to be extended with the following; "In markets where text is written from right to left, these keys may be oriented from right to left in ascending order".

### A.4.5.5 HRcCapabilities

#### A.4.5.5.1 getRepresentation

In the method `getRepresentation(int aCode)`, all references to the parameter `id` shall be considered to be replaced with `keyCode`.

#### A.4.5.5.2 Constructor

The following sentence:

- It is not intended that applications should directly construct `HRcCapabilities` objects.

Shall be considered to be replaced by:

- An interoperable application shall not subclass the `HRcCapabilities` class.

### A.4.5.6 HEventGroup

#### A.4.5.6.1 getKeyEvents

This class shall be considered to be extended with the following method:

```
/**
 * Return the key codes contained in this event group
 */
public int[] getKeyEvents() {}
```

### A.4.5.7 HKeyCapabilities

#### A.4.5.7.1 Constructor

The following sentence:

- It is not intended that applications should directly construct `HKeyCapabilities` objects.

Shall be considered to be replaced by:

- An interoperable application shall not subclass the `HKeyCapabilities` class.

### A.4.5.8 HEventRepresentation

#### A.4.5.8.1 Constructor

The following sentence:

- It is not intended that applications should directly construct `HEventRepresentation` objects.

Shall be considered to be replaced by:

- An interoperable application shall not subclass the `HEventRepresentation` class.

## A.4.6 References to ISO/IEC10646-1:1993

All references to ISO/IEC10646-1:1993 are considered to be non-specific references to ISO/IEC 10646-1 [15]. Also, the term "support for Unicode ranges" is considered to be "support for character ranges as specified by ISO/IEC 10646-1 [15]".

## A.4.7 Chapter 8

### A.4.7.1 Section 8.2.3.3.2 "Compatibility with Existing java.awt Methods"

In this section, the following sentence:

- The `java.awt.Toolkit.getScreenSize` method shall be equivalent to the pixel resolution of the current configuration of the default screen device returned by HScreen.getDefaultGraphicsDevice.

Shall be considered to read:

- The `java.awt.Toolkit.getScreenSize` method shall be equivalent to the pixel resolution of the current configuration of the default screen device returned by HScreen.getDefaultHGraphicsDevice.

# A.5 ISO/IEC 13818-6

With reference to ISO/IEC 13818-6 [38].

## A.5.1 Reconstruction of NPT

In equation 8-3, SCR_Reference is considered to be STC_Reference.

# Annex B (normative): Broadcast filesystem and trigger transport

## B.0 General

The present document does not specify a transport protocol for broadcast file systems or for trigger (event) delivery. It does, however, require that GEM terminal specifications provide a mechanism for delivery of filesystems and triggers.

NOTE: MHP [1], annex B defines a profile of DSMCC object carousels that fulfills the requirements of this annex.

## B.1 Service domain

Terminal specifications based on GEM shall include a mechanism for signalling a service domain. A service domain is an entity that uniquely identifies a filesystem, which can contain files, directories, stream descriptions, trigger objects and trigger events. The format of these is described in the following clauses. This mechanism shall be sufficient to support all functionalities of the API defined in annex P.

Terminal specifications based on GEM shall define a syntax for a locator to refer to a service domain. This locator syntax shall support the encoding of an optional integer, in order to accommodate the requirements of the method ServiceDomain.getLocator() (see clause P.2.5.3).

The signalling for a service domain shall be sufficient to identify the "root" directory of a filesystem, and allow attaching to that filesystem.

The details of mounting a service domain are described in annex P.

## B.2 Filesystem requirements

### B.2.1 Static requirements

Terminal specifications based on GEM shall include a mechanism for delivering a hierarchical file system within a service domain. It shall be possible to construct a locator that refers to files and directories in this hierarchy. The file system delivery mechanism shall satisfy the following minimum requirements. Of course, in addition to these limits, available bandwidth and memory resources would constrain the size of what can practically be broadcast.

**Table 68: Filesystem signalling requirements**

| Area | Minimum requirement |
|---|---|
| Characters Allowed in File and Directory Names | The ASCII character "a.."z", "A.."Z", "0.."9", "$", "-", and "_". After the first character of a file name, "." and " " (the space character) is also permitted |
| Maximum length of file name | 200 characters |
| Number of entries per directory | 500 |
| Maximum Directory Nesting | 20 levels |
| Maximum File Length | $2^{27}$ - 1 bytes |
| Caching | See clause B.2.1.1, "Caching behaviour" |

Filesystems may be either case-sensitive or "case preserving." GEM applications shall be written to work with both of these. "Case preserving" is defined in clause 14.6.1, "Persistent storage". GEM terminal specifications may require case-sensitive filesystems.

### B.2.1.1 Caching behaviour

It shall be possible to signal along with carousel modules, information related to the caching behaviour of a GEM terminal.

This signalling shall contain sufficient information to derive the following.

**Table 69: Caching behaviour signalling requirements**

| Function | Type |
|---|---|
| priority_value | 8 bit unsigned integer |
| transparency_level | 8 bit unsigned integer |

priority_value: indicates the caching priority for the objects within this module. A higher value indicates more importance for caching.

transparency_level: Transparency level that shall be used by the GEM terminal if it caches objects contained in this module. The possible values are listed in table 70. The semantics of the policies are defined in clause B.5.2, "Transparency levels of caching".

**Table 70: Transparency level values**

| Value | Description |
|---|---|
| 0 | Reserved |
| 1 | Transparent caching |
| 2 | Semi-transparent caching |
| 3 | Static caching. |
| 4 to 255 | Reserved for future use |

When not otherwise specified, the default values that shall be assumed are:

- priority_value: 128.
- transparency_level: 1 (transparent caching).

This clause describes the default semantics of these two fields. Additional semantics are defined in clause B.5. GEM terminals that support caching shall comply with these semantics.

NOTE: For the packaged media target, the contents of the filesystem usually does not change. For this reason, explicit signalling of the transparency level is not required.

## B.2.2 Filesystem updates

For broadcast targets, it shall be possible to signal a new version of a file or directory.

## B.2.3 Content type descriptor

Zero or one content type descriptors can be present in the filesystem. Where more than one content type descriptor is used they shall express the same content format. Also, the content type (if any) signalled in the directory binding shall be identical to that signalled in the bound file's header. This optional descriptor identifies the media type of the file.

This content type signalling only applies to objects of type file and is not appropriate for other object types.

If this descriptor is absent or not sufficient to categorize the content type then the extension portion of the file name shall be used be used to provide the media type mapping via table7, "File type identification". The extension portion of the filename shall not be used if this descriptor is provided.

# B.3 Stream description

There shall be a signalling mechanism for sending a description of an MPEG stream. Such a stream can carry a service (e.g. a DVB service).

NOTE 1: The following requirements are modelled on DSMCC BIOP:StreamMessage.

Stream descriptions shall be identified with a special file sent in the hierarchical filesystem described in clause B.2. This file shall contain information sufficient to derive the following:

**Table 71: Stream description**

| Function | Type |
|---|---|
| npt_source | Reference (see text) |
| stream_locator | Locator external form |
| duration | 32 bit unsigned integer |
| audio_stream | flag |
| video_stream | flag |
| data_stream | flag |
| is_mpeg_program | flag |

npt_source: A reference to a source of timeline information. This shall be sufficient to derive timeline values, and the timeline rate. It may indicate that no timeline is associated with this stream collection.

stream_locator: A locator that references the streams of this collection.

duration: The duration of this stream description, in milliseconds. Signalling for this value is not required; if not present it shall always considered to be zero.

NOTE 2: GEM signalling can indicate a value of up to $2^{32}$ seconds with a resolution of microseconds.

audio_stream: True if this stream contains audio.

video_stream: True if this stream contains video.

data_stream: True if this stream contains data.

is_mpeg_program: An indication whether or not this stream collection is an MPEG program.

# B.4 Trigger signalling

## B.4.0 General

There shall be a mechanism for signalling the presence of triggers to an application, and for delivery of those triggers.

## B.4.1 Trigger object

Triggers shall be identified with a special file sent in the hierarchical filesystem described in clause B.2. This file shall contain information sufficient to derive all of the contents of a Stream description, plus the following:

**Table 72: Trigger object**

| Function | Type |
|---|---|
| num_triggers | 16 bit unsigned integer |
| for (i=0; i<N; i++) { | |
| trigger_name | string |
| event_id | 14 bit unsigned integer |
| } | |

num_triggers: The number of trigger events identified in this trigger object

trigger_name: The name of a trigger event. The signalling shall support trigger names up to 200 characters long containing any valid 7-bit ASCII character between 32 and 126, inclusive.

event_id: An integer uniquely identifying a trigger event within the context of the currently selected service.

NOTE: A trigger object corresponds to a BIOP::StreamEvent message in DSMCC.

## B.4.2 Trigger event

It shall be possible to signal a trigger event. For broadcast targets, the signalling shall contain information sufficient to derive the following. For the packaged media profile, it shall be possible to specify trigger events with a payload as described; the distinction between "do it now" events and time-based events is not meaningful and need not be encoded.

NOTE 1: In MHP, applications cannot detect the signalling underlying a trigger event.

**Table 73: Trigger event**

| Function | Type |
|---|---|
| event_name | string |
| is_do_it_now | flag |
| timebase_value | 32 bit unsigned integer |
| payload | byte array |

event_name: The name of the trigger event. The signalling shall support trigger names up to 32 characters long containing the ASCII characters "a".."z", "A".."Z", "0".."9" and "-".

NOTE 2: This may be signalled as an event_id that maps to a trigger identification within a trigger object, as defined in clause B.4.1, "Trigger object".

is_do_it_now: Flag indicating if this is a "do it now" event. If true, this is a "do it now" event that is to be triggered upon reception. If false, this is a scheduled event to be triggered when a given NPT value is reached.

timebase_value: A value that identifies a point in time in a timebase signalled with a service. This is implemented as an MPEG NPT value in the DSMCC signalling.. For "do it now" events, this value is ignored.

NOTE 3: A timebase associated with the stream identified by the Trigger object will be used by the terminal to send a trigger to a registered application.

payload: A sequence of up to 220 bytes containing arbitrary data.

NOTE 4: A trigger event corresponds to a DSMCC section carrying a stream event descriptor.

### B.4.2.1 Extrapolation of timebase values

GEM terminal specifications for broadcast targets shall be written such that, for broadcasts confirming to appropriate broadcast norms and specifications and absent reception errors, any extrapolation of timebase values shall last no more than 5 seconds.

NOTE: This corresponds to the requirements in NPT signalling spelled out in MHP [1], clause B.2.4.4, "Timebases".

GEM terminal specifications for packaged media targets shall require detecting triggers associated with the currently played media stream.

### B.4.2.2 Monitoring of trigger events

For each GEM application, GEM terminal specifications for broadcast targets shall require monitoring at least one stream delivering scheduled stream events, and one stream delivering "do it now" stream events. For broadcasts confirming to appropriate broadcast norms and specifications and absent reception errors, the terminal shall raise an event in response to a scheduled trigger event provided that an application subscribed to the event at least 5 seconds before the scheduled time.

NOTE: This corresponds to the requirements spelled out in MHP [1], clause B.2.4.5, "Monitoring Stream Events." MHP requires that scheduled stream event descriptors be *broadcast* at least 5 s before the scheduled time, but in GEM this requirement is not expressed, because it is a part of appropriate broadcast norms and specifications.

## B.5 Caching

This clause describes the constraints that a GEM terminal shall implement when caching any content from the object carousel in the memory of the GEM terminal. Caching is optional for the GEM terminal, but if implemented shall conform to the constraints set in this clause.

### B.5.1 Determining file version

There is no version number directly related to files (or other BIOP messages), the closest association is the moduleVersion in the DII that references the module that contains the BIOP message. Therefore, to ensure that a file is up to date the GEM terminal must determine that the moduleVersion for the appropriate module is current and reacquire if necessary. The circumstances under which this checking is required are defined by the transparency level as specified in the following clause.

### B.5.2 Transparency levels of caching

The definition of transparency levels describes the behaviour that the GEM terminal shall implement when the content in the object carousel is changing. The transparency level determines how certain the GEM terminal is required to be about the validity of the content when returning the content to the application. The object carousel provides a mechanism for determining version changes of the content by monitoring the DII messages.

Validity of content is specified here in terms of the version number of the module that is broadcast in the DII message. The contents of an object as cached in the memory of the GEM terminal are defined to be valid at a certain point in time when the version number of the module in the cache matches the version number of the module as signalled in the DII message describing that module as it was last broadcast.

NOTE: The definition is based on the DII message that was last broadcast and it may be that the GEM terminal was not filtering for this message at that time and did not receive it.

From the GEM terminal point of view, the transparency level indicates the constraints that the terminal needs to implement for monitoring the DII messages.

The broadcaster can indicate the appropriate transparency level that shall be applied for a given piece of content by using a descriptor associated with a module in the DII message. In the absence of this descriptor from a module, the transparent caching is the default level.

#### B.5.2.1 Transparent caching

The transparent caching is a caching level that ensures that the application can not practically notice a difference in the validity of the returned content between an implementation that caches content and an implementation that does not cache any content. Naturally, an implementation that caches the content will return it to the application faster.

When returning content from the cache to the application, the GEM terminal shall ensure that the version number of the cached content matches the version number indicated in the current DII message describing that module. Once a DII has been received it can be assumed that it is current at least for 500 ms and after that period until receiving the next instance of the relevant DII. If filtering for that DII has not resumed by the end of this period, the state of that DII is to be considered unknown until it is received again.

Therefore, terminals must not return transparently cached data if it has waited more than half a second between receiving the relevant DII and *starting to filter* for that DII again. If the terminal does not resume filtering within the 500 ms grace period, it must download the relevant DII again when it wishes to use that DII to check cache validity.

The choice of 500 ms is based on the normal timing uncertainty in data delivery through the broadcast chain and is independent of the repetition rate of the DII messages.

#### B.5.2.1.1 Active caching

There are several ways the GEM terminal can organize its caching strategy. One possible strategy is so-called active caching. This means that the terminal has a dedicated section filter for each DII message it needs to monitor. Keeping that filter continuously filtering for the DII guarantees that the terminal will notice the update of a module as soon as it happens and can thus be aware of the validity of all the content it has cached.

However, in some cases the DII messages might be sent with a very high repetition rate that may cause a high processing load because the terminal needs to do some processing every DII message that it receives. The 500 ms grace period is designed to help this, as it allows the terminal to stop the section filter for 500 ms after receiving the DII message. This lessens the processing burden on the terminal as it only needs to process each DII message twice a second, even if it may be repeated on the transmission much more frequently.

#### B.5.2.1.2 Passive caching

With active caching, the terminal may need to have a dedicated section filter reserved for each DII message that it needs to monitor. This would effectively limit the amount of content that can be cached, possibly to a very small number. Therefore, the terminal may choose a so-called passive caching strategy. This means that the terminal does not even try to monitor for the DII messages continuously, but each time an application wants to retrieve an object, it at that time retrieves the current DII and checks if the cached content is still valid. Although, this strategy imposes a delay before returning the content to the application, this delay is usually significantly smaller than having to retrieve the content from the broadcast stream.

#### B.5.2.1.3 DII repetition rate

It should be noted that the description of active and passive caching are only informative here and terminal implementations can use any strategy fulfilling the normative constraints set above. However, broadcasters should set the repetition rate of the DIIs so that a terminal implementing the passive caching strategy will provide the expected benefits of caching over a terminal implementing no caching.

### B.5.2.2 Semi-transparent caching

The semi-transparent caching level allows the GEM terminal to cache the data and also return slightly out-dated data to the application. The benefit of this caching level is that it allows terminals to cache larger quantities of content with a reasonable resource usage while allowing the data to be returned usually immediately to the application. The semi-transparent caching level provides less guarantees about validity of the content, but does not cause the delay implied by the passive caching strategy with the transparent caching level.

When returning content from the cache to the application, the terminal shall ensure that the version number of the cached content matches the version number indicated in a valid DII message describing that module. Once a DII has been received it can be assumed to be valid at least for 30 s and after that period until receiving the next instance of the relevant DII. If filtering for that DII has not resumed by the end of this period, the state of that DII is to be considered unknown until it is received again.

Therefore, terminals must not return semi-transparently cached data if it has waited more than 30 seconds between receiving the relevant DII and starting to filter for that DII again. If the terminal does not resume filtering within the 30 s grace period, it must download the relevant DII again when it wishes to use that DII to check cache validity.

#### B.5.2.2.1 Implications for the terminal (informative)

Reasons for selecting the 30 s value for the grace period in the semi-transparent caching level are different from the reasons for the 500 ms grace period in the transparent level. The 30 s grace period in this level is intended e.g. to allow terminals to keep typically a valid copy of each DII by retrieving each DII in a round robin fashion using a single section filter. Naturally, whether this goal can be achieved, depends on the repetition rate of the DIIs and the amount of content that is cached. If this is not possible, the terminal might use the passive caching strategy with this transparency level as well. These strategies are only examples and the terminal may implement any strategy as long the normative constraints defined above are fulfilled (this includes implementing no caching as it is optional, as well as treating the semi-transparent level the same as the transparent level).

### B.5.2.3 Static caching

When using the static caching transparency level, the GEM terminal shall check the validity of the cached content from the version number in the DII message when it is used for the first time during the lifetime of an application instance. After the first usage time, the GEM terminal does not need to check the validity of the content during the lifetime of that application instance.

#### B.5.2.3.1 Implications for the broadcaster (informative)

This has the implication, that content with this transparency level is appropriate for very static content that is updated only rarely and where the possible update of the content does not need to be noticed by the application during the lifetime of one application instance.

#### B.5.2.3.2 Implications for the terminal (informative)

The GEM terminal, however, is allowed to update the contents of the statically cached files if it notices that they have been updated in the carousel as well as use any caching strategy as long as the normative constraint defined above are fulfilled (this includes implementing no caching as it is optional, as well as treating the static level the same as the semi-transparent and/or the transparent level).

## B.5.3 Dynamic carousel structure

The Object Carousel may change structure over time, i.e. both files and directories may be added or deleted. Also, modules are not guaranteed to carry the same objects over the lifetime of the carousel. Therefore receivers shall not assume that directory structures are static or that a given path will resolve always to the same object. All cached directory information shall be cached according to the signalled cache priority. This means that before using an object that has been cached, receivers shall validate the path to it.

NOTE: Validating a path does not necessarily mean downloading all elements in the path every time. For example, simply determining that none of the objects on the path have changed since it was last fully traversed is sufficient to confirm that the path itself has not changed.

# Annex C (informative): Bibliography

## C.1 Television

UK MHEG Profile, 1.05: "Digital Terrestrial Television MHEG-5 Specification", U.K. DTG.

Porter-Duff, , T. Porter and T. Duff, "Compositing Digital Images", SIGGRAPH 84, 253-259.

E-Book, 1.1, EACEM Technical Report Number TR-030, Baseline Digital Terrestrial TV Receiver Specification.

ETSI ES 202 184 (V1.1.1): "MHEG-5 Broadcast Profile".

ITU-R Recommendation BT.601-5: "Studio Encoding Parameters of Digital Television for standard 4:3 and Wide-Screen 16:9 aspect ratios".

PKCS11, 5, PKCS11 Cryptographic Token Interface Standard.

NOTE: Available at http://www.rsasecurity.com/rsalabs/node.asp?id=2133.

CENELEC EN 50049-1: "Domestic and similar electronic equipment interconnection requirements: Peritelevision connector".

Hunt, R.W.G. (1987): "Measuring Colour (Ellis Horwood Series in Applied Science and Industrial Technology)". Ellis Harwood Limited, Chichester, England.

POSIX, ISBN:1-55937-255-9, IEEE Standard 1003.2-1992: "Standard for Information Technology - Portable Operating System Interfaces (POSIX ) - Part 2: Shell and Utilities".

CENELEC EN 50221: "Common interface specification for conditional access and other digital video broadcasting decoder applications".

ETSI TS 101 699 (V1.1.1): "Digital Video Broadcasting (DVB); Extensions to the Common Interface Specification".

R206-001: "Guidelines for Implementation and Use of the Common Interface for DVB Decoder Applications".

ANSI/SCTE 90-1 (2004): "SCTE Applications Platform Part 1 OCAP 1.0 Profile".

IETF RFC 1112 (1989): "Host extensions for IP multicasting".

## C.2 Java

Java Media Player guide, 1.03, Nov 6, 1997, Sun Microsystems Java Media Player guide, Java Media Players.

NOTE: Available at http://java.sun.com/products/java-media/jmf/forDevelopers/playerguide/index.html.

Java VM2, ISBN: 0-201-43294-3, The Java Virtual Machine Specification (2nd edition), T. Lindholm and F. Yellin, Addison-Wesley.

Java Class Libraries Vol. 1, ISBN 0-2011002-3, The Java Class Libraries, Second Edition, Volume 1 by Patrick Chan, Rosanna Lee and Douglas Kramer.

Java Class Libraries Vol. 2, ISBN 0-201-31003-1, The Java Class Libraries, Second Edition, Volume 2 by Patrick Chan and Rosanna Lee.

JSR 927, 1.1, Java TV 1.1.

J2SE 5.0, 5, JavaTM 2 Platform Standard Edition 5.0 Development Kit (JDK 5.0).

Java SE, Java SE at a Glance.

NOTE: Available at http://java.sun.com/javase/.

Compilers: Principles, Techniques, and Tools by Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman (Contributor); Addison-Wesley Pub Co, ISBN: 0201100886.

## C.3 Internet and HTTP

IETF RFC 2068 (January 1997): "Hypertext Transfer Protocol version 1.1".

IETF RFC 2838 (May 2000): "Uniform Resource Identifiers for Television Broadcasts".

W3C Recommendation 5 May 1999: "Web Content Accessibility Guidelines 1.0".

NOTE: Available at http://www.w3.org/TR/WAI-WEBCONTENT.

W3C Recommendation 26 January 2000: "XHTML™ 1.0 The Extensible HyperText Markup Language".

NOTE: Available at http://www.w3.org/TR/xhtml1.

W3C Recommendation 8 October 2008: "XHTML™ Modularization 1.1.

NOTE: Available at http://www.w3.org/TR/xhtml-modularization.

W3C Recommendation 17 Dec 1996: "Cascading Style Sheets, level 1".

NOTE: Available at http://www.w3.org/TR/REC-CSS1/.

CSS 2 (1998): "Cascading Style Sheets, level 2 CSS2 Specification", Including the errata published in REC-CSS2-19980512.

NOTE: Available at http://www.w3.org/TR/REC-CSS2/.

IETF RFC 2318 (1998): "The text/css Media Type".

W3C Recommendation 29 June 1999: "Associating Style Sheets with XML documents, 1.0".

NOTE: Available at http://www.w3.org/TR/xml-stylesheet.

IETF RFC 2109 (1997): "HTTP State Management Mechanism".

IETF RFC 1123 (1989): "Requirements for Internet Hosts Application and Support".

## C.4 Other

IETF RFC 1950 (1996): "ZLIB Compressed Data Format Specification version 3.3".

IETF RFC 1951 (1996): "DEFLATE Compressed Data Format Specification version 1.3".

SMIL 1.0.

NOTE: Available at http://www.w3.org/TR/REC-smil/.

SMIL Boston, "Synchronized Multimedia Integration Language (SMIL)". Boston Specification.

NOTE: Available at http://www.w3.org/TR/smil-boston.

SMIL DOM: "Synchronized Multimedia Integration Language Document". Object Model.

NOTE: Available at http://www.w3.org/TR/smil-boston-dom.

W3C Recommendation 1 October 1998: "Document Object Model (DOM) Level 1 Specification, V1.0".

NOTE: Available at http://www.w3.org/TR/REC-DOM-Level-1.

W3C Recommendation 13 November 2000: "Document Object Model (DOM) Level 2 Core Specification, V1.0".

NOTE: Available at http://www.w3.org/TR/DOM-Level-2-Core.

W3C Recommendation 13 November 2000: "Document Object Model (DOM) Level 2 Events Specification, V1.0".

NOTE: Available at http://www.w3.org/TR/DOM-Level-2-Events.

W3C Recommendation 13 November 2000: "Document Object Model (DOM) Level 2 Style Specification, V1.0".

NOTE: Available at http://www.w3.org/TR/DOM-Level-2-Style.

W3C Recommendation 13 November 2000: "Document Object Model (DOM) Level 2 Views Specification, V1.0".

NOTE: Available at http://www.w3.org/TR/DOM-Level-2-Views.

W3C Recommendation 16 August 2006: "Namespaces in XML 1.0", World Wide Web Consortium 14-January-1999.

NOTE: Available at http://www.w3.org/TR/REC-xml-names.

W3C Candidate Recommendation 2 August 2000: "Scalable Vector Graphics (SVG) 1.0 Specification".

NOTE: Available at http://www.w3.org/TR/2000/CR-SVG-20000802/.

ETSI TS 102 823: "Digital Video Broadcasting (DVB); Specification for the carriage of synchronized auxiliary data in DVB transport streams".

ETSI TR 101 194 (V1.1.1): "Digital Video Broadcasting (DVB); Guidelines for implementation and usage of the specification of network independent protocols for DVB interactive services".

ITU-T Recommendation X.690: "Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)".

# Annex D (normative): Text presentation

## D.1 Scope

This clause addresses the following topics:

- How downloaded fonts are associated with applications and accessed by them.
- The DVB-J APIs that are used for presenting text and their behaviour.

Two levels of interface are addressed:

- Simple string rendering as supported by `java.awt.Graphics.drawString`.
- More complex object rendering as supported by DVB Text Layout Manager as described in annex U, "Extended graphics APIs".

Other parts of the present document that are related to this topic are:

- For character sets supported by implementations see annex E, "Character set".
- For the font families, sizes, styles and weights supported by implementations see and the presentation of this to the API clause G.4, "Resident fonts and text rendering".
- For the content formats used to deliver fonts see clause 7.4, "Downloadable fonts".

## D.2 Fonts

### D.2.1 Embedded fonts

See clause G.4, "Resident fonts and text rendering".

### D.2.2 Downloaded fonts

#### D.2.2.1 Font technology

See clause 7.4, "Downloadable fonts".

#### D.2.2.2 Font index files

##### D.2.2.2.1 Format of file

The font index file provides a mapping between a font face name and a file containing the font data. The file syntax is defined by the XML DTD shown in table 74.

**Table 74: Font index file syntax definition**

```
<!ELEMENT fontdirectory (font+)>
    <!-- a font definition -->

<!ELEMENT font (name,fontformat,filename,style*,size?)>
    <!-- filename of the font file.
    Because the font directory is per directory, this should
    not contain any directories, but just be a file in that
    directory -->

<!ELEMENT filename (#PCDATA)>
    <!-- font format, e.g. "PFR", "OTF", "TTF" or "TTC" -->

<!ELEMENT fontformat (#PCDATA)>
    <!-- symbolic name of the font -->

<!ELEMENT style (#PCDATA)>
    <!-- font style -->

<!ELEMENT name (#PCDATA)>

<!ELEMENT size EMPTY>
<!ATTLIST size
    min CDATA "0"
    max CDATA "maxint"
>
```

The `PublicLiteral` to be used for specifying this DTD in document type declarations of the XML files is:

```
"-//DVB//DTD Font Directory 1.0//EN"
```

and the URL for the `SystemLiteral` is:

```
"http://www.dvb.org/mhp/dtd/fontdirectory-1-0.dtd"
```

The `Name` used in the document type declaration shall be "fontdirectory".

### D.2.2.2.2 Element semantics

font: There shall be one font element per font file included in the font directory.

name: Contains the font face name of the font (e.g. "Helvetica").

fontformat: The file format of the font. For font formats used in the present document, this shall be the following:

- For OpenType fonts with TrueType outlines, this shall be "TTF" or "OTF".
- For OpenType fonts with TrueType collections, this shall be "TTC" or "OTF".
- For PFR fonts, this shall be "PFR".

filename: Relative path to the font file. This is relative to the directory containing the font index file. The separator character for directories is "/". As this is a relative path, it shall not begin with a "/" character.

style: The style elements contain the names of the styles of the font that are contained in this font file. The possible values for GEM are "PLAIN", "BOLD", "ITALIC" and "BOLD_ITALIC". There is one style element included per style contained in the indicated font file, except when all the usable styles of the font are in the same file in which case the style elements can be left out. When different styles of the font are contained in separate files, these are included in the directory as separate font entities with the same name but different style and filename.

size: Indicates the size range for which this font file can be used. The min. attribute contains the minimum size in points (default is "0"). The max attribute contains the maximum size in points or "maxint" if the maximum size is not limited (default is "maxint").

If all the usable sizes of the font are generated using the same font file, the size element can be left out. If there are separate files for different sizes, these are included in the directory as separate font entities with the same name and style but different size definition and filename.

#### D.2.2.2.3 Example

**Table 75: Example index file**

```
<?xml version="1.0"?>

    <!DOCTYPE fontdirectory PUBLIC "-//DVB//DTD Font Directory 1.0//EN"
    "http://www.dvb.org/mhp/dtd/fontdirectory-1-0.dtd">
    <fontdirectory>
        <font>
            <name>Tiresias</name>
            <fontformat>PFR</fontformat>
            <filename>tiresias.pfr</filename>
            <style>BOLD</style>
        </font>
        <font>
            <name>Broadcaster X Screen Font</name>
            <fontformat>PFR</fontformat>
            <filename>brxsf.pfr</filename>
        </font>
    </fontdirectory>
```

### D.2.2.3 Name and location of font index files

#### D.2.2.3.1 General

The specification of font paths and fonts by an application are private to that application, they are not available to other applications.

#### D.2.2.3.2 Name of file

The file name shall be:

`"dvb.fontindex"`

#### D.2.2.3.3 Location

The font index file shall be placed in the base directory of the application.

### D.2.2.4 Specification of fonts at run time

#### D.2.2.4.1 DVB-J

The implementation locates the font file using the parameters passed to `org.dvb.ui.FontFactory.createFont()`.

The font file is located by searching the directory for a file where the name element matches the name parameter of a call to `FontFactory.createFont`, a `style` element matches the style parameter and the `size` parameter is within the limitations in the size element.

It is font format specific how the font information for a given name, style and size is encoded in the font file. The name and style need to match the corresponding entry in the font file. The font size needs to be within the range supported by the font file.

# D.3 Text rendering

## D.3.1 Low and high level rendering

Two levels of interface are addressed:

### D.3.1.1 Low level rendering

Simple string rendering as supported by:

- `java.awt.Graphics.drawString`
- `java.awt.Graphics.drawChars`
- `java.awt.Graphics.drawBytes`

This is referred to as "low level" rendering implying that the application author has significant responsibilities for ensuring that the text is visible. This rendering obeys the normal AWT rules. For example, the author is responsible for placing individual words or lines of text on to a component.

`org.havi.ui.HDefaultTextLayoutManager` shall also be considered as supporting low level rendering except that implementations shall respect the rendering settings on the `HVisible` passed as argument to the render method (such as alignment, font and foreground colour).

### D.3.1.2 High level rendering

More complex object rendering as supported by:

- `org.dvb.ui.DVBTextLayoutManager`

This is referred to as "high level" rendering implying that the application author may need less effort to ensure that the text is visible. For example, the author could use the text layout manager to handle flowing paragraphs of text into an `org.havi.ui.HText` object.

## D.3.2 Philosophy

This clause describes "logical" rules that ensure text flows predictably on all receivers and defines some rendering requirements to ensure that a minimum acceptable level of text legibility is achieved. The scope of this is much broader for "high level" rendering, where rules address text flow in addition to just string rendering.

No restriction is placed on the rendering technology used in a receiver provided that it achieves the deterministic text flow characteristics and the minimum rendering requirements described in this clause.

### D.3.2.1 High level rendering conceptual process

The conceptual rendering process can be described as follows:

a) Based on:

- The size of the object to render into.
- The characteristics of the font (e.g. see clause D.3.3.1, "Font bounds").
- Any automatic or application controlled insets (see clause D.3.5, "Rendering within limits and insets").

Calculate:

- The maximum number of lines of text that may be rendered (see clause D.3.8.1, "Number of lines").
- The width available for rendering on each line (see clause D.3.9.1, "Available width").

b) Determine how the text to render flows, effectively defining a series of lines to render using:

- The "logical" rules for calculating the width of rendered text (see clause D.3.6, ""logical" text width rules").
- The available width for rendering (see clause D.3.9.1, "Available width").
- The rules for breaking text (see clause D.3.7, "Line breaking").

c) Determine where each line of text to render is placed vertically within the object (see clauseD.3.8, "Positioning lines of text vertically within an object"). This needs to consider that:

- If the number of lines of text to render (from step b) exceeds the maximum number of lines of text that may be rendered (from step a) "vertical truncation" may be required, i.e. discard some of the lines to render.
- The positioning of lines to render is affected by the vertical justification setting of the object.

d) Determine the placement of individual characters in a line to render (see clause D.3.9, "Rendering lines of text horizontally"). This needs to consider that:

- Even having correctly applied the rules relating to the flow of text, in some extreme circumstances the length of the line of text to render can be wider than the available width for rendering (from step a). To handle this "horizontal truncation" may be required, i.e. discard some of the characters from the line to render.
- The placement of characters in the line to render is affected by the horizontal justification setting of the object.
- The placement of characters in the line to render should ensure sensible and consistent spacing between adjacent characters (see clause D.3.12, "Placing runs of characters and words").

In addition special rules exist for handling tabulation (clause D.3.11, "Tabulation") which need to be taken into account in the implementation of many of these steps.

Behaviour equivalent to the above is required to ensure conformant rendering. The order of these steps is illustrative and may vary between implementations for reasons of convenience or efficiency.

## D.3.3 Font Definition

The nomenclature used in this clause is derived from the resident font format(s). The nomenclature and the numerics provided here are directly applicable to downloaded fonts.

### D.3.3.1 Font bounds

The font definition induces a set of parameters `xMin`, `xMax`, `yMin` and `yMax` that are properties of the font. These define the maximum extent of the outline representation of characters within the physical font, and as such are defined in terms of outline resolution units (outlineResolution).

(`xMin`, `yMin`) and (`xMax`, `yMax`) are the bottom-left and top-right corners of an imaginary bounding rectangle within which all characters in the font can be completely enclosed.

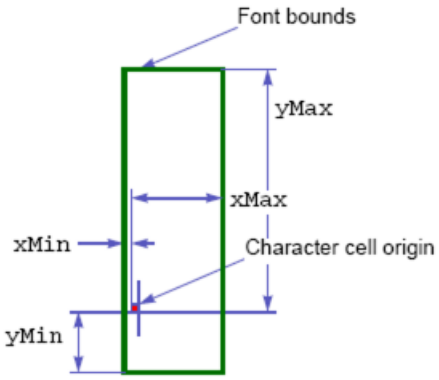



**Figure 37: Font bounds**

In "Low level rendering" the author is responsible for using knowledge of these values to correctly position text.

In "High level rendering" the text layout manager uses this information when managing text flow to guarantee that the extremities of all characters are completely within the object. See clause D.3.5, "Rendering within limits and insets".

### D.3.3.2 "Physical" font data

"Physical" font data such as horizontal escapement and kerning is defined in font metrics resolution units. This is a high-resolution representation, abstracted from any actual rendering system.

In OpenType, the same font resolution units are used for font metric (metric resolution) and outline coordinates (outline resolution), the value is defined by `unitsPerEm` value in the Font Header table. In PFR fonts, the font metrics resolution is defined by the `metricsResolution` field of the Physical Font Record.

NOTE: The `outlineResolution` and `metricsResolution` defined in the PFR are not necessarily the same.

### D.3.3.3 Ligatures and Glyph Substitution

When ligatures and glyph substitution are not required by the font, GEM terminals shall not automatically transform letter pairs to ligatures. If ligatures and glyph substitution are supported by the font format, GEM terminals shall automatically select appropriate glyph forms and transform character combinations to ligatures.

GEM applications that utilise PFR fonts and wish to see ligatures rendered should use the appropriate Unicode value (if available) and shall ensure that the glyph associated with this Unicode character is found in the font being used.

## D.3.4 Converting font metrics to display pixels

Many of the calculations in this clause are in a high resolution physical coordinate system, either metrics or outline resolutions. These values need to be converted into the pixel resolution of the HGraphicsDevice to allow characters to be rendered.

Values in terms of these high level resolutions can be simply converted to values in terms of points by multiplying by the font size (in points) and dividing by the resolution, i.e. metricsResolution or outlineResolution as appropriate. However, this value in points still needs to be converted into a value in pixels.

Computer display systems typically assume a 72 pixel per inch display. So, as each point is 1/72 inch, the horizontal and vertical size of each pixel is 1 point.

### D.3.4.1 Vertical resolution

Each pixel in the graphics device (HGraphicsDevice for DVB-J applications) containing the component is equivalent to a single point.

### D.3.4.2 Horizontal resolution

The horizontal relationship depends on the characteristics of the graphics device (HGraphicsDevice for DVB-J applications). For a square pixel graphics device the 1 pixel = 1 point convention can be preserved.

However, for a graphics device whose pixel aspect ratio is given by `org.havi.ui.HScreenConfiguration.getPixelAspectRatio` the horizontal resolution is the pixel aspect ratio x1 point.

The following table defines the pixel aspect ratios which shall be used when converting typographic pixels for rendering in the graphics system of a GEM terminal for each graphics device aspect ratio.

**Table 76: Pixel width for non-square pixel graphics devices**

| Graphics device resolution | Graphics device aspect ratio | Typographic pixel size width in points |
|---|---|---|
| 720 x 480 | 4:3 | 40/45 |
| | 14:9 | 47/45 (see note) |
| | 16:9 | 53/45 (see note) |
| 720 x 576 | 4:3 | 48/45 |
| | 14:9 | 56/45 |
| | 16:9 | 64/45 |
| NOTE: Approximated value. | | |

Table 76 is not required for GEM terminal specifications that do not support a graphics device resolution of 720x576, however a functional equivalent shall be specified in GEM terminal specifications.

An emulated graphics could be constructed with 14:9 aspect ratio. This could be used where text is required to be acceptable when viewed on either a 4:3 or 16:9 display. The 14:9 aspect ratio is an artificial construct intended to allow easier authoring but text will always be slightly distorted when displayed. 14:9 should be used when maintaining character aspect ratio is less important to the application than text occupying the same number of pixels regardless of the display size which the GEM terminal is using. A possible example of this is illustrated in figure 38.

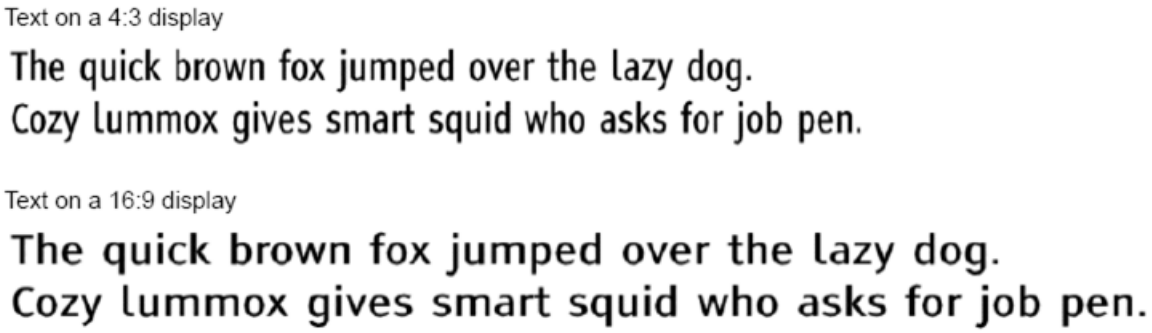



**Figure 38: Example of 14:9 text on either 4:3 or 16:9 display**

## D.3.5 Rendering within limits and insets

When typesetting for print, character extremities may extend beyond the nominal text flow area. However, print has margins so the edge of the text flow is not the technical limit to the area that can be printed.

### D.3.5.1 Low level rendering

In "Low level rendering" the author is responsible for placing the text so that it is not clipped.

## D.3.5.2 High level rendering

In "High level rendering" text is automatically rendered with at least a sufficient inset from the object edge (derived from the font properties `xMin`, `yMin`, `xMax` and `yMax`) to ensure that all presented characters are completely rendered within the bounds of the object. Applications can increase or decrease this minimum (font property determined) inset in the following ways:

- The `Insets` parameter on the `org.dvb.ui.DVBTextLayoutManager.render` method.
- The `org.dvb.ui.DVBTextLayoutManager.setInsets` method.

Each of these mechanisms can be used independently. Their effect is cumulative.

For simplicity, these application controlled insets are not described in the specification clauses that follow. In relation to the rendering process their behaviour is equivalent to a reduction in the size of the object within which the rendering processes acts.

## D.3.5.3 Conversion of units

As stated previously, these parameters are defined in outline resolution units and so need to be converted to device pixels. Based on the principles described previously (see clause D.3.4, "Converting font metrics to display pixels") this can be achieved by the following:

### D.3.5.3.1 yOffsetTop

$$\text{yOffsetTop}_{\text{pixels}} = \begin{cases} div(\text{yMax}_{\text{outlineResolution}} \times \text{fontSize}, \text{outlineResolution}) & \text{yMax} > 0 \\ 0 & \text{yMax} \le 0 \end{cases}$$

### D.3.5.3.2 yOffsetBottom

$$\text{yOffsetBotom}_{\text{pixels}} = \begin{cases} div(-\text{yMin}_{\text{outlineResolution}} \times \text{fontSize}, \text{outlineResolution}) & \text{yMin} < 0 \\ 0 & \text{yMin} \ge 0 \end{cases}$$

### D.3.5.3.3 xOffsetLeft

#### D.3.5.3.3.1 With 14:9 graphics

For 720x576 graphics device resolution:

$$\text{xOffsetLeft}_{\text{pixels}} = \begin{cases} div(-\text{xMin}_{\text{outlineResolution}} \times \text{fontSize} \times 45, \text{outlineResolution} \times 56) & \text{xMin} < 0 \\ 0 & \text{xMin} \ge 0 \end{cases}$$

For 720x480 graphics device resolution:

$$\text{xOffsetLeft}_{\text{pixels}} = \begin{cases} div(-\text{xMin}_{\text{outlineResolution}} \times \text{fontSize} \times 45, \text{outlineResolution} \times 47(56)) & \text{xMin} < 0 \\ 0 & \text{xMin} \ge 0 \end{cases}$$

#### D.3.5.3.3.2 With 4:3 graphics

For 720x576 graphics device resolution:

$$\text{xOffsetLeft}_{\text{pixels}} = \begin{cases} div(-\text{xMin}_{\text{outlineResolution}} \times \text{fontSize} \times 45, \text{outlineResolution} \times 48) & \text{xMin} < 0 \\ 0 & \text{xMin} \ge 0 \end{cases}$$

For 720x480 graphics device resolution:

$$\text{xOffsetLeft}_{\text{pixels}} = \begin{cases} div(-\text{xMin}_{\text{outlineResolution}} \times \text{fontSize} \times 45, \text{outlineResolution} \times 40) & \text{xMin} < 0 \\ 0 & \text{xMin} \geq 0 \end{cases}$$

#### D.3.5.3.3.3 With 16:9 graphics

For 720x576 graphics device resolution

$$\text{xOffsetLeft}_{\text{pixels}} = \begin{cases} div(-\text{xMin}_{\text{outlineResolution}} \times \text{fontSize} \times 45, \text{outlineResolution} \times 64) & \text{xMin} < 0 \\ 0 & \text{xMin} \geq 0 \end{cases}$$

For 720x480 graphics device resolution

$$\text{xOffsetLeft}_{\text{pixels}} = \begin{cases} div(-\text{xMin}_{\text{outlineResolution}} \times \text{fontSize} \times 45, \text{outlineResolution} \times 53) & \text{xMin} < 0 \\ 0 & \text{xMin} \geq 0 \end{cases}$$

### D.3.5.3.4 outlineResolution

outlineResolution is extracted from the font data (see clauses 11.4.1.5, "Font bindings" and D.3.3.2, ""Physical" font data"), div (A, B) = ceil (A / B), "/" is a rational divide and ceil(A) is the first integral number greater than or equal to A.

### D.3.5.3.5 Font bounds

*yOffsetTop*, *yOffsetBottom* and *xOffsetLeft* are all greater than or equal to zero and represent the number of available pixels required above, below and to the left of the character origin to prevent clipping of any character in the font. A value *xOffsetRight* could be defined along similar lines but is not relevant to GEM.

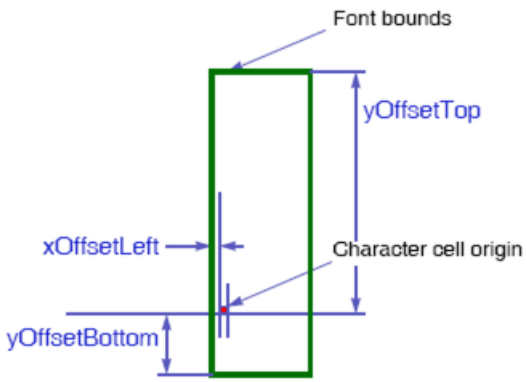



**Figure 39: Font bounds**

## D.3.6 "logical" text width rules

This clause applies to both "Low level rendering" and "High level rendering". Its purpose is to ensure that text will flow predictably on different receivers and authoring stations, regardless of the quality of the character rendering, a set of "logical" text width rules are defined here.

NOTE: I.e. lines and words will break at the same character position.

These rules are a simplification of the rules that might be applied in a typographic rendering system. The objective of these simplifications is to reduce the receiver complexity required to ensure exact correlation of text flow behaviour.

The calculation of "logical" text width is based on "physical" font data. This data provides a description of the font at a very high resolution, abstracted from any actual rendering system. Consequently, the calculation of the "logical" width of a string of characters involves, computing their width at this high resolution and then converting to units appropriate to the rendering system, e.g. device pixels, before making decisions about text flow (see clause D.3.4, "Converting font metrics to display pixels").

In the case of "Low level rendering" it defines the internal computation performed by the AWT routines that measure the width of text:

- `java.awt.FontMetrics.charWidth.`
- `java.awt.FontMetrics.stringWidth.`
- `java.awt.FontMetrics.bytesWidth.`

Due to the rounding processes within the calculations invoking these methods on subsets of a string may not yield the same total result as invoking the methods on the complete string. In particular the total of the values returned by `java.awt.FontMetrics.getWidths` may be different from the value returned by `java.awt.FontMetrics.stringWidth` for the same string.

In the case of "High level rendering" it defines the computations that the layout manager uses in the following cases:

- To determine when to wrap lines of text within an object.
- To determine which tab stops text has passed when implementing tab characters.

## D.3.6.1 Computing "logical" text width

The key parameters when calculating the width of a string of N characters are:

- Text font size.
- charSetWidth.
- The metricsResolution.
- Any kerning adjustment.

### D.3.6.1.1 Font sizes

Font sizes are expressed as the size of an "Em" in units of "points".

NOTE 1: Broadly speaking an Em is the minimum distance between the baselines of consecutive lines of text in the given font. If text is 48 point then the Em at that size is 48 points.

NOTE 2: The point is an archaic typographical unit. Traditionally there were 72,27 points to an inch. Computerized systems now use 72 points per inch for simplicity.

### D.3.6.1.2 Character widths

The font definition gives the width of each character relative to the size of an Em in metricsResolution units.

NOTE: If metrics are specified in 1/1 000 ths of an Em a character with a width of 0,6 Em will have a set width of 600.

### D.3.6.1.3 Kerning

For certain character combinations (a "kerning pair") a kerning adjustment may also be provided. Typically kerning reduces character spacing for pairs such as AV instead of A V, these provide a signed adjustment to the nominal charSetWidth of the first character.

Like charSetWidth kerning adjustments are in terms of metricsResolution units.

Kerning adjustments only apply between non whitespace characters, not between the start of a line of text and the edge of the object. If justification is being used, only whitespace between words may be adjusted.

#### D.3.6.1.4 Letter spacing

Letter spacing allows for expansion/condensation of the character spacing for all of the characters in a object including whitespace.

NOTE: In printing, this is sometimes referred to as tracking.

### D.3.6.2 Logical text width

The equation below shows how the width of a string of N characters is computed.

$$\text{logical width of N characters}_{\text{points}} =$$

$$div((N-1) \times \text{letterspace}, 256) \div div\left(\text{fontsize} \times \left(\sum_{1}^{N} \text{charSetWidth}\left[\right] + \sum_{1}^{N-1} \text{kern}\left[i+1\right]\right), \text{metricsResolution}\right)$$

$$\text{logical width of N characters}_{\text{pixels}} = div\left(\text{logical width of N characters}_{\text{points}} \times H, W\right)$$

Where in div( A, B ):

- B is unsigned and A is signed; and
- div( A, B ) = ceil( A / B ).

Where '/' is a rational divide and ceil(A) is the first integral number greater than or equal to A. So, the calculations round up when reducing precision and tend to over estimate the width of text.

Where H and W are respectively the height and width returned by `org.havi.ui.HScreenConfiguration.getPixelAspectRatio()`.

## D.3.7 Line breaking

### D.3.7.1 Text wrapping setting is false

When the text wrapping setting is false, text shall break onto a new line only where a Carriage Return character is present in the text. The number of lines to render shall be equal to the number of Carriage Returns present, plus one.

NOTE: The "plus one" ensures that the last line of any text can be presented even if it has not been terminated with a Carriage Return.

### D.3.7.2 Text wrapping setting is true

When the text wrapping setting is true, the text flow may break on to new lines at positions in addition to those caused by Carriage Return characters. These additional break positions are defined below. This behaviour is independent of the object's horizontal alignment settings and is considered to take place before any truncation (see clause D.3.8.2, "Truncation").

NOTE 1: This clause may not apply for non-western languages. GEM terminal specifications that support languages for which this clause does not apply, provide a functional equivalent.

The effect of text wrapping on the rendering process is equivalent to substituting "breaking characters" (defined in table 77 "Special characters") with Carriage Return characters at positions determined by the wrapping rules.

The behaviour of the wrapping process is described below:

a) Based on the "logical" width of the text, receivers shall determine for each line, the first contiguous sequence of non-breaking characters that meets both of the following requirements:

   - Would not completely fit within the available width (see clause D.3.9.1, "Available width").
   - That follows one or more breaking characters.

   If such a sequence exists, the breaking character preceding it shall be replaced with a Carriage Return character.

b) All trailing breaking characters shall be discarded from all lines of text. (Preceding breaking characters are not affected).

NOTE 2: In general use this identifies the word in a body of text that if rendered would cause the current line to spill outside of the text object. This word becomes the first word in the next line. However, if a line starts with a single word that is longer than the width of the object then the line is broken just before a following word.

EXAMPLE 1: This illustrates the most common case where the word "won't" does not fit and so is broken to the next line.

EXAMPLE 2: This illustrates the particular case of a long first word. In this case "is" is the first word that does not fit AND is preceded by a breaking character.

EXAMPLE 3: This illustrates the consumption of trailing spaces. The line breaks just before the second "spaces" (so the next line starts with this word) and all trailing spaces (e.g. after the first "spaces,") are discarded).

NOTE 3: The importance of the removal of trailing breaking characters is particularly visible if the text is centred or right justified. For example, the text to be centred becomes "spaces" rather than "  spaces".

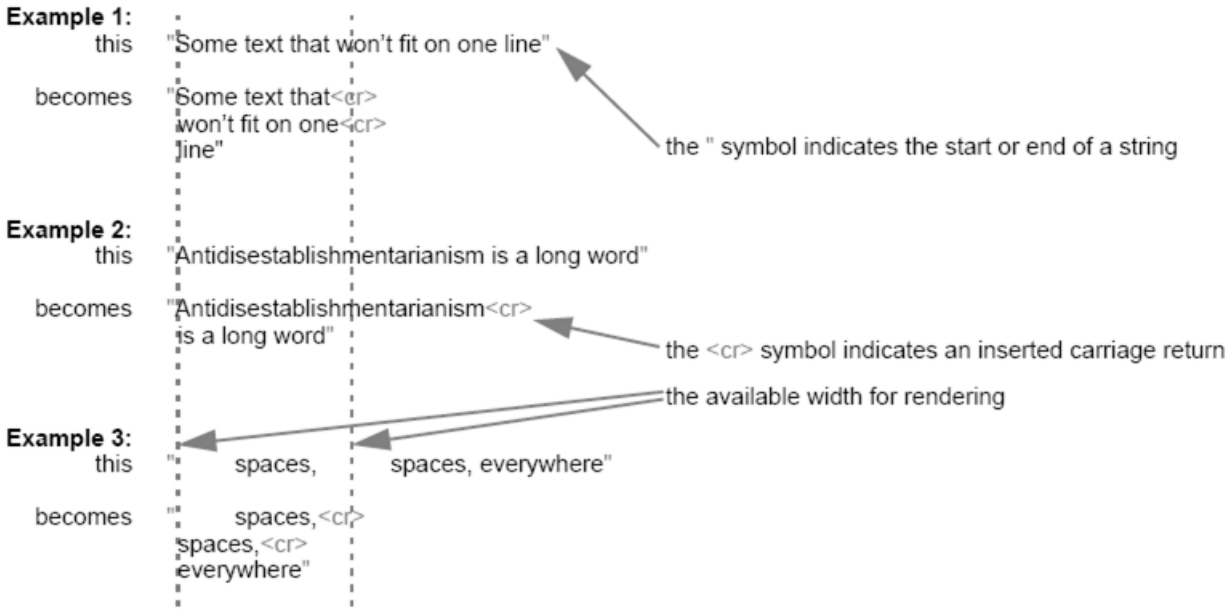



**Figure 40: Text wrapping examples**

After text wrapping the text truncation rules have to be considered. For example, in example 2 in figure 40 the word "Antidisestablishmentarianism" will be truncated (see clause D.3.9.2, "Truncation").

## D.3.8 Positioning lines of text vertically within an object

### D.3.8.1 Number of lines

Assuming that all characters in a line of text share a common baseline then, regardless of the vertical alignment setting the number of lines of text that can be presented within an object is:

```
num_lines = floor((object_height - (yOffsetBottom + yOffsetTop)) / linespace) + 1
```

All values are in pixels.

object_height: is the height of the component less any margin.

linespace: is an attribute of the object that defines the space between the baselines of consecutive lines of text. This is defined in units of points but is converted to pixels (see clause D.3.4, "Converting font metrics to display pixels").

The function floor(A) rounds A to the first integral number less than or equal to A.

### D.3.8.2 Truncation

When the number of lines of text to be rendered exceeds the available height within the object then lines of text shall be discarded as follows:

- If the vertical alignment setting is VERTICAL_CENTER or VERTICAL_START_ALIGN then lines are discarded from the end of the text.
- If the vertical alignment setting is VERTICAL_END_ALIGN then lines are discarded from the beginning of the text.

This truncation will result in a number of lines that can be completely displayed within the object. These shall be presented according to clause D.3.8.3, "Positioning".

See also clause D.3.10, "Text overflow".

### D.3.8.3 Positioning

The origin of any character shall be at least yOffsetTop inside the top edge of the object and at least yOffsetBottom inside its bottom edge. The spacing between the baselines of text shall be `linespace`.

#### D.3.8.3.1 Vertical alignment setting is VERTICAL_START_ALIGN

When the text is top aligned the baseline of the top most line shall be yOffsetTop below the top of the object.

#### D.3.8.3.2 Vertical alignment setting is VERTICAL_END_ALIGN

When the text is bottom aligned the baseline of the bottom most line shall be yOffsetBottom above the bottom of the object.

#### D.3.8.3.3 Vertical alignment setting is VERTICAL_CENTER

When the text is vertically centred the positioning of the lines shall be such that the space above the first line of text equals the space below the last line of text (to allow for rounding the lower gap may be up to one pixel greater than the upper gap).

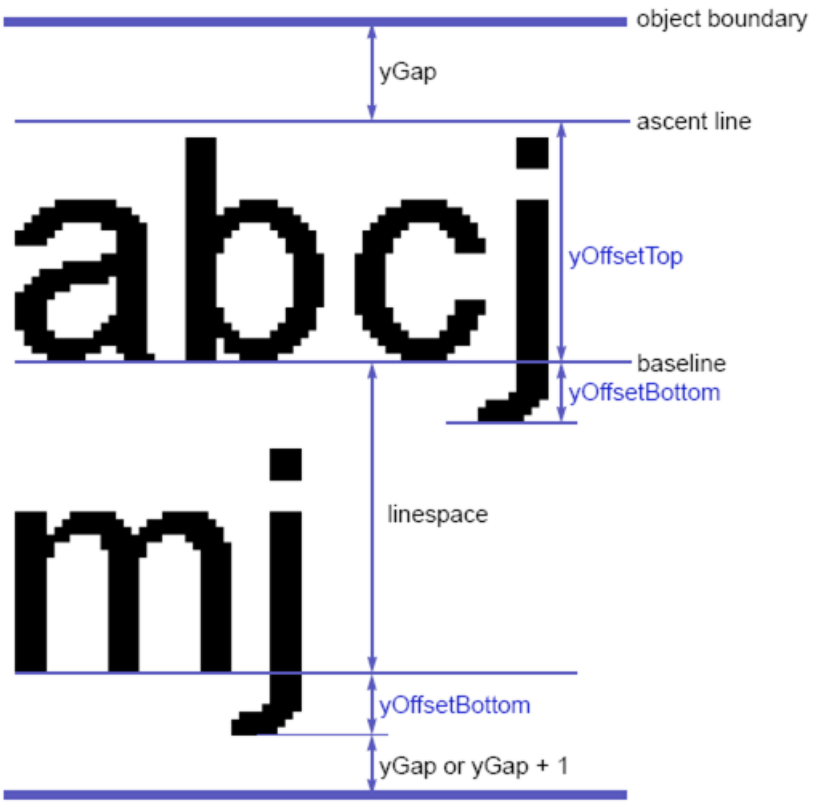



**Figure 41: Vertical measures**

### D.3.8.3.4 Examples

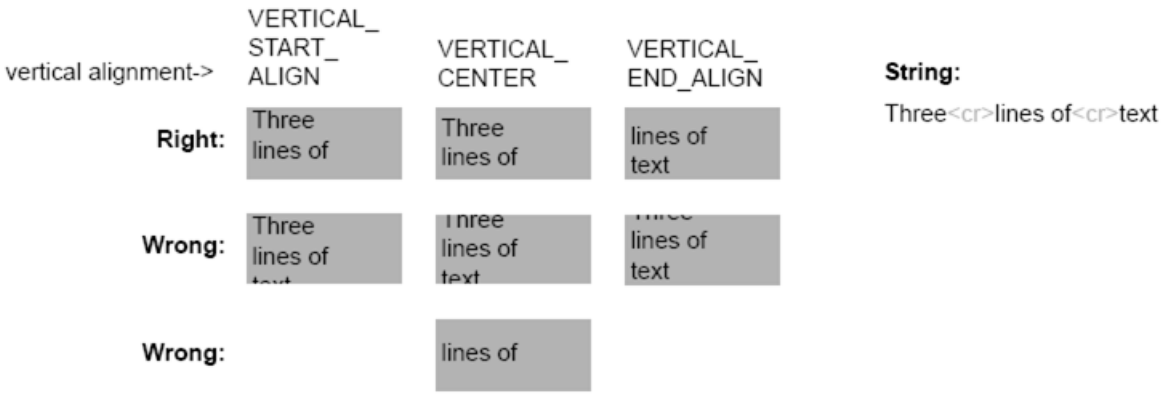



**Figure 42: Vertical positioning example 1**

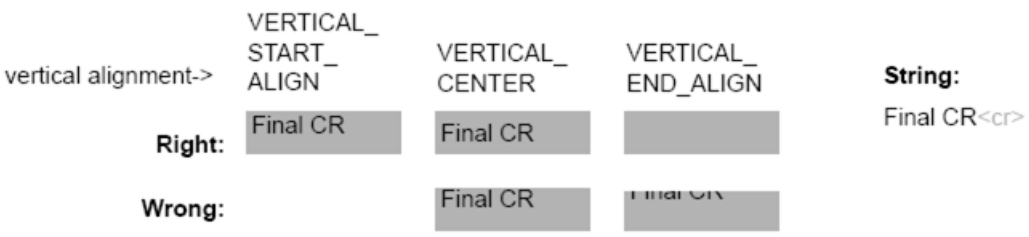



**Figure 43: Vertical positioning example 2**

# D.3.9 Rendering lines of text horizontally

## D.3.9.1 Available width

The number of characters that may be rendered on a line is not simply dependent upon the width of the object and the horizontal escapement for each character, but also needs to consider that the rendering of the first character in a line may extend to the left of its origin. Thus, regardless of the horizontal alignment setting the space available for rendering a line of text within an object is:

```
available_width = object_width - xOffsetLeft
```

All values are in pixels. This derivation of `available_width` shall be used with the "logical" text width rules to determine text flow.

## D.3.9.2 Truncation

Where a line of text is too long to fit within the `available_width` of the object, the line shall be truncated.

NOTE: This addresses the case where text wrapping has not made the text fit. This can happen either because the text wrapping setting is false, or because a single word is too long to fit.

The result shall be as if the complete line were aligned appropriately on the object (taking into account the horizontal alignment setting) with only those characters which can be completely presented being rendered. That is those characters:

- Whose origin is more than xOffsetLeft right of the left hand edge of the object.
- Whose right hand edge falls inside the right hand edge of the object.

For example:

- If the horizontal alignment setting is `HORIZONTAL_END_ALIGN` then a portion of text from the right end of the line will be rendered with its edge aligned to the right edge of the object.
- If the horizontal alignment setting is `HORIZONTAL_CENTER` the centre of the string will appear at the centre of the object with excess characters being truncated from each end.

See also clause D.3.10, "Text overflow".

## D.3.9.3 Placement

- When the horizontal alignment setting is `HORIZONTAL_START_ALIGN` the origin of the left most character shall be xOffsetLeft inside the left edge of the object.
- When the horizontal alignment setting is `HORIZONTAL_CENTER` the positioning of the characters shall be such that the gap to the left of the text equals the gap to the right (to within one pixel).
- When the horizontal alignment setting is `HORIZONTAL_END_ALIGN` the origin of the right most character shall be as necessary to ensure that it is completely visible when rendered.

NOTE 1: xOffsetLeft allows for characters (such as capital "J") that extend to the left of their origin. xOffsetLeft is a property of the font and so represents a consistent indent, i.e. it is independent of the first character.

NOTE 2: There are characters (e.g. fractional divisor) that can extend beyond the escapement indicated by their charSetWidth value. These can theoretically extend beyond the right hand edge of the object if the character is at the end of a word and that word is at the end of a line and the "logical width" of the line just fits the object. With the "fractional divisor" character this is a very unlikely combination of circumstances as this character is normally embedded within a "word". The handling of cases such as this by the GEM terminal is implementation dependent.

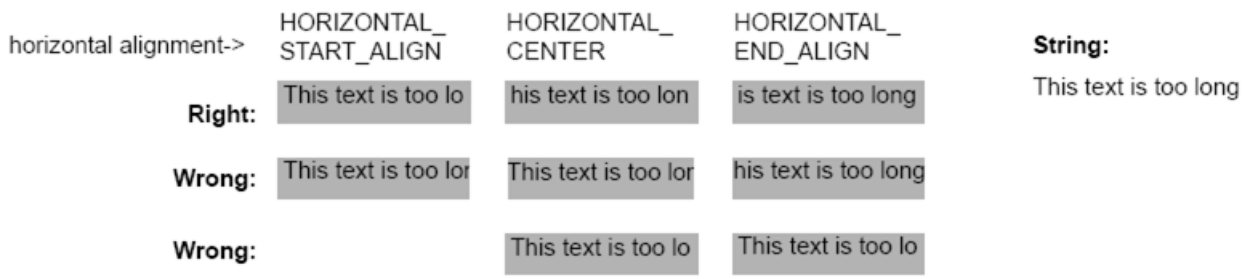



**Figure 44: Horizontal positioning examples**

## D.3.10 Text overflow

The `org.dvb.ui.TextOverflowListener.notifyTextOverflow` method is called if truncation occurs.

NOTE: This can be used by an application to alert the viewer or take other remedial action.

## D.3.11 Tabulation

In left aligned text (the horizontal alignment setting is `HORIZONTAL_START_ALIGN`) tab stops are defined at intervals of `horizontalTabSpace` from the left edge of the object. By default the value of `horizontalTabSpace` is 56 points. "Horizontal Tabulation" advances the origin of the next character to be rendered to the next tab stop in the direction that the text is currently flowing (the character repertoire in Table 81 only requires left to right text).

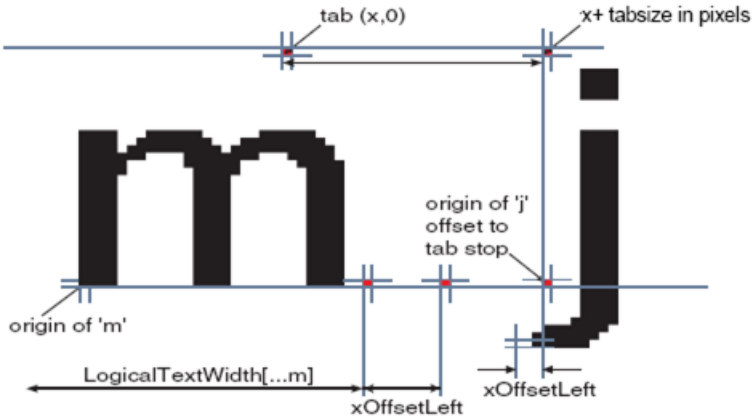



**Figure 45: Effect of horizontal tabulation**

- Tab characters only have meaning in left aligned text (the horizontal alignment setting is `HORIZONTAL_START_ALIGN`). If the text is right aligned, centred or justified then tab character shall be treated as a space character.
- A tab logically advances the rendering of the text by at least the width xOffsetLeft. If the normal origin of the next character to be rendered after the tab character is after a tab stop, a tab character will advance the rendering to the subsequent tab stop.
- The tab stops are at regular intervals from the left edge of the object and are not affected by the xOffsetLeft offset to the origin of the first character.
- The tab size of 56 points shall be converted to pixels using table 76.

## D.3.12 Placing runs of characters and words

A run of characters starts from a well defined point:

- The start edge of the object.
- A tab stop.

After this origin the fine positioning of character cells and the gaps between words is not fully specified.

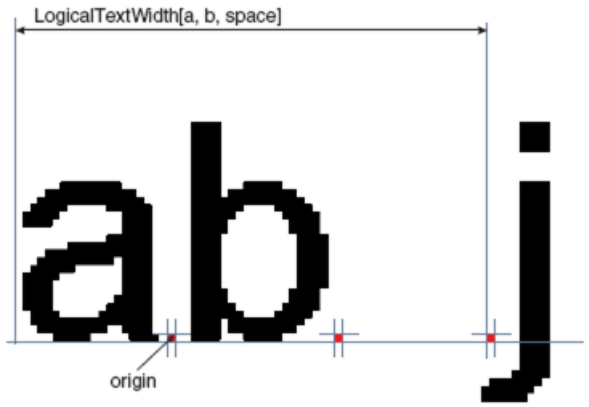



**Figure 46: Calculation of character placement**

However, the following rendering requirements shall be observed to ensure that a minimum acceptable level of text legibility is achieved:

- The spacing between any pair of non whitespace characters should be consistent wherever that pair of characters is displayed.
- At the default character spacing no two non-whitespace characters should "appear" to touch, except for special cases such as where ligatures dipthongs etc. are being used. The definition of the underlying font may make this requirement impossible.
- The physical rendering of a run of text as determined by the "logical" rules shall be achieved completely within the space used for the "logical" calculation.
- No partially rendered characters shall be presented.

NOTE: The "logical" position of each character as determined by the "logical" rules is likely to be a non-integer. For renderers unable to support suitable sub-pixel positioning, e.g. a 1-bit/pixel bitmap renderer, this is impossible to implement. Thus, whilst the "logical" rules are used to determine the flow of text, they are not required to be observed for individual character placement within this flow and other strategies may be employed as long as the requirements above are satisfied.

## D.3.13 Control of text flow

See the definition of the DVB text layout manager.

# D.4 Text mark-up

This clause on text mark-up applies to "High level rendering".

## D.4.1 White Space Characters

Certain non-printing characters have special meaning. These are identified in table 77.

**Table 77: Special characters**

| UTF8 Value(s) | Character | Name | Breaking/ Non-breaking | Meaning |
|---|---|---|---|---|
| 0x09 | 0x0009 | Tab | Breaking | See clause D.3.11, "Tabulation". |
| 0x0A | 0x000A | Carriage Return | N/A | Causes the text flow to break. The origin for the next character to be rendered moves to a new baseline "linespace" below that just rendered. The horizontal position of the next line will depend on the horizontal alignment setting. |
| 0x0D | 0x000D | | | |
| 0x0D0A (see note) | 0x000D 0x000A | | | |
| 0x20 | 0x0020 | Space | Breaking | Spaces text by the width defined for the space character. When an object has its text wrapping setting set to "true", lines may be broken at a space. See clause D.3.7, "Line breaking". |
| 0xC2A0 | 0x00A0 | Non-breaking Space | Non-breaking | Identical spacing characteristics to 0x20 but is not seen as word boundary for deciding a position to break a line of text. (0xC2A0 is the UTF-8 representation of 0x00A0) |
| 0xE28087 | 0x2007 | Figure Space | Non-breaking | Can be used in a string of numerals as an alternative to using comma to denote "thousands". This character is not treated as a word boundary when deciding a position to break a line of text. |
| NOTE: The character sequence 0x0D0A shall be rendered identically to a single 0x0D. | | | | |

## D.4.2 Marker characters

The codes 0x1C to 0x1F are zero width, non-spacing, non-printing characters available for use by authors as markers in objects, i.e. when using string operations.

## D.4.3 Non-printing characters

Certain characters (or character sequences) have no immediate visual representation.

These include:

- 0x1C to 0x1F marker characters (see clause D.4.2).
- Format control mark-up (see clause D.4.4).
- Other characters not recognized by the receiver.

When presenting text that includes these characters the character placement shall be as if the non-printing characters were eliminated from the text before rendering. In particular, the character spacing and inter character kerning shall be computed as if the non-printing characters were not present.

## D.4.4 Format Control Mark-up

Within objects mark-up codes can be used to control the presentation of text. The sequence in table 78 marks the start of some marked-up text. For each "start of mark-up" a corresponding "end of mark-up" is defined. The byte sequence for the "end of mark-up" is illustrated in Table 79. The minimum number of supported mark-up instances, where each instance is a start and end mark-up pair, is 256.

**Table 78: General format for start of text mark-up**

| | Bits | Value | Note |
|---|---|---|---|
| start_of_markup | 8 | 0x1B | Escape |
| markup_start_identifier | 8 | 0x40 to 0x5E | "@" to "^" |
| parameters_length | 8 | N | |
| for( i=0; i<N; i++ ) { | | | |
| parameter_byte | 8 | 0x00 to 0xFF | |
| } | | | |

**Table 79: General format for end of text mark-up**

| | Bits | Value | Note |
|---|---|---|---|
| end_of_markup | 8 | 0x1B | Escape |
| markup_end_identifier | 8 | 0x60 to 0x7E | "`" to "~" |

**Table 80: Codes defined for use in marked-up text files**

| Min. Nesting | Start mark-up | End mark-up | Description |
|---|---|---|---|
| | 0x1B 0x42 0x00 | 0x1B 0x62 | Applies "bold" style to the text enclosed (see note 1) |
| 16 | 0x1B 0x43 0x04 0xrr 0xgg 0xbb 0xtt | 0x1B 0x63 | Applies colour to the text enclosed. 0xrr specifies the red intensity, 0xgg the green, 0xbb the blue and 0xtt the transparency (where 0x00 is fully opaque and 0xff fully transparent). |
| NOTE 1: Not supported in this version of GEM.<br>NOTE 2: The required text encoding for this mark-up format is described in clause 11.2.11, "Text Encodings". | | | |

## D.4.5 Future compatibility

Compatible extensions to the set of mark-up codes may be defined in future versions of GEM. For each the `markup_end_identifier` will be 32 (0x20) greater than the `markup_start_identifier`. GEM terminals shall ignore unrecognized mark-up and shall display any text enclosed within an unrecognized mark-up.

# Annex E (normative): Character set

A GEM implementation shall be able to *display* and *accept as input* at least the set of characters shown in table 81.

The range of characters accepted as input may be limited by the capabilities of the available input devices. This clause does not imply that keyboards (real and virtual) for GEM terminals are required to support input of this full character set.

**Table 81: Extended Latin set**

| Code | ISO/IEC 10646-1 [15] Character name |
|---|---|
| 0x0020 | SPACE |
| 0x0021 | EXCLAMATION MARK |
| 0x0022 | QUOTATION MARK |
| 0x0023 | NUMBER SIGN |
| 0x0024 | DOLLAR SIGN |
| 0x0025 | PERCENT SIGN |
| 0x0026 | AMPERSAND |
| 0x0027 | APOSTROPHE |
| 0x0028 | LEFT PARENTHESIS |
| 0x0029 | RIGHT PARENTHESIS |
| 0x002A | ASTERISK |
| 0x002B | PLUS SIGN |
| 0x002C | COMMA |
| 0x002D | HYPHEN-MINUS |
| 0x002E | FULL STOP |
| 0x002F | SOLIDUS |
| 0x0030 | DIGIT ZERO |
| 0x0031 | DIGIT ONE |
| 0x0032 | DIGIT TWO |
| 0x0033 | DIGIT THREE |
| 0x0034 | DIGIT FOUR |
| 0x0035 | DIGIT FIVE |
| 0x0036 | DIGIT SIX |
| 0x0037 | DIGIT SEVEN |
| 0x0038 | DIGIT EIGHT |
| 0x0039 | DIGIT NINE |
| 0x003A | COLON |
| 0x003B | SEMICOLON |
| 0x003C | LESS-THAN SIGN |
| 0x003D | EQUALS SIGN |
| 0x003E | GREATER-THAN SIGN |
| 0x003F | QUESTION MARK |
| 0x0040 | COMMERCIAL AT |
| 0x0041 | LATIN CAPITAL LETTER A |
| 0x0042 | LATIN CAPITAL LETTER B |
| 0x0043 | LATIN CAPITAL LETTER C |
| 0x0044 | LATIN CAPITAL LETTER D |
| 0x0045 | LATIN CAPITAL LETTER E |
| 0x0046 | LATIN CAPITAL LETTER F |
| 0x0047 | LATIN CAPITAL LETTER G |
| 0x0048 | LATIN CAPITAL LETTER H |
| 0x0049 | LATIN CAPITAL LETTER I |
| 0x004A | LATIN CAPITAL LETTER J |
| 0x004B | LATIN CAPITAL LETTER K |
| 0x004C | LATIN CAPITAL LETTER L |
| 0x004D | LATIN CAPITAL LETTER M |
| 0x004E | LATIN CAPITAL LETTER N |
| 0x004F | LATIN CAPITAL LETTER O |
| 0x0050 | LATIN CAPITAL LETTER P |
| 0x0051 | LATIN CAPITAL LETTER Q |

| Code | ISO/IEC 10646-1 [15] Character name |
|---|---|
| 0x0052 | LATIN CAPITAL LETTER R |
| 0x0053 | LATIN CAPITAL LETTER S |
| 0x0054 | LATIN CAPITAL LETTER T |
| 0x0055 | LATIN CAPITAL LETTER U |
| 0x0056 | LATIN CAPITAL LETTER V |
| 0x0057 | LATIN CAPITAL LETTER W |
| 0x0058 | LATIN CAPITAL LETTER X |
| 0x0059 | LATIN CAPITAL LETTER Y |
| 0x005A | LATIN CAPITAL LETTER Z |
| 0x005B | LEFT SQUARE BRACKET |
| 0x005C | REVERSE SOLIDUS |
| 0x005D | RIGHT SQUARE BRACKET |
| 0x005E | CIRCUMFLEX ACCENT |
| 0x005F | LOW LINE |
| 0x0060 | GRAVE ACCENT |
| 0x0061 | LATIN SMALL LETTER A |
| 0x0062 | LATIN SMALL LETTER B |
| 0x0063 | LATIN SMALL LETTER C |
| 0x0064 | LATIN SMALL LETTER D |
| 0x0065 | LATIN SMALL LETTER E |
| 0x0066 | LATIN SMALL LETTER F |
| 0x0067 | LATIN SMALL LETTER G |
| 0x0068 | LATIN SMALL LETTER H |
| 0x0069 | LATIN SMALL LETTER I |
| 0x006A | LATIN SMALL LETTER J |
| 0x006B | LATIN SMALL LETTER K |
| 0x006C | LATIN SMALL LETTER L |
| 0x006D | LATIN SMALL LETTER M |
| 0x006E | LATIN SMALL LETTER N |
| 0x006F | LATIN SMALL LETTER O |
| 0x0070 | LATIN SMALL LETTER P |
| 0x0071 | LATIN SMALL LETTER Q |
| 0x0072 | LATIN SMALL LETTER R |
| 0x0073 | LATIN SMALL LETTER S |
| 0x0074 | LATIN SMALL LETTER T |
| 0x0075 | LATIN SMALL LETTER U |
| 0x0076 | LATIN SMALL LETTER V |
| 0x0077 | LATIN SMALL LETTER W |
| 0x0078 | LATIN SMALL LETTER X |
| 0x0079 | LATIN SMALL LETTER Y |
| 0x007A | LATIN SMALL LETTER Z |
| 0x007B | LEFT CURLY BRACKET |
| 0x007C | VERTICAL LINE |
| 0x007D | RIGHT CURLY BRACKET |
| 0x007E | TILDE |
| 0x00A0 | NO-BREAK SPACE |
| 0x00A1 | INVERTED EXCLAMATION MARK |
| 0x00A2 | CENT SIGN |
| 0x00A3 | POUND SIGN |
| 0x00A4 | CURRENCY SIGN |
| 0x00A5 | YEN SIGN |
| 0x00A8 | DIAERESIS |
| 0x00A9 | COPYRIGHT SIGN |
| 0x00AA | FEMININE ORDINAL INDICATOR |
| 0x00AB | LEFT-POINTING DOUBLE ANGLE QUOTATION MARK |
| 0x00AC | NOT SIGN |
| 0x00AD | SOFT HYPHEN |
| 0x00AE | REGISTERED SIGN |
| 0x00B0 | DEGREE SIGN |
| 0x00B4 | ACUTE ACCENT |
| 0x00B6 | PILCROW SIGN |
| 0x00B7 | MIDDLE DOT |
| 0x00B8 | CEDILLA |
| 0x00BA | MASCULINE ORDINAL INDICATOR |

| Code | ISO/IEC 10646-1 [15] Character name |
|---|---|
| 0x00BB | RIGHT-POINTING DOUBLE ANGLE QUOTATION MARK |
| 0x00BC | VULGAR FRACTION ONE QUARTER |
| 0x00BD | VULGAR FRACTION ONE HALF |
| 0x00BE | VULGAR FRACTION THREE QUARTERS |
| 0x00BF | INVERTED QUESTION MARK |
| 0x00C0 | LATIN CAPITAL LETTER A WITH GRAVE |
| 0x00C1 | LATIN CAPITAL LETTER A WITH ACUTE |
| 0x00C2 | LATIN CAPITAL LETTER A WITH CIRCUMFLEX |
| 0x00C3 | LATIN CAPITAL LETTER A WITH TILDE |
| 0x00C4 | LATIN CAPITAL LETTER A WITH DIAERESIS |
| 0x00C5 | LATIN CAPITAL LETTER A WITH RING ABOVE |
| 0x00C6 | LATIN CAPITAL LETTER AE |
| 0x00C7 | LATIN CAPITAL LETTER C WITH CEDILLA |
| 0x00C8 | LATIN CAPITAL LETTER E WITH GRAVE |
| 0x00C9 | LATIN CAPITAL LETTER E WITH ACUTE |
| 0x00CA | LATIN CAPITAL LETTER E WITH CIRCUMFLEX |
| 0x00CB | LATIN CAPITAL LETTER E WITH DIAERESIS |
| 0x00CC | LATIN CAPITAL LETTER I WITH GRAVE |
| 0x00CD | LATIN CAPITAL LETTER I WITH ACUTE |
| 0x00CE | LATIN CAPITAL LETTER I WITH CIRCUMFLEX |
| 0x00CF | LATIN CAPITAL LETTER I WITH DIAERESIS |
| 0x00D0 | LATIN CAPITAL LETTER ETH |
| 0x00D1 | LATIN CAPITAL LETTER N WITH TILDE |
| 0x00D2 | LATIN CAPITAL LETTER O WITH GRAVE |
| 0x00D3 | LATIN CAPITAL LETTER O WITH ACUTE |
| 0x00D4 | LATIN CAPITAL LETTER O WITH CIRCUMFLEX |
| 0x00D5 | LATIN CAPITAL LETTER O WITH TILDE |
| 0x00D6 | LATIN CAPITAL LETTER O WITH DIAERESIS |
| 0x00D7 | MULTIPLICATION SIGN |
| 0x00D8 | LATIN CAPITAL LETTER O WITH STROKE |
| 0x00D9 | LATIN CAPITAL LETTER U WITH GRAVE |
| 0x00DA | LATIN CAPITAL LETTER U WITH ACUTE |
| 0x00DB | LATIN CAPITAL LETTER U WITH CIRCUMFLEX |
| 0x00DC | LATIN CAPITAL LETTER U WITH DIAERESIS |
| 0x00DD | LATIN CAPITAL LETTER Y WITH ACUTE |
| 0x00DE | LATIN CAPITAL LETTER THORN |
| 0x00DF | LATIN SMALL LETTER SHARP S |
| 0x00E0 | LATIN SMALL LETTER A WITH GRAVE |
| 0x00E1 | LATIN SMALL LETTER A WITH ACUTE |
| 0x00E2 | LATIN SMALL LETTER A WITH CIRCUMFLEX |
| 0x00E3 | LATIN SMALL LETTER A WITH TILDE |
| 0x00E4 | LATIN SMALL LETTER A WITH DIAERESIS |
| 0x00E5 | LATIN SMALL LETTER A WITH RING ABOVE |
| 0x00E6 | LATIN SMALL LETTER AE |
| 0x00E7 | LATIN SMALL LETTER C WITH CEDILLA |
| 0x00E8 | LATIN SMALL LETTER E WITH GRAVE |
| 0x00E9 | LATIN SMALL LETTER E WITH ACUTE |
| 0x00EA | LATIN SMALL LETTER E WITH CIRCUMFLEX |
| 0x00EB | LATIN SMALL LETTER E WITH DIAERESIS |
| 0x00EC | LATIN SMALL LETTER I WITH GRAVE |
| 0x00ED | LATIN SMALL LETTER I WITH ACUTE |
| 0x00EE | LATIN SMALL LETTER I WITH CIRCUMFLEX |
| 0x00EF | LATIN SMALL LETTER I WITH DIAERESIS |
| 0x00F0 | LATIN SMALL LETTER ETH |
| 0x00F1 | LATIN SMALL LETTER N WITH TILDE |
| 0x00F2 | LATIN SMALL LETTER O WITH GRAVE |
| 0x00F3 | LATIN SMALL LETTER O WITH ACUTE |
| 0x00F4 | LATIN SMALL LETTER O WITH CIRCUMFLEX |
| 0x00F5 | LATIN SMALL LETTER O WITH TILDE |
| 0x00F6 | LATIN SMALL LETTER O WITH DIAERESIS |
| 0x00F7 | DIVISION SIGN |
| 0x00F8 | LATIN SMALL LETTER O WITH STROKE |
| 0x00F9 | LATIN SMALL LETTER U WITH GRAVE |
| 0x00FA | LATIN SMALL LETTER U WITH ACUTE |

| Code | ISO/IEC 10646-1 [15] Character name |
|---|---|
| 0x00FB | LATIN SMALL LETTER U WITH CIRCUMFLEX |
| 0x00FC | LATIN SMALL LETTER U WITH DIAERESIS |
| 0x00FD | LATIN SMALL LETTER Y WITH ACUTE |
| 0x00FE | LATIN SMALL LETTER THORN |
| 0x00FF | LATIN SMALL LETTER Y WITH DIAERESIS |
| 0x0100 | LATIN CAPITAL LETTER A WITH MACRON |
| 0x0101 | LATIN SMALL LETTER A WITH MACRON |
| 0x0102 | LATIN CAPITAL LETTER A WITH BREVE |
| 0x0103 | LATIN SMALL LETTER A WITH BREVE |
| 0x0104 | LATIN CAPITAL LETTER A WITH OGONEK |
| 0x0105 | LATIN SMALL LETTER A WITH OGONEK |
| 0x0106 | LATIN CAPITAL LETTER C WITH ACUTE |
| 0x0107 | LATIN SMALL LETTER C WITH ACUTE |
| 0x0108 | LATIN CAPITAL LETTER C WITH CIRCUMFLEX |
| 0x0109 | LATIN SMALL LETTER C WITH CIRCUMFLEX |
| 0x010A | LATIN CAPITAL LETTER C WITH DOT ABOVE |
| 0x010B | LATIN SMALL LETTER C WITH DOT ABOVE |
| 0x010C | LATIN CAPITAL LETTER C WITH CARON |
| 0x010D | LATIN SMALL LETTER C WITH CARON |
| 0x010E | LATIN CAPITAL LETTER D WITH CARON |
| 0x010F | LATIN SMALL LETTER D WITH CARON |
| 0x0110 | LATIN CAPITAL LETTER D WITH STROKE |
| 0x0111 | LATIN SMALL LETTER D WITH STROKE |
| 0x0112 | LATIN CAPITAL LETTER E WITH MACRON |
| 0x0113 | LATIN SMALL LETTER E WITH MACRON |
| 0x0116 | LATIN CAPITAL LETTER E WITH DOT ABOVE |
| 0x0117 | LATIN SMALL LETTER E WITH DOT ABOVE |
| 0x0118 | LATIN CAPITAL LETTER E WITH OGONEK |
| 0x0119 | LATIN SMALL LETTER E WITH OGONEK |
| 0x011A | LATIN CAPITAL LETTER E WITH CARON |
| 0x011B | LATIN SMALL LETTER E WITH CARON |
| 0x011C | LATIN CAPITAL LETTER G WITH CIRCUMFLEX |
| 0x011D | LATIN SMALL LETTER G WITH CIRCUMFLEX |
| 0x011E | LATIN CAPITAL LETTER G WITH BREVE |
| 0x011F | LATIN SMALL LETTER G WITH BREVE |
| 0x0120 | LATIN CAPITAL LETTER G WITH DOT ABOVE |
| 0x0121 | LATIN SMALL LETTER G WITH DOT ABOVE |
| 0x0122 | LATIN CAPITAL LETTER G WITH CEDILLA |
| 0x0123 | LATIN SMALL LETTER G WITH CEDILLA |
| 0x0124 | LATIN CAPITAL LETTER H WITH CIRCUMFLEX |
| 0x0125 | LATIN SMALL LETTER H WITH CIRCUMFLEX |
| 0x0126 | LATIN CAPITAL LETTER H WITH STROKE |
| 0x0127 | LATIN SMALL LETTER H WITH STROKE |
| 0x0128 | LATIN CAPITAL LETTER I WITH TILDE |
| 0x0129 | LATIN SMALL LETTER I WITH TILDE |
| 0x012A | LATIN CAPITAL LETTER I WITH MACRON |
| 0x012B | LATIN SMALL LETTER I WITH MACRON |
| 0x012E | LATIN CAPITAL LETTER I WITH OGONEK |
| 0x012F | LATIN SMALL LETTER I WITH OGONEK |
| 0x0130 | LATIN CAPITAL LETTER I WITH DOT ABOVE |
| 0x0131 | LATIN SMALL LETTER DOTLESS I |
| 0x0132 | LATIN CAPITAL LIGATURE IJ |
| 0x0133 | LATIN SMALL LIGATURE IJ |
| 0x0134 | LATIN CAPITAL LETTER J WITH CIRCUMFLEX |
| 0x0135 | LATIN SMALL LETTER J WITH CIRCUMFLEX |
| 0x0136 | LATIN CAPITAL LETTER K WITH CEDILLA |
| 0x0137 | LATIN SMALL LETTER K WITH CEDILLA |
| 0x0138 | LATIN SMALL LETTER KRA |
| 0x0139 | LATIN CAPITAL LETTER L WITH ACUTE |
| 0x013A | LATIN SMALL LETTER L WITH ACUTE |
| 0x013B | LATIN CAPITAL LETTER L WITH CEDILLA |
| 0x013C | LATIN SMALL LETTER L WITH CEDILLA |
| 0x013D | LATIN CAPITAL LETTER L WITH CARON |
| 0x013E | LATIN SMALL LETTER L WITH CARON |

| Code | ISO/IEC 10646-1 [15] Character name |
|---|---|
| 0x013F | LATIN CAPITAL LETTER L WITH MIDDLE DOT |
| 0x0140 | LATIN SMALL LETTER L WITH MIDDLE DOT |
| 0x0141 | LATIN CAPITAL LETTER L WITH STROKE |
| 0x0142 | LATIN SMALL LETTER L WITH STROKE |
| 0x0143 | LATIN CAPITAL LETTER N WITH ACUTE |
| 0x0144 | LATIN SMALL LETTER N WITH ACUTE |
| 0x0145 | LATIN CAPITAL LETTER N WITH CEDILLA |
| 0x0146 | LATIN SMALL LETTER N WITH CEDILLA |
| 0x0147 | LATIN CAPITAL LETTER N WITH CARON |
| 0x0148 | LATIN SMALL LETTER N WITH CARON |
| 0x014A | LATIN CAPITAL LETTER ENG |
| 0x014B | LATIN SMALL LETTER ENG |
| 0x014C | LATIN CAPITAL LETTER O WITH MACRON |
| 0x014D | LATIN SMALL LETTER O WITH MACRON |
| 0x0152 | LATIN CAPITAL LIGATURE OE |
| 0x0153 | LATIN SMALL LIGATURE OE |
| 0x0154 | LATIN CAPITAL LETTER R WITH ACUTE |
| 0x0155 | LATIN SMALL LETTER R WITH ACUTE |
| 0x0156 | LATIN CAPITAL LETTER R WITH CEDILLA |
| 0x0157 | LATIN SMALL LETTER R WITH CEDILLA |
| 0x0158 | LATIN CAPITAL LETTER R WITH CARON |
| 0x0159 | LATIN SMALL LETTER R WITH CARON |
| 0x015A | LATIN CAPITAL LETTER S WITH ACUTE |
| 0x015B | LATIN SMALL LETTER S WITH ACUTE |
| 0x015C | LATIN CAPITAL LETTER S WITH CIRCUMFLEX |
| 0x015D | LATIN SMALL LETTER S WITH CIRCUMFLEX |
| 0x015E | LATIN CAPITAL LETTER S WITH CEDILLA |
| 0x015F | LATIN SMALL LETTER S WITH CEDILLA |
| 0x0160 | LATIN CAPITAL LETTER S WITH CARON |
| 0x0161 | LATIN SMALL LETTER S WITH CARON |
| 0x0162 | LATIN CAPITAL LETTER T WITH CEDILLA |
| 0x0163 | LATIN SMALL LETTER T WITH CEDILLA |
| 0x0164 | LATIN CAPITAL LETTER T WITH CARON |
| 0x0165 | LATIN SMALL LETTER T WITH CARON |
| 0x0166 | LATIN CAPITAL LETTER T WITH STROKE |
| 0x0167 | LATIN SMALL LETTER T WITH STROKE |
| 0x0168 | LATIN CAPITAL LETTER U WITH TILDE |
| 0x0169 | LATIN SMALL LETTER U WITH TILDE |
| 0x016A | LATIN CAPITAL LETTER U WITH MACRON |
| 0x016B | LATIN SMALL LETTER U WITH MACRON |
| 0x016C | LATIN CAPITAL LETTER U WITH BREVE |
| 0x016D | LATIN SMALL LETTER U WITH BREVE |
| 0x016E | LATIN CAPITAL LETTER U WITH RING ABOVE |
| 0x016F | LATIN SMALL LETTER U WITH RING ABOVE |
| 0x0172 | LATIN CAPITAL LETTER U WITH OGONEK |
| 0x0173 | LATIN SMALL LETTER U WITH OGONEK |
| 0x0174 | LATIN CAPITAL LETTER W WITH CIRCUMFLEX |
| 0x0175 | LATIN SMALL LETTER W WITH CIRCUMFLEX |
| 0x0176 | LATIN CAPITAL LETTER Y WITH CIRCUMFLEX |
| 0x0177 | LATIN SMALL LETTER Y WITH CIRCUMFLEX |
| 0x0178 | LATIN CAPITAL LETTER Y WITH DIAERESIS |
| 0x0179 | LATIN CAPITAL LETTER Z WITH ACUTE |
| 0x017A | LATIN SMALL LETTER Z WITH ACUTE |
| 0x017B | LATIN CAPITAL LETTER Z WITH DOT ABOVE |
| 0x017C | LATIN SMALL LETTER Z WITH DOT ABOVE |
| 0x017D | LATIN CAPITAL LETTER Z WITH CARON |
| 0x017E | LATIN SMALL LETTER Z WITH CARON |
| 0x01CD | LATIN CAPITAL LETTER A WITH CARON |
| 0x01CE | LATIN SMALL LETTER A WITH CARON |
| 0x02C6 | MODIFIER LETTER CIRCUMFLEX ACCENT |
| 0x02C7 | CARON (Mandarin Chinese third tone) |
| 0x02C9 | MODIFIER LETTER MACRON (Mandarin Chinese first tone) |
| 0x02D8 | BREVE |
| 0x02D9 | DOT ABOVE (Mandarin Chinese light tone) |

| Code | ISO/IEC 10646-1 [15] Character name |
|---|---|
| 0x02DA | RING ABOVE |
| 0x02DB | OGONEK |
| 0x02DC | SMALL TILDE |
| 0x066B | ARABIC DECIMAL SEPARATOR |
| 0x1E80 | LATIN CAPITAL LETTER W WITH GRAVE |
| 0x1E81 | LATIN SMALL LETTER W WITH GRAVE |
| 0x1E82 | LATIN CAPITAL LETTER W WITH ACUTE |
| 0x1E83 | LATIN SMALL LETTER W WITH ACUTE |
| 0x1E84 | LATIN CAPITAL LETTER W WITH DIAERESIS |
| 0x1E85 | LATIN SMALL LETTER W WITH DIAERESIS |
| 0x1EF2 | LATIN CAPITAL LETTER Y WITH GRAVE |
| 0x1EF3 | LATIN SMALL LETTER Y WITH GRAVE |
| 0x2007 | FIGURE SPACE |
| 0x2013 | EN DASH |
| 0x2014 | EM DASH |
| 0x2018 | LEFT SINGLE QUOTATION MARK |
| 0x2019 | RIGHT SINGLE QUOTATION MARK |
| 0x201A | SINGLE LOW-9 QUOTATION MARK |
| 0x201C | LEFT DOUBLE QUOTATION MARK |
| 0x201D | RIGHT DOUBLE QUOTATION MARK |
| 0x201E | DOUBLE LOW-9 QUOTATION MARK |
| 0x2022 | BULLET |
| 0x2026 | HORIZONTAL ELLIPSIS |
| 0x2030 | PER MILLE SIGN |
| 0x2039 | SINGLE LEFT-POINTING ANGLE QUOTATION MARK |
| 0x203A | SINGLE RIGHT-POINTING ANGLE QUOTATION MARK |
| 0x2044 | FRACTION SLASH |
| 0x20AC | EURO SIGN |
| 0x2122 | TRADE MARK SIGN |
| 0x2190 | LEFTWARDS ARROW |
| 0x2191 | UPWARDS ARROW |
| 0x2192 | RIGHTWARDS ARROW |
| 0x2193 | DOWNWARDS ARROW |
| 0x2212 | MINUS SIGN |
| 0x2214 | DOT PLUS |
| 0x2215 | DIVISION SLASH |
| 0x221E | INFINITY |
| 0x266B | BEAMED EIGHTH NOTES |
| 0x2713 | CHECK MARK |
| 0x2717 | BALLOT X |

# Annex F (informative): Authoring and implementation guidelines

## F.1 Authoring Guidelines

- Authoring guidelines are needed to specify those methods, classes and interfaces which are intended for use by implementations of JMF players. These methods, classes and interfaces are not intended for use by applications except for this purpose and that should be made clear.
- Authoring guidelines are needed to make it clear that it is optional for JMF controls to have an associated java.awt component. This is in the JMF documentation but the phrasing implies this an exceptional case. In many GEM receivers, the presence of such a component would be the exceptional case.To be completed.

## F.2 Implementation Guidelines

To be completed.

## F.3 Authoring guidelines for DVB-J

To be completed.

# Annex G (normative): Minimum platform capabilities

## G.1 Graphics

In the area of graphics capability the following requirements are made on GEM terminals.

### G.1.1 Device capabilities

#### G.1.1.1 General

- The number of DVB-J applications concurrently owning instances of `HScene` is not limited except by underlying platform resources like the total memory of the GEM terminal. The GEM terminal is required to support either the "Platforms Supporting a Restricted Multi-Window System" or the "Platforms Supporting a Full Multi-Window System" implementation scenarios as defined in `org.havi.ui.HSceneFactory`. The implementation scenario "Platforms Supporting a Single Window System" is not a valid choice for GEM.
- The GEM terminal is not required to provide a mechanism for the end-user to change user input focus between `HScenes`. Hence GEM applications wishing to receive user input focus must explicitly request it.
- The GEM terminal shall implement at least one `HGraphicsDevice` which shall be full screen.
- The GEM terminal shall implement at least one `HBackgroundDevice`. These are always full screen.
- The GEM terminal shall implement at least one `HVideoDevice` which is always capable of being configured to be full screen.
- The GEM terminal shall implement at least one `HScreen` which shall support at least one video, graphics and background device as defined immediately above.

#### G.1.1.2 Standard definition

The following tables define HScreenConfigurations and sets of coherent HScreenConfigurations for GEM terminals not supporting high definition video.

GEM terminals capable of 4:3 output and 16:9 output (such as set top boxes but also IDTVs capable of a "reduced scan") shall, regardless of the aspect ratio of the attached display (to the extent this is known), support all mandatory configurations and sets of coherent screen configurations in the appropriate table. GEM terminals capable of outputting a signal in only one aspect ratio shall support all mandatory configurations and sets of coherent configurations for that screen aspect ratio.

**Table 82: Sets of coherent configurations for standard definition - 625-line markets**

| Set | Screen Aspect Ratio | HGraphics Configuration | | HVideo Configuration | | HBackground Configuration | | Notes |
|---|---|---|---|---|---|---|---|---|
| | | Resolution | Pixel aspect ratio | Resolution | Pixel aspect ratio | Resolution | Pixel aspect ratio | |
| 1,2 | 4:3 | 720x576 | 56/45 | 720x576 | 16/15 | 720x576 | 16/15 | Default for 4:3 output |
| 3,4 | 4:3 | 720x576 | 16/15 | 720x576 | 16/15 | 720x576 | 16/15 | |
| *5,6* | *4:3* | *768x576* | *1/1* | 720x576 | 16/15 | 720x576 | 16/15 | |
| 7,8 | 16:9 | 720x576 | 56/45 | 720x576 | 64/45 | 720x576 | 64/45 | Default for 16:9 output |
| 9,10 | 16:9 | 720x576 | 64/45 | 720x576 | 64/45 | 720x576 | 64/45 | |
| *11,12* | *16:9* | *1024x576* | *1/1* | 720x576 | 64/45 | 720x576 | 64/45 | |

**Table 83: Sets of coherent configurations for standard definition - 525-line markets**

| Set | Screen Aspect Ratio | HGraphics Configuration | | HVideo Configuration | | HBackground Configuration | | Notes |
|---|---|---|---|---|---|---|---|---|
| | | Resolution | Pixel aspect ratio | Resolution | Pixel aspect ratio | Resolution | Pixel aspect ratio | |
| 1,2 | 4:3 | 720x480 | 47/45 | 720x480 | 8/9 | 720x480 | 8/9 | Default for 4:3 output |
| 3,4 | 4:3 | 720x480 | 8/9 | 720x480 | 8/9 | 720x480 | 8/9 | |
| *5,6* | *4:3* | *640x480* | *1/1* | 720x480 | 8/9 | 720x480 | 8/9 | |
| 7,8 | 16:9 | 720x480 | 47/45 | 720x480 | 32/27 | 720x480 | 32/27 | Default for 16:9 output |
| 9,10 | 16:9 | 720x480 | 32/27 | 720x480 | 32/27 | 720x480 | 32/27 | |
| *11,12* | *16:9* | *854x480* | *1/1* | 720x480 | 32/27 | 720x480 | 32/27 | |

The following additional requirements shall apply concerning both the above tables:

- HScreenConfigurations and sets of coherent screen configurations shown in *italics* are always optional.
- Each row shall correspond to two sets of coherent configurations since for each possible combination of a HGraphicsConfiguration and a HVideoConfiguration, 2 HBackgroundConfigurations shall be supported - one a HStillImageBackgroundConfiguration (supporting the display of MPEG stills) and one just a HBackgroundConfiguration only supporting a single colour.
- All configurations in the table shall exactly cover the full area from (0,0) to (1,1).

NOTE 1: For video, whether the video picture covers the entire HVideoDevice depends on factors including scaling, positioning and decoder format conversion.

- Where a GEM application has the HGraphicsDevice reserved, the receiver's picture shape and/or wide screen signalling shall be set to the output aspect ratio of the HGraphicsDevice. Where the HGraphicsDevice is not reserved by a GEM application, the picture shape and/or wide screen signalling shall be set according to the configuration of the HVideoDevice and the decoder format conversion being applied to the video.
- Each required HVideoConfiguration shall have its pixels aligned (as defined by HScreenConfigTemplate.VIDEO_GRAPHICS_PIXEL_ALIGNED) with all required HGraphicsConfigurations and vice-versa.
- HGraphicsConfigurations whose pixel aspect ratio is 56/45 are HEmulatedGraphicsConfigurations corresponding to the pixel aspect ratio of a logical 14:9 screen as defined in clause 13.3.7 "14:9 Aspect Ratio Support".

NOTE 2: Although default configurations are shown, application developers should note that another application may already have changed the graphics configuration from the default before they start running or may change it while they are running.

### G.1.1.3 High definition

GEM terminals supporting HD shall support at least 2 HGraphicsDevices as follows;

- The one in front shall have 4 possible HGraphicsConfigurations:
  - full screen 1 280 x 720;
  - full screen 960 x 540;
  - full screen standard definition (16:9);
  - full screen standard definition (14:9 - HEmulatedGraphicsConfiguration).

- The one behind shall have 3 possible HGraphicsConfigurations:
  - full screen 1 280 x 720;
  - full screen standard definition (16:9);
  - full screen standard definition (14:9 - HEmulatedGraphicsConfiguration).

GEM terminals shall support the simultaneous display of full screen 1 280 x 720 and full screen standard definition graphics. GEM does not define how this is supported. Some possibilities include actual hardware graphics planes, hardware assisted composition ("blitting") and rendered into a single actual hardware graphics plane (e.g. by applying a transformation in the graphics library). Where a single real hardware graphics plane is used for both these resolutions, it shall have a resolution of at least 1 280 x 720. In this case, it is an allowed implementation option that applications in the rear HGraphicsDevice are clipped by all applications in front of them, both those in the same HGraphicsDevice area and those in the front HGraphicsDevice.

The resolution of 960 x 540 shall always be supported with an actual hardware graphics plane whose resolution is at least 960 x 540. GEM permits (but does not require) that the front HGraphicsDevice can only be configured to 960 x 540 when the rear HGraphicsDevice is disabled as defined by HScreenConfigTemplate.DISABLED. Applications which want to use 960 x 540 should be careful to ask for that resolution using HScreen.getCoherentConfigurations without putting requirements on the configuration of the rear graphics plane.

In the absence of GEM application control, the default shall be full screen 1 280 x 720 graphics in front of full screen SD graphics.

NOTE: Unbound applications running on GEM terminals supporting HD should ensure that a resolution of 1 280 x 720 is always available except when it has been arranged that no applications require it. Conversely service bound GEM applications wishing to obtain a resolution of 960 x 540 on GEM terminals supporting unbound applications may need to co-ordinate this with the owner of the unbound applications.

When HD video is being decoded, the resolution of the HVideoDevice is not specified by GEM. Some possible resolutions include the following;

- The resolution of the video being decoded. Typically used if the HD display does the conversion between the transmitted video resolution and the display resolution.
- The resolution of the HD display - assuming the GEM terminal has this information. Typically used for (old) displays which only support one resolution on their input.
- Some other (neutral) resolution.

In any case, it is expected that there is a single HVideoConfiguration for HD video decoding Hence it is meaningless for GEM applications to reserve or set the HVideoConfiguration when HD video is being decoded.

When HD video is being decoded, clause X.1.6 "MPEG I frame and Video drips" of the present document defines that support for MPEG I-frames (and hence HStillImageBackgroundConfigurations) is optional. Hence HBackgroundDevices shall support at least the following HBackgroundConfigurations;

- An HStillImageBackgroundConfiguration which is required to be available except when HD video is being decoded (in which case it is optional).
- A full screen HBackgroundConfiguration which is not an HStillImageBackgroundConfiguration and which is valid at all times.

The following tables define mandatory and optional HScreenConfigurations and sets of coherent screen configurations for high definition markets. Optional HScreenConfigurations and sets of coherent screen configurations are shown in *italics*.

**Table 84: Sets of coherent configurations for high definition**

| Set Number | Video Being Decoded | Front HGraphics Configuration | Rear HGraphics Configuration | HVideo Configuration | HBackground Configuration | Notes |
|---|---|---|---|---|---|---|
| 1,2 | SD | 1280x720 | 16:9 SD | 16:9 SD | 16:9 SD, one configuration capable of showing a still and one not. | |
| 3,4 | SD | | 14:9 SD | | | default for SD |
| 5,6 | SD | 16:9 SD | 1280x720 | | | |
| 7,8 | SD | 14:9 SD | | | | |
| 9,10 | SD | 960x540 | Not specified in present document | | | |
| *11,12* | *SD* | *1920x1080* | | | | |
| 13 | HD | 1280x720 | 14:9 SD | HD video | Not specified in present document | default for HD |
| 14 | HD | 1280x720 | 16:9 SD | HD video | | |
| 15 | HD | 16:9 SD | 1280x720 | HD video | | |
| 16 | HD | 14:9 SD | 1280x720 | HD video | | |
| 17 | HD | 960x540 | Not specified in present document | HD video | | |
| *18* | *HD* | *1920x1080* | | HD video | | |

The following additional requirements shall apply concerning table 84:

- "16:9 SD" shall mean the appropriate resolution with the terminal in standard definition mode.
- "14:9 SD" shall mean the appropriate resolution with the terminal in standard definition mode.
- Each of the rows shown as corresponding to two sets of coherent configurations means set where the HbackgroundConfiguration shall be a HStillImageBackgroundConfiguration (supporting the display of MPEG stills) and one set where it is just a HBackgroundConfiguration only supporting a single colour.
- "HD video" means the single HVideoConfiguration for HD video referred to earlier in this clause.
- HScreenConfigurations and sets of coherent screen configurations shown in *italics* are always optional.

## G.1.2 Video presentation capabilities

- GEM terminal specifications for packaged media targets require support for DFC_PROCESSING_FULL only.
- The following set of standard decoder format conversions shall be supported by all GEM terminals (except packaged media targets):
  - DFC_PROCESSING_CCO.
  - DFC_PROCESSING_FULL.
  - DFC_PROCESSING_LB_16_9.
  - DFC_PROCESSING_PAN_SCAN.
- The following modes are optional:
  - DFC_PROCESSING_LB_14_9.
  - DFC_PROCESSING_LB_2_21_1_ON_16_9.
  - DFC_PROCESSING_LB_2_21_1_ON_4_3.
- GEM terminals are required to support displaying MPEG video with horizontal scale factors of 2, 3/2, 1, 3/4, 1/2, 1/3, 1/4, and vertical scale factors of 2, 1, 1/2, 1/3, 1/4. At least half of the decoded image height must remain within the display raster. For horizontal scaling factors of full size and above, at least one quarter of the decoded image width must remain within the display raster. For horizontal scaling factors below full size, at least half the decoded image width must remain within the display raster. Use of the scale factor 0 for both vertical and horizontal scaling shall result in the video being discarded and the video plane being transparent to the plane behind it, as defined by the GEM graphics reference model. These requirements also apply to HD video on GEM terminals supporting that feature.

NOTE: GEM does not specify behaviour when either but not both of the vertical or the horizontal scaling factors are zero.

- Support for component-based JMF players is only required in GEM terminals that:

  a) Support the simultaneous decoding of more than one broadcast video stream (as defined in clause 7.2.2, "Video") on a single display (HScreen instance).

  b) Expose through the GEM-defined API, the ability to initiate and/or control such decoding.

  In these GEM terminals, all JMF players shall be able to be used as a component-based player when decoding video. In other GEM terminals, this support is optional.

## G.1.3 Image processing capabilities

- All DVBGraphics objects shall support SRC and CLEAR and SRC_OVER.
  When SRC_OVER is used with DVBGraphics objects with a sample model of the type TYPE_BASE a perfect result is only guaranteed to be produced with alpha values of 0 and 1. Alpha values other than 0 and 1 can be approximated. DVBGraphics object with a type of TYPE_ADVANCED will produce a result as expected but those SRC_OVER operations are likely to be slow.

- DVBGraphics object created from a DVBBufferedImage with the type TYPE_ADVANCED shall perform SRC_OVER operations without approximations of the compositing rule.

### G.1.3.1 Composition rules

GEM terminals are required to implement at least the SRC, SRC_OVER and CLEAR rules when composing graphics over video and graphics over graphics.

## G.1.4 Alpha capabilities

In the composition of the graphics (AWT/HAVi components) with background and video planes (see figure 23), GEM terminals are required to implement at least 16 levels of alpha evenly spaced between 0 (completely transparent) and 255 (opaque).

Allowed approximations for the composition of graphics over other graphics are defined in clause 13.6.1.1.2 "Graphics over other graphics".

## G.1.5 Colour capabilities

The colour model is a "true colour" one. There shall be at least 4 bits per pixel for each of R. G and B.

## G.1.6 MPEG I frame and Video drips

The minimum positioning and scaling capabilities defined above for MPEG video shall also apply to MPEG I frame and Video drips.

The GEM terminal shall support at least one `HStillImageBackgroundConfiguration`.

GEM terminals shall support simultaneously the display of already decoded MPEG I-frames and the decoding and display of video. It is implementation dependent as to whether the decoding of MPEG I-frames is supported simultaneously with the decoding of video.

The display of I-frames simultaneously with HD video is optional.

# G.2 Audio

No audio mixing is required.

Audio played from memory may pre-empt any audio from the transport stream. This may disturb decoding of any broadcast video stream.

Audio from memory shall be output in preference to audio from stream if overall audio output has not been disabled by the user. On platforms capable of mixing audio from memory with audio from the stream it shall do this if there is a stream playing. Where audio from the stream is interrupted, decoding of it shall automatically resume when audio from memory ceases if the stream concerned is still playing. This only applies where the audio from memory and the stream are both under the control of the same GEM application. Where multiple applications are involved see clause 9.4 "Inter application resource management".

# G.3 Video

The GEM terminal is only required to support decoding of a single video stream at a given time. The number of implemented video decoders will affect the functionality of the video and background devices.

GEM terminals shall be able to present un-synchronised broadcast video and audio where explicitly requested by GEM applications. See clause 11.4.2.5.5, "Clarifications".

# G.4 Resident fonts and text rendering

## G.4.1 The built-in font

At least the font Tiresias [63] shall be provided.

NOTE 1: Tiresias is the trade name of a product supplied by Bitstream and owned by the Royal National Institute of the Blind. This information is given for the convenience of users of GEM and does not constitute an endorsement by ETSI of the product named. Equivalent products may be used if they can be shown to lead to the same results.

NOTE 2: As described in clause 7.3, "Resident fonts" in the present document, the resident font might not be built into the terminal in all cases.

The built-in font shall be supported with the weight ("PLAIN"). The default size shall be 26 points.

The built-in font shall be rendered from an outline representation with a continuous range of point sizes from 12 to 96.

NOTE 3: Application developers should be aware that implementations may cache glyphs for recently used sizes. Using many different sizes simultaneously may degrade performance.

Table 85 gives example sizes for various use cases.

**Table 85: Example font sizes and use cases**

| Size (points) | TV lines over "Cap-V" (see notes 1 and 2) | Informative Name |
|---|---|---|
| 36 | 24 | Heading / Large subtitle |
| 31 | 21 | Subtitle |
| 26 | 18 | Body |
| 24 | 16 | Footnote |
| NOTE 1: The primary definition of the character size is the font size in points, the height of a capital letter "V" in TV lines is provided for information only.<br>NOTE 2: The second column of this table is informative and it may not apply to all GEM terminal specifications. For clarity GEM terminal specifications may specify an equivalent of these values, though if they do not this does not imply that the second column of the table is accurate for these specifications. | | |

## G.4.2 Presentation to DVB-J

The embedded font "Tiresias" shall have:

- The logical name "SansSerif" (for example returned by `java.awt.Toolkit.getFontList`).
- The family name "Tiresias" (for example returned by `java.awt.Font.getFamily`).
- The font face name "Tiresias PLAIN".

## G.4.3 Text directions

The `DVBTextLayoutManager` is only required to support the following configuration of text direction:

- LINE_ORIENTATION_HORIZONTAL and START_CORNER_UPPER_LEFT.

# G.5 Input events

Remote control devices should be designed such that the mechanism for generating each required event is obvious to typical end-users. Hence, application authors may assume that the required keys can be generated without recourse to non-obvious input gestures.

Some downloaded resident applications specified as extensions to GEM may perform some of the functions of the GEM navigator, e.g. the monitor application defined OCAP [5]. In this case, such downloaded software shall implement a policy that is consistent with the requirements of GEM.

GEM terminal specifications should not define additional input event codes where GEM defines an appropriate VK_ code.

**Table 86: Minimum set of input events**

| Input event |
|---|
| VK_0 to VK_9 |
| VK_UP |
| VK_DOWN |
| VK_LEFT |
| VK_RIGHT |
| VK_ENTER |
| VK_TELETEXT (optional) |
| VK_COLORED_KEY_0 |
| VK_COLORED_KEY_1 |
| VK_COLORED_KEY_2 |
| VK_COLORED_KEY_3 (optional) |

**Table 87: Additional input events required by privileged applications**

| Input event |
|---|
| VK_CHANNEL_DOWN |
| VK_CHANNEL_UP |
| VK_HOME |

These tables define the minimum set of input events which shall be available to the set of running GEM applications if they are interested in receiving them. For DVB-J applications, this is described in more detail in clause 11.4.1.4, "Television viewing mode".

NOTE 1: They are not guaranteed to be always available to any one GEM application because another application running at the same time may have one of these events exclusively reserved. The application with focus (if any) always receives all of these events unless another application within the same service has requested and been granted exclusive access to one or more events. The process for event distribution for DVB-J applications is described in more detail in annex J "DVB-J event API".

NOTE 2: The user input device for a GEM terminal may support more events than this however this is implementation dependent. If more events than this are supported, it is equally implementation dependent whether the additional events are sent to GEM applications or sent to the GEM navigator. Events which are always sent to the GEM navigator may not be visible at all to GEM applications. For example, a GEM receiver using a conventional remote control will probably have program up/program down keys which are only ever sent to the navigator and cause service selection when received there.

NOTE 3: The specification for `java.awt.event.keyEvent`, `org.havi.ui.event.HRcEvent` and `org.dvb.event.UserEvent` do not define which VK_ codes have "valid" Unicode characters. Hence DVB-J applications cannot rely on the return value of the `getKeyChar` method for KEY_PRESSED or KEY_RELEASED events. Applications which want to use Unicode characters should use KEY_TYPED events instead.

GEM terminals intending to provide a remote control key (or equivalent) labelled "back" (or equivalent) shall use VK_F9 as its code. GEM applications decoding MHEG-5 Broadcast Profile content shall use this key as the MHEG-5 "cancel key function". The present document does not require support for this key in all GEM terminals.

The `HRcCapabilities.getRepresentation(int)` method shall return an `HEventRepresentation` instance for each input event in the above table. Each such `HEventRepresentation` instance shall return at least one of ER_TYPE_COLOR, ER_TYPE_STRING or ER_TYPE_SYMBOL from calls to its `getType()` method. Where ER_TYPE_SYMBOL is returned, the corresponding `java.awt.Image` instance shall have the size of 32 by 32. The representations supported shall match the user input device generating the input events listed above which is shipped with the GEM terminal.

NOTE 4: Application developers need to be aware that a small proportion of end-users will be using user input devices not shipped with the GEM terminal, e.g. replacements and/or programmable remote controls. There is no requirement for GEM terminals to detect this.

NOTE 5: This requires a meaningful representation of how a given supported keycode is generated.

The following mapping from key codes to remote control labelling is recommended:

- VK_COLORED_KEY_0 Red.
- VK_COLORED_KEY_1 Green.
- VK_COLORED_KEY_2 Yellow.
- VK_COLORED_KEY_3 Blue.

Additionally, it is recommended that these keys be in the order red, green yellow then blue.

# G.6 Memory

In order to be able to execute GEM conformance tests, the following minimum memory requirements are defined for GEM terminals. All of these are to be measured during normal usage and operational conditions of a GEM terminal. All are to be measured in the `initXlet` method of a DVB-J application which is both the only auto-start application signalled in a service and the only application running at that time.

- Enough memory to successfully load any arbitrary 262 144 (or less) Bytes of Java class files into the memory space of the Java virtual machine. Execution of code called as part of initializing fields in classes is excluded from consideration as part of "load"ing here. RAM usage by the bytecode verifier is included in consideration as part of "load"ing here.

Enough memory to do the above and individually each of the following:

- Enough memory to successfully create a Java byte array of lengths from 1 entry to 262 144 entries.
- Enough memory to successfully load a PNG image whose size is the same as the resolution of an `HGraphicsDevice` at standard definition, as discussed in clause G.1.
- Enough memory to successfully load from file and play from memory 5 seconds of audio at 128 kbit/s (where kbit/s is as used in TS 101 154 [2]). It shall be measured using files that do not include any optional extension fields.
- Enough memory to successfully allocate an array of `java.lang.Object` of length 16 384 and fill each element of this array with a distinct instance of `java.lang.Object`.

The memory requirements detailed in this clause are not exhaustive. For example, the specific requirement concerning an array of type byte in no way implies that GEM terminals are exempt from requirements found elsewhere in the present document (including normatively referenced specifications) for supporting arrays of other types.

NOTE: Additional detail may be added to these requirements in order to properly enable the GEM conformance tests.

# G.7 Other resources

**Table 88: Minimum requirements for other resources**

| Feature | Specification |
|---|---|
| gamma correction in the receiver | none |
| HAVi mattes | Platforms are not required to implement the functionality of mattes in HAVi. Non-implementation should be implemented as specified by HAVi. |
| Overlapping applications | GEM terminals are not required to support overlapping top level UI containers (e.g. HScenes where DVB-J applications are concerned). |
| AIT section filtering | The implementation is not required to dedicate more than one section filter to monitoring the AIT.<br>NOTE 1: This entry applies only to GEM terminal specifications that include the functional equivalent named "Application Signalling" as defined in clause 15.6, "Functional equivalents". However, attention is drawn to the requirement on detecting Application Description changes in clause 10.4.1. |
| Key lengths for TLS | Receivers shall support certificate key lengths up to and including 2 048 bits for TLS (see clause 12.10, "Security on the return channel").<br>NOTE 2: The key lengths for application authentication apply only to codesigning using the MHP model. |
| Key lengths for Application Authentication | Receivers shall support certificate key lengths up to and including 2 048 bits for application authentication (see clause 12.2, "Authentication of applications"). |
| Internet clients | WWW and email clients required, usenet news client optional. |
| Non-CA Smart Card Reader | GEM terminals are not required to include this |
| Stored services | No requirement for persistent storage for stored services even if other forms of persistent storage are supported. |
| PSTN/ISDN modem (where present) | Where such a modem is present, the GEM terminal should provide a mechanism to configure a dialing prefix, e.g. '0', to be dialled in front of the number specified by the GEM application. Where such a prefix is supported, this shall be invisible to GEM applications, for example, it shall not need to be included in entries in the permission request file. |

NOTE: The values in the table below are set for the purposes of conformance testing and should not be used by application or GEM terminal developers as being indicative of the capabilities of commercial products.

**Table 89: Minimum requirements for other resources for conformance purposes**

| Feature | Specification |
|---|---|
| Application accessible timers (see note 2) | At least 4 timers for each ServiceContext which can be presenting GEM applications at the same time. (i.e. shared between the applications signalled as part of the same service) (see note 1). |
| MPEG-2 transport stream network interface | Shall support at least one network interface enabling reception of an MPEG-2 transport stream and selection of that transport stream from among those available to be received. This shall support those broadcast channel protocols supported by the GEM terminal (see clause 6.2, "Broadcast channel protocols") and the GEM APIs defined to interface to these protocols and to control these interfaces. |
| Bidirectional IP network interface | GEM terminals supporting the interactive broadcast profile shall support at least one network interface for bidirectional IP traffic. This shall support those interaction channel protocols supported by the GEM terminal (see clause 6.3, "Interaction channel protocols") and the GEM APIs defined to interface to these protocols and to control these interfaces. |
| Conditional access | None required.<br>The absence or optional presence of one shall be correctly reported through the Conditional access API.<br>Local regulation may require more support than this minimum.<br>See note 4. |
| Persistent storage | At least 131 072 bytes. This shall not include the requirements of clause 12.12, "Platform minima" for persistent storage of CRLs and RCMMs. |
| Application accessible MPEG-2 section filters (see note 2) | At least 2 shared among all applications signalled as part of the same service (see note 1). |
| Application accessible DVB-J threads (see note 2) | At least 4 shared among all applications signalled as part of the same service. Threads created by the platform and used to call methods of the application are excluded from this number (see note 1). |
| Applications in a service | GEM terminals shall not impose any arbitrary limit on the number of applications which they can support at the same time. |
| Memory for stored services as defined in clause 9.6.2, "Stored services" | Either 0 or greater than or equal to 65 536 bytes. This memory shall be non-volatile. If greater than zero, there shall be a means to empty this memory, at least for the purposes of running conformance tests (see note 3). |
| NOTE 1: These requirements apply to one set of GEM applications signalled as part of the same service. If an GEM terminal supports simultaneous execution of more than one set of signalled applications then it shall make available at least these minimum resources for each set of signalled applications which can be executed simultaneously.<br>NOTE 2: "Application accessible" means guaranteed to be accessible to GEM applications through the API defined in the present document for the feature concerned. This must be regardless of the extent of any usage of that feature or function as part of the GEM terminal implementation.<br>NOTE 3: Implementors should choose an actual size based on the sizes of GEM applications found in their market and not based on this largely arbitrary number.<br>NOTE 4: These requirements apply only to GEM terminal specifications that support the functional equivalent named "Conditional Access". | |

# Annex H (normative): Extensions

Private protocols and possibly APIs are not precluded and are outside of the scope of the GEM specification.

The addition of public or protected constructors, methods or fields to classes and interfaces in the `org.dvb` namespace is not allowed.

Restrictions on proprietary extensions to the `org.havi.ui` packages are found in section 7.2.2 ("Profile #1: DCMs and Application Modules") of HAVi [50].

# Annex I (normative): DVB-J fundamental classes

# Package

# org.dvb.lang

### Description

Provides those core platform related features not found in the java.lang package.

| Class Summary | |
|---|---|
| Classes<br>DVBClassLoader | This class loader is used to load classes and resources from a search path of URLs referring to locations where Java class files may be stored. |

# org.dvb.lang

# DVBClassLoader

### Declaration

```
public abstract class DVBClassLoader extends java.security.SecureClassLoader
```

```
java.lang.Object
  |
  +--java.lang.ClassLoader
        |
        +--java.security.SecureClassLoader
              |
              +--org.dvb.lang.DVBClassLoader
```

### Description

This class loader is used to load classes and resources from a search path of URLs referring to locations where Java class files may be stored.

The classes that are loaded are by default only allowed to load code through the parent classloader, or from the URLs specified when the DVBClassLoader was created.

### Constructors

### DVBClassLoader(URL[])

```
public DVBClassLoader(java.net.URL[] urls)
```

Constructs a new DVBClassLoader for the given URLs. The URLs will be searched in the order specified for classes and resources.

If there is a security manager, this method first calls the security manager's `checkCreateClassLoader` method to ensure creation of a class loader is allowed.

### Parameters:

`urls` - the URLs from which to load classes and resources.

Throws:

`java.lang.SecurityException` - if a security manager exists and its `checkCreateClassLoader` method does not allow creation of a class loader.

See Also:

`java.lang.SecurityManager.checkCreateClassLoader()`

### DVBClassLoader(URL[], ClassLoader)

```
public DVBClassLoader(java.net.URL[] urls, java.lang.ClassLoader parent)
```

Constructs a new DVBClassLoader for the given URLs. The URLs will be searched in the order specified for classes and resources.

If there is a security manager, this method first calls the security manager's `checkCreateClassLoader` method to ensure creation of a class loader is allowed.

Parameters:

`urls` - the URLs from which to load classes and resources

`parent` - the parent classloader for delegation

Throws:

`java.lang.SecurityException` - if a security manager exists and its `checkCreateClassLoader` method does not allow creation of a class loader.

See Also:

`java.lang.SecurityManager.checkCreateClassLoader()`

## Methods

### findClass(String)

```
protected java.lang.Class findClass(java.lang.String name)
    throws ClassNotFoundException
```

Finds and loads the class with the specified name from the URL search path. Any URLs are searched until the class is found.

Overrides:

`findClass` in class `ClassLoader`

Parameters:

`name` - the name of the class.

Returns:

the resulting class.

Throws:

`java.lang.ClassNotFoundException` - if the named class could not be found.

### newInstance(URL[])

```
public static org.dvb.lang.DVBClassLoader newInstance(java.net.URL[] urls)
```

Creates a new instance of DVBClassLoader for the specified URLs. If a security manager is installed, the `loadClass` method of the DVBClassLoader returned by this method will invoke the `SecurityManager.checkPackageAccess` method before loading the class.

Parameters:

urls - the URLs to search for classes and resources.

Returns:

the resulting class loader

### newInstance(URL[], ClassLoader)

```
public static org.dvb.lang.DVBClassLoader newInstance(java.net.URL[] urls,
java.lang.ClassLoader parent)
```

Creates a new instance of DVBClassLoader for the specified URLs. If a security manager is installed, the `loadClass` method of the DVBClassLoader returned by this method will invoke the `SecurityManager.checkPackageAccess` method before loading the class.

Parameters:

urls - the URLs to search for classes and resources.

parent - the parent class loader for delegation.

Returns:

the resulting class loader.

# Annex J (normative): DVB-J event API

Applications can use the `org.dvb.event` API, to receive events without being focused and/or to have exclusive access to events.

NOTE: The return value of `org.dvb.event.UserEvent.getKeyChar()` is defined in terms of `CHAR_UNDEFINED`. The value of `getKeyChar()` for this method is not defined in an interoperable way. In no case should an interoperable application rely on the return value of `getKeyChar()`; further, the value returned for an undefined key is not defined interoperably and may vary across platform implementations.

## J.1 Overview

This API provides a mechanism allowing GEM applications to influence the routing of events to either GEM applications or the navigator. This typically would be used in a mode where the receiver is primarily used for TV viewing, allowing some events to be received by the application, and allowing the navigator to receive those events that are not requested by any GEM application. This API enables GEM applications to choose between the following mechanisms for receiving events:

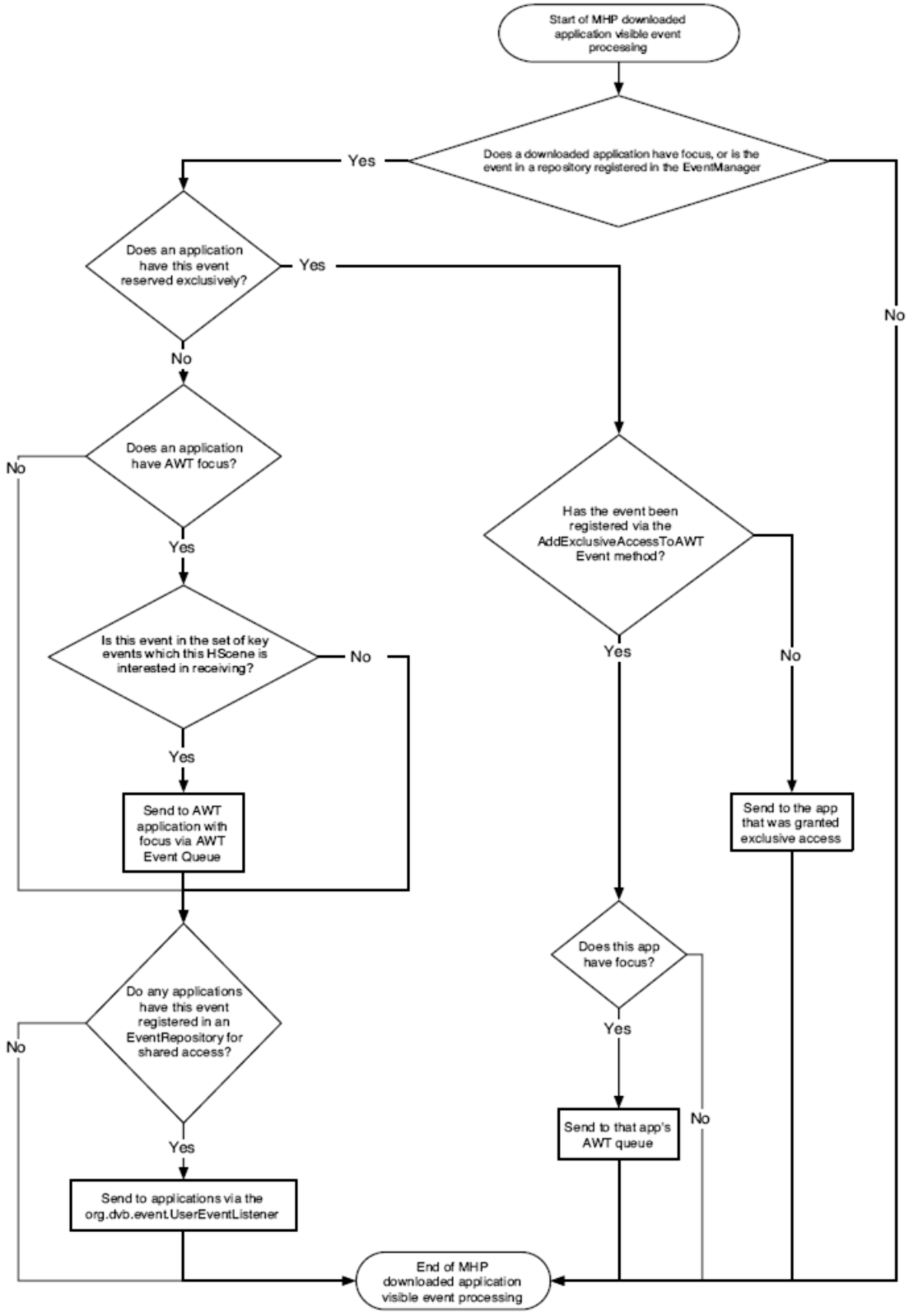
- through the standard `java.awt` mechanism;
- through the standard `java.awt` mechanism, but for some events to be exclusively accessed by the application;
- through a mechanism defined by this API;
- through the mechanism defined by this API, but for some events to be exclusively accessed by the application.

The last two solutions could be used by non-graphical applications in order to receive events that are coming from the user. It could also be used by an invisible application if it wants to be presented when a specific key is pressed.

If an application wants to have exclusive access to some events and to manage them through the java.awt then it must use this API so that it can be aware of the fact that it has lost or gained access to these events. One must notice that an application based on awt event mechanism will receive events only if it is focused.

If an application has acquired exclusive access to receive an event through either of these mechanisms, it will not receive this same event through the other mechanism. An application can obtain exclusive access to an event through only one of the two mechanisms at a time.

The diagram below shows how the GEM terminal decides on the processing of an event (defined by a set of keychar/keycode, event id and modifiers). This includes deciding on the mechanisms by which the event will be dispatched and the appropriate Java class to encapsulate the event for each of these mechanisms.




NOTE: In this figure the terms "AWT queue" and "AWT Event Queue" are intended to apply to both events delivered directly through `java.awt` and to events delivered in HAVi event classes. Events delivered in HAVi event classes include `HFocusEvents` with a keycode as the `transfer-id` as well as `HKeyEvents`. Hence if a keycode is exclusively reserved by a `UserEventListener`, or by another application through `EventManager.addExclusiveAccessToAWTEvent()`, this does block `HFocusEvents` being generated if that keycode would be the transfer-id of the `HFocusEvent`.

**Figure 47: The GEM downloaded application visible event distribution mechanism**

# J.2 The resource management

An application asking for exclusive access to some events will use the resource framework defined in DAVIC 1.4.1p9 [17] so that it can be aware of the fact that it has lost access to the user events it asked for (see the example below).

# J.3 The Event Repository

The UserEventRepository is the class that is used by the application to define the user events it intends to use. For the moment user events are just key events but it is a place-holder for new families of events (voice command for example). If an application asks for an exclusive access to events by means of a repository, this exclusive access will be lost at the time when one of the event is grabbed by another application. User events that can be accessed by an application are defined in the UserEvent class.

## J.3.1 Example

```
import org.davic.resources.ResourceClient.* ;
   import org.dvb.event.* ;
   class Example implements UserEventListener, ResourceStatusListener, ResourceClient {
      private int myStatus ;
      public Example () {
         EventManager em ;
         UserEventRepository repository ;
         em = EventManager.getInstance () ;
         repository = new UserEventRepository ("R1") ;

         repository.addKey (UserEvent.VK_ENTER) ;
         em.addUserListener ((UserEventListener)this, (ResourceClient)this, repository) ;
         em.addResourceStatusEventListener (this) ;
      }
      /**
      * methods defined by the UserEventListener interface.
      */
      public void UserEventReceived (UserEvent e) {
      }
      /**
      * Methods defined by the ResourceClient interface.
      */
      /**
      * In the case a cooperative application asks for an user event
      * exclusively used by me.
      */
      public boolean requestRelease(ResourceProxy proxy, Object requestData) {
         String name ;
         // let's retrieve the name of the repository, that I have created, and
         // which contains the user event that the other application asks for.
         name = (RepositoryDescriptor)proxy.getName () ;
         if ((name.compareTo ("R1") == 0) & (myStatus ==  )) {
            // Ok I release this event.
            return true ;
         } else {
            // No I need this event, sorry !
            return false ;
         }
      }
      public void release (ResourceProxy proxy) {

      }
      public void notifyRelease (ResourceProxy proxy) {

      }
      public void statusChanged (ResourceStatusEvent event) {

      }
   }
```

## J.4 Unicode

References to "Unicode" in the following API description shall be interpreted as references to ISO/IEC 10646-1 [15].

## J.5 Virtual keyboards

On platforms where key events are generated from a sequence of other (intermediate) key events, the intermediate key events shall not be visible to GEM applications by any mechanism. Examples of these intermediate key events include;

- For a virtual keyboard, the sequence of keys used to navigate around that keyboard (e.g. `VK_UP`, `VK_LEFT`, `VK_ENTER`).
- For multi-key press entry (as used in some mobile phones), the keys pressed before the final value is resolved.

# Package org.dvb.event

Description

Provides access to user input events before they are processed through the event mechanism of the java.awt package.

The algorithm used for generating UserEvents by the GEM terminal when reporting user input to GEM applications shall be the same as that used for java.awt.event.KeyEvent. For example, pressing the Shift key will cause a KEY_PRESSED event with a VK_SHIFT keyCode, while pressing the 'a' key will result in a VK_A keyCode. After the 'a' key is released, a KEY_RELEASED event will be fired with VK_A, followed by a KEY_TYPED event with a key-Char value of 'A'.

| Class Summary | |
|---|---|
| Interfaces | |
| `UserEventListener` | The listener interface for receiving user inputs. |
| Classes | |
| `EventManager` | The event manager allows an application to receive events coming from the user. |
| `OverallRepository` | This class defines a repository which initially contains all the user events which can be delivered to an application. |
| `RepositoryDescriptor` | An instance of this class will be sent to clients of the DVB event API to notify them (through the interface org.davic.resources.ResourceClient) when they are about to lose, or have lost, access to an event source. |
| `UserEvent` | Represents a user event. |
| `UserEventAvailableEvent` | This event is sent to the resource status event listeners when user input events which had been exclusively reserved by an application are no longer exclusively reserved. |
| `UserEventRepository` | The application will use this class to define the events that it wants to receive. |
| `UserEventUnavailableEvent` | This event is sent to the resource status event listeners when user input events are exclusively reserved by an application. |

# org.dvb.event

# EventManager

### Declaration

```
public class EventManager implements org.davic.resources.ResourceServer

java.lang.Object
  |
  +--org.dvb.event.EventManager
```

### All Implemented Interfaces:

```
org.davic.resources.ResourceServer
```

### Description

The event manager allows an application to receive events coming from the user. These events can be sent exclusively to an application or can be shared between applications. The Event Manager allows also the application to ask for exclusive access to some events, these events being received either from the standard java.awt event mechanism or by the mechanism defined in this package. The EventManager is either a singleton for each GEM application or a singleton for the GEM terminal.

The right to receive events is considered as the same resource regardless of whether it is being handled exclusively or shared. An application successfully obtaining exclusive access to an event results in all other applications losing access to that event, whether the access of those applications was shared or exclusive.

### Constructors

#### EventManager()

```
protected EventManager()
```

Constructor for instances of this class. This constructor is provided for the use of implementations and specifications which extend the present document. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

### Methods

#### addExclusiveAccessToAWTEvent(ResourceClient, UserEventRepository)

```
public boolean addExclusiveAccessToAWTEvent(org.davic.resources.ResourceClient client,
org.dvb.event.UserEventRepository userEvents)
```

An application should use this method to express its intend to have exclusive access to some events, but for these events to be received through the java.awt mechanism. The events the application wishes to receive are defined by the means of the UserEventRepository class. This repository is resolved at the time when this method call is made and adding or removing events from the repository after this method call does not affect the subscription to those events. An exclusive event will be sent to the application if this latest is focused.

The effect of multiple calls to this method by the same application with different instances of UserEventRepository shall be cumulative. If multiple calls to this method succeed in acquiring the events in the specified repositories then the semantics of each successful method call shall be obeyed as specified.

NOTE: This method is intended for applications that wish to ensure particular input events are exclusively handled by a particular UI component - for example, ensuring that number keys go to a PIN number entry widget and nowhere else. Reservations made by this method are not automatically cancelled when the component or application loses focus. Hence if the reservation is not released, no other application will receive the reserved events - those events will be silently discarded. This is due to the absence of any requirement for the platform to provide end users with a visual indication of which application has focus. Without such an indication, if events were not silently discarded, the PIN code events could be received by the wrong application. Applications which have reserved input events using this method should monitor for loss of focus to a different application and change the appearence of the Component - e.g. to indicate that PIN entry is no longer happening. Once the appearance has changed, the application should release the exclusive access so that the previously reserved input events become available to other applications.

Parameters:

`client` - resource client.

`userEvents` - the user events the application wants to be inform of.

Returns:

true if the events defined in the repository have been acquired, false otherwise.

Throws:

`java.lang.IllegalArgumentException` - if the client argument is set to null.

### addResourceStatusEventListener(ResourceStatusListener)

```
public void addResourceStatusEventListener(org.davic.resources.ResourceStatusListener listener)
```

Adds the specified resource status listener so that an application can be aware of any changes regarding exclusive access to some events.

Specified By:

`addResourceStatusEventListener` in interface `ResourceServer`

Parameters:

`listener` - the resource status listener.

### addUserEventListener(UserEventListener, ResourceClient, UserEventRepository)

```
public boolean addUserEventListener(org.dvb.event.UserEventListener listener,
org.davic.resources.ResourceClient client, org.dvb.event.UserEventRepository userEvents)
```

Adds the specified listener to receive events coming from the user in an exclusive manner. The events the application wishes to receive are defined by the means of the UserEventRepository class. This repository is resolved at the time when this method call is made and adding or removing events from the repository after this method call does not affect the subscription to those events. The ResourceClient parameter indicates that the application wants to have an exclusive access to the user event defined in the repository.

The effect of multiple calls to this method by the same application with different instances of UserEventRepository shall be cumulative. If multiple calls to this method succeed in acquiring the events in the specified repositories then the semantics of each successful method call shall be obeyed as specified. Note that this can result in applications receiving the same event through more than one event listener.

Parameters:

`listener` - the listener to receive the user events.

`client` - resource client.

`userEvents` - a class which contains the user events it wants to be informed of.

Returns:

true if the events defined in the repository have been acquired, false otherwise.

Throws:

`java.lang.IllegalArgumentException` - if the client argument is set to null.

### addUserEventListener(UserEventListener, UserEventRepository)

```
public void addUserEventListener(org.dvb.event.UserEventListener listener,
org.dvb.event.UserEventRepository userEvents)
```

Adds the specified listener to receive events coming from the user. The events the application wishes to receive are defined by the means of the UserEventRepository class. This repository is resolved at the time when this method call is made and adding r removing events from the repository after this method call does not affect the subscription to those events.

The effect of multiple calls to this method by the same application with different instances of UserEventRepository shall be cumulative. If multiple calls to this method succeed in acquiring the events in the specified repositories then the semantics of each successful method call shall be obeyed as specified. Note that this can result in applications receiving the same event through more than one event listener.

Parameters:

`listener` - the listener to receive the user events.

`userEvents` - a class which contains the user events it wants to be informed of.

### getInstance()

```
public static org.dvb.event.EventManager getInstance()
```

This method returns the sole instance of the EventManager class. The EventManager class is a singleton.

Returns:

the instance of the EventManager.

### removeExclusiveAccessToAWTEvent(ResourceClient).

```
public void removeExclusiveAccessToAWTEvent(org.davic.resources.ResourceClient client)
```

The application should use this method to release its exclusive access to user events defined by the means of the addExclusiveAccessToAWTEvent method.

Parameters:

`client` - the client that is no longer interested in events previously registered.

### removeResourceStatusEventListener(ResourceStatusListener)

```
public void removeResourceStatusEventListener(org.davic.resources.ResourceStatusListener listener)
```

Removes the specified resource status listener.

Specified By:

`removeResourceStatusEventListener` in interface `ResourceServer`

Parameters:

`listener` - the listener to remove.

### removeUserEventListener(UserEventListener)

```
public void removeUserEventListener(org.dvb.event.UserEventListener listener)
```

Removes the specified listener so that it will no longer receives user events. If it is appropriate (i.e the application has asked for an exclusive access), the exclusive access is lost.

Parameters:

`listener` - the user event listener.

# org.dvb.event

# OverallRepository

### Declaration

```
public class OverallRepository extends UserEventRepository
```

```
java.lang.Object
  |
  +--org.dvb.event.RepositoryDescriptor
        |
        +--org.dvb.event.UserEventRepository
              |
              +--org.dvb.event.OverallRepository
```

### All Implemented Interfaces:

`org.davic.resources.ResourceProxy`

### Description

This class defines a repository which initially contains all the user events which can be delivered to an application. This includes all keycodes for which KEY_PRESSED and KEY_RELEASED events can be generated and all keychars for which KEY_TYPED events can be generated. Note that the set of keycodes and keychars which can be generated is dependent on the input devices of the GEM terminal. For example, this pre-defined repository could be used by an application, which requires a pin code from the user, in order to prevent another applications from receiving events.

### See Also:

`UserEvent`, `org.havi.ui.event.HKeyCapabilities`

### Constructors

### OverallRepository()

`public` OverallRepository()

The constructor for the repository. The name of the constructed instance (as returned by getName()) is implementation dependent.

### OverallRepository(String)

```
public OverallRepository(java.lang.String name)
```

The constructor for the repository with a name.

### Parameters:

`name` - the name to use for the repository

# org.dvb.event RepositoryDescriptor

### Declaration

```
public class RepositoryDescriptor implements org.davic.resources.ResourceProxy
```

```
java.lang.Object
  |
  +--org.dvb.event.RepositoryDescriptor
```

### All Implemented Interfaces:

```
org.davic.resources.ResourceProxy
```

### Direct Known Subclasses:

```
UserEventRepository
```

### Description

An instance of this class will be sent to clients of the DVB event API to notify them (through the interface org.davic.resources.ResourceClient) when they are about to lose, or have lost, access to an event source. This object can be used by the application to get the name of the repository from which it will no longer be able to receive events. All instances of RepositoryDescriptor are also instances of UserEventRepository. This class is preserved for backwards compatibility with existing applications.

### Methods

### getClient()

```
public org.davic.resources.ResourceClient getClient()
```

Return the object which asked to be notified about withdrawal of the event source. This is the object passed as the `ResoourceClient` to whichever of the various 'add' methods on EventManager was used by the application to express interest in this repository.

### Specified By:

`getClient` in interface `ResourceProxy`

### Returns:

the object which asked to be notified about withdrawal of the event source. If the `UserEventRepository` has not yet been added to an `EventManager` then null shall be returned. Once the `UserEventRepository` has been added, the last used `ResourceClient` shall be returned even if the `UserEventRepository` has been since removed.

### getName()

```
public java.lang.String getName()
```

Returns the name of the repository to which the lost, or about to be lost, user event belongs.

### Returns:

String the name of the repository.

# org.dvb.event

# UserEvent

### Declaration

```
public class UserEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.event.UserEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Represents a user event. A user event is defined by a family, a type and either a code or a character. Unless stated otherwise, all constants used in the specification of this class are defined in `java.awt.event.KeyEvent` and its parent classes.

### Fields

#### UEF_KEY_EVENT

```
public static final int UEF_KEY_EVENT
```

the family for events that are coming from the remote control or from the keyboard.

### Constructors

#### UserEvent(Object, int, char, long)

```
public UserEvent(java.lang.Object source, int family, char keyChar, long when)
```

Constructor for a new UserEvent object representing a key being typed. This is the combination of a key being pressed and then being released. The type of UserEvents created with this constructor shall be KEY_TYPED. Key combinations which do not result in characters, such as keys like the red key on a remote control, shall not generate KEY_TYPED events. `KEY_TYPED` events shall have no modifiers and hence shall not report any modifiers as being down.

#### Parameters:

`source` - the `EventManager` which is the source of the event.

`family` - the event family.

`keyChar` - the character typed.

`when` - a long integer that specifies the time the event occurred.

#### Since:

MHP 1.0.1

#### UserEvent(Object, int, int, int, int, long)

```
public UserEvent(java.lang.Object source, int family, int type, int code, int modifiers, long when)
```

Constructor for a new UserEvent object representing a key being pressed.

#### Parameters:

`source` - the `EventManager` which is the source of the event.

`family` - the event family.

`type` - the event type. Either one of KEY_PRESSED or KEY_RELEASED.

`code` - the event code. One of the constants whose name begins in "VK_" defined in java.awt.event.KeyEvent or org.havi.ui.event.HRcEvent.

`modifiers` - the modifiers active when the key was pressed. These have the same semantics as modifiers in `java.awt.event.KeyEvent`.

`when` - a long integer that specifies the time the event occurred.

### Methods

#### getCode()

```
public int getCode()
```

Returns the event code. For KEY_TYPED events, the code is VK_UNDEFINED.

**Returns:**

an int representing the event code.

#### getFamily()

```
public int getFamily()
```

Returns the event family. Could be UEF_KEY_EVENT.

**Returns:**

an int representing the event family.

#### getKeyChar()

```
public char getKeyChar()
```

Returns the character associated with the key in this event. If no valid Unicode character exists for this key event, key-Char is CHAR_UNDEFINED.

**Returns:**

a character.

**Since:**

MHP 1.0.1.

#### getModifiers()

```
public int getModifiers()
```

Returns the modifiers flag for this event. This method shall return 0 for UserEvents constructed using a constructor which does not include an input parameter specifying the modifiers.

**Returns:**

the modifiers flag for this event.

**Since:**

MHP 1.0.1.

#### getType()

```
public int getType()
```

Returns the event type. Could be KEY_PRESSED, KEY_RELEASED or KEY_TYPED.

Returns:

an int representing the event type.

getWhen()

```
public long getWhen()
```

Returns the timestamp of when this event occurred.

Returns:

a long.

Since:

MHP 1.0.2.

isAltDown()

```
public boolean isAltDown()
```

Returns whether or not the Alt modifier is down on this event. This method shall return false for UserEvents constructed using a constructor which does not include an input parameter specifying the modifiers.

Returns:

whether the Alt modifier is down on this event.

Since:

MHP 1.0.1.

isControlDown()

```
public boolean isControlDown()
```

Returns whether or not the Control modifier is down on this event. This method shall return false for UserEvents constructed using a constructor which does not include an input parameter specifying the modifiers.

Returns:

whether the Control modifier is down on this event.

Since:

MHP 1.0.1.

isMetaDown()

```
public boolean isMetaDown()
```

Returns whether or not the Meta modifier is down on this event. This method shall return false for UserEvents constructed using a constructor which does not include an input parameter specifying the modifiers.

Returns:

whether the Meta modifier is down on this event.

Since:

MHP 1.0.1.

isShiftDown()

```
public boolean isShiftDown()
```

Returns whether or not the Shift modifier is down on this event. This method shall return false for UserEvents constructed using a constructor which does not include an input parameter specifying the modifiers.

Returns:

whether the Shift modifier is down on this event.

Since:

MHP 1.0.1.

# org.dvb.event

# UserEventAvailableEvent

Declaration

```
public class UserEventAvailableEvent extends org.davic.resources.ResourceStatusEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.davic.resources.ResourceStatusEvent
              |
              +--org.dvb.event.UserEventAvailableEvent
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

This event is sent to the resource status event listeners when user input events which had been exclusively reserved by an application are no longer exclusively reserved. Where one change in user input event reservation results in instances of this event being sent to several applications, the following shall apply.

- Each application shall receive its own instance of the `UserEventRepository` object which forms the source to this event. Any changes made to that repository by any one application shall not impact the instance seen by any other application.
- Any application receiving an instance of this event is allowed to attempt to exclusively reserve some of the newly available user events. In this situation, the normal resource management policy of the platform as described elsewhere in the present document shall be obeyed.
- Any applications which have registered for shared access to any of these user events shall start receiving those events following receipt of this event.

Since:

MHP 1.0.2.

Constructors

UserEventAvailableEvent(Object)

```
public UserEventAvailableEvent(java.lang.Object source)
```

Constructor for the event.

Parameters:

`source` - a `UserEventRepository` which contains the events which stopped being exclusively reserved.

Since:

MHP 1.0.2.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Returns a `UserEventRepository` which contains the events which were formerly exclusively reserved as passed into the constructor of the instance.

Overrides:

`getSource` in class `ResourceStatusEvent`

Returns:

a `UserEventRepository`.

Since:

MHP 1.0.2.

# org.dvb.event UserEventListener

### Declaration

```
public interface UserEventListener extends java.util.EventListener
```

All Superinterfaces:

```
java.util.EventListener
```

### Description

The listener interface for receiving user inputs.

### Methods

#### userEventReceived(UserEvent)

```
public void userEventReceived(org.dvb.event.UserEvent e)
```

Called by the platform when a user input is received.

Parameters:

`e` - the user input event which was received.

# org.dvb.event UserEventRepository

### Declaration

```
public class UserEventRepository extends RepositoryDescriptor
```

```
java.lang.Object
  |
  +--org.dvb.event.RepositoryDescriptor
        |
        +--org.dvb.event.UserEventRepository
```

All Implemented Interfaces:

`org.davic.resources.ResourceProxy`

Direct Known Subclasses:

`OverallRepository`

Description

The application will use this class to define the events that it wants to receive. Events that are able to be put in the repository are defined in the UserEvent class.

Where a repository includes a `KEY_PRESSED` type event without the `KEY_RELEASED` type event for the same key code or vice versa then exclusive reservations shall be made for both event types but only the one requested shall be received by the listener. Where a repository includes a `KEY_TYPED` event without the corresponding `KEY_PRESSED` and `KEY_RELEASED` events (excluding `KEY_PRESSED` or `KEY_RELEASED` events for modifiers), when an exclusive reservation is requested, it shall also be made for those corresponding `KEY_PRESSED` and `KEY_RELEASED` events but only the requested event shall be received by the listener.

Repositories do not keep a count of the number of times a particular user event is added or removed. Repeatedly adding an event to a repository has no effect. Removing an event removes it regardless of the number of times it has been added. For example, org.dvb.event.UserEventRepository.addUserEvent(UserEvent event) does nothing in case that the event is already in the repository. Events are considered to be already in the repository if an event with the same triplet of family, type and code is already in the repository.

If an application loses exclusive access to a repository, it shall lose access to all events defined in that repository. Repositories are resolved when they are passed into the methods of EventManager. Adding or removing events from the repository after those method calls does not affect the subscription to those events.

Unless stated otherwise, all constants used in the specification of this class are defined in `java.awt.event.KeyEvent` and its parent classes and not in this class.

See Also:

`UserEvent`

Constructors

UserEventRepository(String)

```
public UserEventRepository(java.lang.String name)
```

The method to construct a new UserEventRepository.

Parameters:

`name` - the name of the repository.

Methods

addAllArrowKeys()

```
public void addAllArrowKeys()
```

Adds the key codes for the arrow keys (VK_LEFT, VK_RIGHT, VK_UP, VK_DOWN). Any key codes already in the repository will not be added again. After calling this method, the keycodes shall be present for both the `KEY_PRESSED` and `KEY_RELEASED` modes.

addAllColourKeys()

```
public void addAllColourKeys()
```

Adds the key codes for the colour keys (VK_COLORED_KEY_0, VK_COLORED_KEY_1, VK_COLORED_KEY_2, VK_COLORED_KEY_3). Any key codes already in the repository will not be added again. After calling this method, the keycodes shall be present for both the `KEY_PRESSED` and `KEY_RELEASED` modes.

### addAllNumericKeys()

```
public void addAllNumericKeys()
```

Adds the key codes for the numeric keys (VK_0, VK_1, VK_2, VK_3, VK_4, VK_5, VK_6, VK_7, VK_8, VK_9). Any key codes already in the repository will not be added again. After calling this method, the keycodes shall be present for both the `KEY_PRESSED` and `KEY_RELEASED` modes.

### addKey(int)

```
public void addKey(int keycode)
```

Adds the specified keycode to the repository. Keycodes added in this way shall be listed in the list of user events returned by the `getUserEvent` method. If a key is already in the repository, this method has no effect. After calling this method, the keycode shall be present for both the `KEY_PRESSED` and `KEY_RELEASED` modes.

#### Parameters:

`keycode` - the key code.

### addUserEvent(UserEvent)

```
public void addUserEvent(org.dvb.event.UserEvent event)
```

Adds the given user event to the repository. The values of the modifiers (if any) in the `UserEvent` shall be ignored by the GEM terminal. The value of the source used to construct the specified `UserEvent` shall be ignored by the GEM terminal when the `UserEventRepository` is used to specify events which an application wants to receive.

#### Parameters:

`event` - the user event to be added in the repository.

### getName()

```
public java.lang.String getName()
```

Returns the name of the current repository as passed to the constructor.

#### Overrides:

`getName` in class `RepositoryDescriptor`

#### Returns:

a String with the name of the repository.

### getUserEvent()

```
public org.dvb.event.UserEvent[] getUserEvent()
```

Returns the list of the user events that are in the repository.

#### Returns:

an array which contains the user events that are in the repository.

### removeAllArrowKeys()

```
public void removeAllArrowKeys()
```

Removes the key codes for the arrow keys (VK_LEFT, VK_RIGHT, VK_UP, VK_DOWN). Key codes from this set which are not present in the repository will be ignored. After calling this method, the keycodes shall not be present for both the `KEY_PRESSED` and `KEY_RELEASED` modes.

### removeAllColourKeys()

```
public void removeAllColourKeys()
```

Removes the key codes for the colour keys (VK_COLORED_KEY_0, VK_COLORED_KEY_1, VK_COLORED_KEY_2, VK_COLORED_KEY_3). Key codes from this set which are not present in the repository will be ignored. After calling this method, the keycodes shall not be present for both the `KEY_PRESSED` and `KEY_RELEASED` modes.

### removeAllNumericKeys()

```
public void removeAllNumericKeys()
```

Remove the key codes for the numeric keys (VK_0, VK_1, VK_2, VK_3, VK_4, VK_5, VK_6, VK_7, VK_8, VK_9). Key codes from this set which are not present in the repository will be ignored. After calling this method, the keycodes shall not be present for both the `KEY_PRESSED` and `KEY_RELEASED` modes.

### removeKey(int)

```
public void removeKey(int keycode)
```

The method to remove a key from the repository. Removing a key which is not in the repository has no effect. After calling this method, the keycode shall not be present for both the `KEY_PRESSED` and `KEY_RELEASED` modes.

### Parameters:

`keycode` - the key code.

### removeUserEvent(UserEvent)

```
public void removeUserEvent(org.dvb.event.UserEvent event)
```

Remove a user event from the repository. Removing a user event which is not in the repository shall have no effect.

### Parameters:

`event` - the event to be removed from the repository.

# org.dvb.event

# UserEventUnavailableEvent

### Declaration

```
public class UserEventUnavailableEvent extends org.davic.resources.ResourceStatusEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.davic.resources.ResourceStatusEvent
              |
              +--org.dvb.event.UserEventUnavailableEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This event is sent to the resource status event listeners when user input events are exclusively reserved by an application.

Each application shall receive its own instance of the `UserEventRepository` object which forms the source to this event. Any changes made to that repository by any one application shall not impact the instance seen by any other application.

Any applications which have registered for shared access to any of these user events shall stop receiving those user events following receipt of this event. If such user events become available again, a `UserEventAvailableEvent` shall be generated by the platform before any more of those user events are received by applications.

Since:

MHP 1.0.2.

Constructors

UserEventUnavailableEvent(Object)

```
public UserEventUnavailableEvent(java.lang.Object source)
```

Constructor for the event.

Parameters:

`source` - a `UserEventRepository` which contains the events which were exclusively reserved.

Since:

MHP 1.0.2.

Methods

getSource()

```
public java.lang.Object getSource()
```

Returns a `UserEventRepository` which contains the events which were exclusively reserved as passed into the constructor of the instance.

Overrides:

`getSource` in class `ResourceStatusEvent`

Returns:

a `UserEventRepository`.

Since:

MHP 1.0.2.

# Annex K (normative): DVB-J persistent storage API

# Package

# org.dvb.io.persistent

### Description

Provides extensions to the java.io package for access to files held in persistent storage.

| Class Summary | |
|---|---|
| Classes | |
| FileAccessPermissions | This class encapsulates file access permissions, world, Organisation and owner. |
| FileAttributes | This class encapsulates the attributes of a file stored in persistent storage. |

# org.dvb.io.persistent

# FileAccessPermissions

### Declaration

```
public class FileAccessPermissions

java.lang.Object
  |
  +--org.dvb.io.persistent.FileAccessPermissions
```

### Description

This class encapsulates file access permissions, world, Organisation and owner. World means all applications authorised to access persistent storage. Owner means the application which created the file. Organisation is defined as applications with the same organisation id as defined elsewhere in the present document.

### Constructors

#### FileAccessPermissions(boolean, boolean, boolean, boolean, boolean, boolean)

```
public FileAccessPermissions(boolean readWorldAccessRight, boolean writeWorldAccessRight,
boolean readOrganisationAccessRight, boolean writeOrganisationAccessRight,
boolean readApplicationAccessRight, boolean writeApplicationAccessRight)
```

This constructor encodes all the file access permissions as a set of booleans.

#### Parameters:

`readWorldAccessRight` - read access for all applications.

`writeWorldAccessRight` - write access for all applications.

`readOrganisationAccessRight` - read access for organisation.

`writeOrganisationAccessRight` - write access for organisation.

`readApplicationAccessRight` - read access for the owner.

`writeApplicationAccessRight` - write access for the owner.

### Methods

#### hasReadApplicationAccessRight()

```
public boolean hasReadApplicationAccessRight()
```

Query whether this permission includes read access for the owning application.

Returns:

true if the owning application can have read access, otherwise false.

#### hasReadOrganisationAccessRight()

```
public boolean hasReadOrganisationAccessRight()
```

Query whether this permission includes read access for the organisation

Returns:

true if applications in this organisation can have read access, otherwise false.

#### hasReadWorldAccessRight()

```
public boolean hasReadWorldAccessRight()
```

Query whether this permission includes read access for the world.

Returns:

true if all applications can have read access, otherwise false.

#### hasWriteApplicationAccessRight()

```
public boolean hasWriteApplicationAccessRight()
```

Query whether this permission includes write access for the owning application.

Returns:

true if the owning application can have write access, otherwise false.

#### hasWriteOrganisationAccessRight()

```
public boolean hasWriteOrganisationAccessRight()
```

Query whether this permission includes write access for the organisation.

Returns:

true if applications in this organisation can have read access, otherwise false.

#### hasWriteWorldAccessRight()

```
public boolean hasWriteWorldAccessRight()
```

Query whether this permission includes write access for the world.

Returns:

true if all applications can have write access, otherwise false.

#### setPermissions(boolean, boolean, boolean, boolean, boolean, boolean)

```
public void setPermissions(boolean ReadWorldAccessRight, boolean WriteWorldAccessRight,
boolean ReadOrganisationAccessRight, boolean WriteOrganisationAccessRight,
boolean ReadApplicationAccessRight, boolean WriteApplicationAccessRight)
```

This method allows to modify the permissions on this instance of the FileAccessPermission class.

Parameters:

`ReadWorldAccessRight` - read access for all applications.

`WriteWorldAccessRight` - write access for all applications.

`ReadOrganisationAccessRight` - read access for organisation.

`WriteOrganisationAccessRight` - write access for organisation.

`ReadApplicationAccessRight` - read access for the owner.

`WriteApplicationAccessRight` - write access for the owner.

# org.dvb.io.persistent

# FileAttributes

## Declaration

```
public class FileAttributes

java.lang.Object
  |
  +--org.dvb.io.persistent.FileAttributes
```

## Description

This class encapsulates the attributes of a file stored in persistent storage. The default attributes for a file are low priority, owner read/write only permissions and null expiration date.

## Fields

### PRIORITY_HIGH

```
public static final int PRIORITY_HIGH
```

Value for use as a file priority.

### PRIORITY_LOW

```
public static final int PRIORITY_LOW
```

Value for use as a file priority.

### PRIORITY_MEDIUM

```
public static final int PRIORITY_MEDIUM
```

Value for use as a file priority.

## Constructors

### FileAttributes(Date, FileAccessPermissions, int)

```
public FileAttributes(java.util.Date expiration_date, org.dvb.io.persistent.FileAccessPermissions p,
int priority)
```

Constructor.

Parameters:

`expiration_date` - an expiration date or null.

`p` - the access permissions to use.

`priority` - the priority to use in persistent storage.

### Methods

#### getExpirationDate()

```
public java.util.Date getExpirationDate()
```

Returns the expiration date. It will return the value used by the platform, which need not be the same as the value set.

Returns:

the expiration date.

#### getFileAttributes(File)

```
public static org.dvb.io.persistent.FileAttributes getFileAttributes(java.io.File f)
throws IOException
```

Get the attributes of a file.

Parameters:

`f` - the file to use.

Returns:

a copy of the attributes of a file.

Throws:

`java.lang.SecurityException` - if the application is denied access to the file or to directories needed to reach the file by security policy.

`java.io.IOException` - if access to the file fails due to an IO error or if the file reference is not to a valid location in persistent storage.

#### getPermissions()

```
public org.dvb.io.persistent.FileAccessPermissions getPermissions()
```

Returns the file access permissions.

Returns:

the file access permissions.

#### getPriority()

```
public int getPriority()
```

Returns the priority to use in persistent storage.

Returns:

the priority.

#### setExpirationDate(Date)

```
public void setExpirationDate(java.util.Date d)
```

Sets the expiration date. This field is a hint to the platform to identify the date after which a file is no longer useful as percieved by the application. The platform may choose to use a different date than the one given as a parameter.

Parameters:

`d` - the expiration date.

### setFileAttributes(FileAttributes, File)

```
public static void setFileAttributes(org.dvb.io.persistent.FileAttributes p, java.io.File f)
throws IOException
```

Associate a set of file attributes with a file.

Parameters:

`p` - the file attributes to use.

`f` - the file to use.

Throws:

`java.lang.SecurityException` - if the application is either denied access to the file or directories needed to reach the file by security policy or is not authorised to modify the attributes of the file.

`java.io.IOException` - if access to the file fails due to an IO error or if the file reference is not to a valid location in persistent storage

### setPermissions(FileAccessPermissions)

```
public void setPermissions(org.dvb.io.persistent.FileAccessPermissions p)
```

Sets the file access permissions.

Parameters:

`p` - the file access permissions.

### setPriority(int)

```
public void setPriority(int priority)
```

Sets the priority to use in persistent storage.

Parameters:

`priority` - the priority to set.

# Annex L (normative): User settings and preferences API

In the class `org.dvb.user.GeneralPreference`, the preference "User Name" requires that name be reported as first name followed by last name. It is understood that "first name" and "last name" are ambiguous concepts in some locales. For this reason, in GEM this property is only required to contain the name of the user, in some order that is suitable for presentation to an end user.

# Package org.dvb.user

### Description

Provides access to settings and preferences configured by the end-user.

| Class Summary | |
|---|---|
| **Interfaces** | |
| `UserPreferenceChangeListener` | An application wishing to be informed of any change to a user preference implements this interface. |
| **Classes** | |
| `Facility` | A facility maps a preference's name to a single value or to an array of values. |
| `GeneralPreference` | This class defines a set of general preferences. |
| `Preference` | This abstract class defines the Preference object. |
| `UserPreferenceChangeEvent` | This class defines the event sent to appropriate listeners when a user preference has been changed. |
| `UserPreferenceManager` | The UserPreferenceManager class gives access to the user preference settings. |
| `UserPreferencePermission` | This class is for user preference and setting permissions. |
| **Exceptions** | |
| `UnsupportedPreferenceException` | Thrown when a non-supported preference is used. |

# org.dvb.user Facility

### Declaration

```
public class Facility

java.lang.Object
  |
  +--org.dvb.user.Facility
```

### Description

A facility maps a preference's name to a single value or to an array of values. A facility enables an application to define the list of values supported for a specified preference. For example, if an application is available in English or French then it can create a Facility ("User Language", {"English", "French"}). When the application will retrieve the "User Language" from the general preference it will specify the associated facility in order to get a Preference which will contain a set a values compatible with those supported by the application.

### Constructors

#### Facility(String, String)

```
public Facility(java.lang.String preference, java.lang.String value)
```

Creates a Facility with a single value. This facility can be used by an application to retrieve a preference compatible with its capabilities.

Parameters:

`preference` - a String representing the name of the preference.

`value` - a String representing the value of the preference.

#### Facility(String, String[])

```
public Facility(java.lang.String preference, java.lang.String[] values)
```

Creates a Facility with a set of values. This facility can be used by an application to retrieve a preference compatible with its capabilities.

Parameters:

`preference` - a String representing the name of the preference.

`values` - an array of String representing the set of values.

# org.dvb.user

# GeneralPreference

### Declaration

```
public final class GeneralPreference extends Preference
```

```
java.lang.Object
  |
  +--org.dvb.user.Preference
        |
        +--org.dvb.user.GeneralPreference
```

### Description

This class defines a set of general preferences. These preferences are read from the receiver and each application (downloaded or not) can access them through the `UserPreferenceManager.read` method. The standardized preferences are "User Language", "Parental Rating", "User Name", "User Address", "User @", "Country Code", "Default Font Size", "Post Code".

When constructed, objects of this class are empty and have no values defined. Values may be added using the add methods inherited from the Preference class or by calling `UserPreferenceManager.read`.

The encodings of these standardized preferences are as follows:

- User Language: 3 letter ISO 639 [84] language codes.
- Parental Rating: string using the same encoding as returned by `javax.tv.service.guide.Content-RatingAdvisory.getDisplayText`.
- User Name: Name of the user. This shall be in an order that is appropriate for presentation directly to the user, e.g. in Western Europe, listing the first name first and the family name last is recommended as being culturally appropriate in many locales.
- User Address: postal address of the user, may contain multiple lines separated by carriage return characters (as defined in table 77).

- User @: e-mail address of the user in the SMTP form as defined in RFC 821 [71].
- Country Code: two letter ISO 3166-1 [25] country code.
- Default Font Size: preferred font size for normal body text expressed in points, decimal integer value encoded as a string (26 is the default; differing size indicates a preference of different font size than usual).
- "Post Code": no standard encoding is defined since the formats of post codes (US zip codes, German Postleitzahl) are normally country specific. The format used should be the most natural one for the country identified by the "Country Code" preference.

The preference names are treated as case-insensitive. The preference names shall be considered equal at least when the method java.lang.String.equalsIgnoreCase() returns true for the strings when the locale Locale.UK is used. Depending on the locale used in the implementation, implementations are allowed to consider equal also other upper and lower case character pairs in addition to those defined by the Locale.UK locale.

The standardized preference names in GEM shall only use such letters where the upper and lower case characters are recognized by the Locale.UK locale. Since the "Post Code" preference forms part of the "User Address" preference, successful calls to UserPreferenceManager.write(..) for a GeneralPreference("Post Code") shall modify the "User Address" preference. Hence UserPreferenceChangeEvents shall be generated for both "Post Code" and "User Address" in this specific situation. Successful calls to UserPreferenceManager.write(...) for a GeneralPreference("User Address") shall generate a UserPreferenceChangeEvent for "Post Code" if and only if the post code part of the address changes.

### Constructors

#### GeneralPreference(String)

```
public GeneralPreference(java.lang.String name)
throws IllegalArgumentException
```

Constructs a GeneralPreference object. A general preference maps a preference name to a list of strings.

#### Parameters:

`name` - the general preference name.

#### Throws:

`java.lang.IllegalArgumentException` - if the preference's name is not supported.

# org.dvb.user

# Preference

### Declaration

```
public abstract class Preference

java.lang.Object
  |
  +--org.dvb.user.Preference
```

### Direct Known Subclasses:

`GeneralPreference`

### Description

This abstract class defines the Preference object. A Preference maps a name to a list of favourite values. The first element in the list is the favourite value for this preference.

The preference names are treated as case-insensitive. The preference names shall be considered equal at least when the method java.lang.String.equalsIgnoreCase() returns true for the strings when the locale java.util.Locale.UK is used. Depending on the locale used in the implementation, implementations are allowed to consider equal also other upper and lower case character pairs in addition to those defined by the Locale.UK locale.

The standardized preference names in GEM shall only use such letters where the upper and lower case characters are recognized by the Locale.UK locale.

### Constructors

#### Preference()

```
protected Preference()
```

This protected constructor is only present to enable sub-classes of this one to be defined by the platform. It is not intended to be used by inter-operable applications.

#### Preference(String, String)

```
public Preference(java.lang.String name, java.lang.String value)
```

Creates a new preference with the specified name and the specified value. This single value will be the favourite one for this preference.

##### Parameters:

`name` - a String object representing the name of the preference.

`value` - a String object representing the value of the preference.

#### Preference(String, String[])

```
public Preference(java.lang.String name, java.lang.String[] value)
```

Creates a new preference with the specified name and the specified value set. Each value in the value set must appear only once. The behaviour if a value is duplicated is implementation dependent.

##### Parameters:

`name` - a String object representing the name of the preference.

`value` - an array of String objects representing the set of values for this preference ordered from the most favourite to the least favourite.

### Methods

#### add(int, String)

```
public void add(int position, java.lang.String value)
```

Adds a new value for this preference. The value is inserted at the specified position. If the value is already in the list then it is moved to the position specified. 1 If the position is greater than or equal to the length of the list, then the value is added to the end of this list. If the position is negative, then the value is added to the beginning of this list.

##### Parameters:

`position` - an int representing the position in the list counting from zero.

`value` - a String representing the new value to insert.

#### add(String)

```
public void add(java.lang.String value)
```

Adds a new value for this preference. The value is added to the end of the list. If the value is already in the list then it is moved to the end of the list.

##### Parameters:

`value` - a String object representing the new value.

### add(String[])

```
public void add(java.lang.String[] values)
```

Adds several new values for this preferences. The values are added to the end of the list in the same order as they are found in the array passed to this method. Any values already in the list are moved to the position in the list which they would have if they were not already present.

Parameters:

`values` - an array of strings representing the values to add.

Since:

MHP 1.0.1.

### getFavourites()

```
public java.lang.String[] getFavourites()
```

Returns the list of favourite values for this preference. Returns an empty array if no value sets are defined for this preference.

Returns:

an array of String representing the favourite values for this preference.

### getMostFavourite()

```
public java.lang.String getMostFavourite()
```

Returns the most favourite value for this preference, that is, the first element of the list.

Returns:

a String representing the most favourite value for this preference. Returns null if no value is defined for this preference.

### getName()

```
public java.lang.String getName()
```

Returns the name of the preference.

Returns:

a String object representing the name of the preference.

### getPosition(String)

```
public int getPosition(java.lang.String value)
```

Returns the position in the list of the specified value.

Parameters:

`value` - a String representing the value to look for.

Returns:

an integer representing the position of the value in the list counting from zero. If the value is not found then it returns -1.

### hasValue()

```
public boolean hasValue()
```

Tests if this preference has at least one value set.

Returns:

true if this preference has at least one value set, false otherwise.

remove(String)

```
public void remove(java.lang.String value)
```

Removes the specified value from the list of favourites. If the value is not in the list then the method call has no effect.

Parameters:

`value` - a String representing the value to remove.

removeAll()

```
public void removeAll()
```

Removes all the values of a preference

Since:

MHP 1.0.1.

setMostFavourite(String)

```
public void setMostFavourite(java.lang.String value)
```

Sets the most favourite value for this preference. If the value is already in the list, then it is moved to the head. If the value is not already in the list then it is added at the head.

Parameters:

`value` - the most favourite value.

toString()

```
public java.lang.String toString()
```

Convert name and favourites to a String.

Overrides:

`toString` in class `Object`.

Returns:

the preference name and favourites.

# org.dvb.user UnsupportedPreferenceException

Declaration

```
public class UnsupportedPreferenceException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.user.UnsupportedPreferenceException
```

All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown when a non-supported preference is used.

### Constructors

#### UnsupportedPreferenceException()

```
public UnsupportedPreferenceException()
```

Constructs a `UnsupportedPreferenceException` with no detail message.

#### UnsupportedPreferenceException(String)

```
public UnsupportedPreferenceException(java.lang.String a)
```

Constructs a `UnsupportedPreferenceException` with a detail message.

### Parameters:

`a` - the detail message.

# org.dvb.user UserPreferenceChangeEvent

### Declaration

```
public class UserPreferenceChangeEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.user.UserPreferenceChangeEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This class defines the event sent to appropriate listeners when a user preference has been changed.

### Constructors

#### UserPreferenceChangeEvent(String)

```
public UserPreferenceChangeEvent(java.lang.String preferenceName)
```

Constructs a new event.

### Parameters:

`preferenceName` - the name of the modified preference.

### Methods

#### getName()

```
public java.lang.String getName()
```

Returns the name of the modified Preference.

### Returns:

the Preference name.

# org.dvb.user UserPreferenceChangeListener

### Declaration

```
public interface UserPreferenceChangeListener extends java.util.EventListener
```

### All Superinterfaces:

```
java.util.EventListener
```

### Description

An application wishing to be informed of any change to a user preference implements this interface.

### Methods

#### receiveUserPreferenceChangeEvent(UserPreferenceChangeEvent)

```
public void receiveUserPreferenceChangeEvent(org.dvb.user.UserPreferenceChangeEvent e)
```

This method is called when a user preference changes.

#### Parameters:

`e` - the event notifying this event.

# org.dvb.user UserPreferenceManager

### Declaration

```
public class UserPreferenceManager

java.lang.Object
  |
  +--org.dvb.user.UserPreferenceManager
```

### Description

The UserPreferenceManager class gives access to the user preference settings. This class provides a set of methods that allow an application to read or save user settings. It also provides a mechanism to notify applications when a preference has been modified. The value of a user setting, retrieved with the read method, is a copy of the value that is stored in the receiver. The write method, if authorized, overwrites the stored value.

NOTE: GEM implementations are not required to validate the values in Preference objects, even those which are saved using the write method. Applications with write permissions need to be very careful that the values written are valid. Applications reading permissions need to be aware of the possibility that a previous application has set an invalid value.

### Methods

#### addUserPreferenceChangeListener(UserPreferenceChangeListener)

```
public void addUserPreferenceChangeListener(org.dvb.user.UserPreferenceChangeListener l)
```

Adds a listener for changes in user preferences as held in the MHP terminal. Specifically this includes changes made by MHP applications succeeding in calling the write() method on this class. If the implementation of the MHP terminal allows the end user to change preferences then these changes also includes changes made to preferences by this mechanism. It does not include changes made to a Preference instance within the scope of a single MHP application.

Parameters:

`l` - the listener to add.

### getInstance()

```
public static org.dvb.user.UserPreferenceManager getInstance()
```

Return an instance of the `UserPreferenceManager` for this application. Repeated calls to this method by the same application shall return the same instance.

Returns:

an instance of `UserPreferenceManager`.

### read(Preference)

```
public void read(org.dvb.user.Preference p)
```

Allows an application to read a specified user preference. When end-user preferences are read into a `Preference` object from the MHP terminal, the ordering of these values shall be as determined by the end-user, from most preferred to least preferred to the extent that this is known.

Parameters:

`p` - an object representing the preference to read.

Throws:

`java.lang.SecurityException` - if the calling application is denied access to this preference.

### read(Preference, Facility)

```
public void read(org.dvb.user.Preference p, org.dvb.user.Facility facility)
```

Allows an application to read a specified user preference taking into account the facility defined by the application. After this method returns, the values in the `Preference` object shall be the values of that user preference with any unsupported values from the `Facility` removed from that list. Note that the order of values returned here need not be the same as that returned by `read(Preference)`.

If the intersection between the two sets of values is empty then the preference will have no value. If there is a mis-match between the name of the preference used when constructing the facility and the name of the preference used in this method then the preference will have no value.

Parameters:

`p` - an object representing the preference to read.

`facility` - the preferred values of the application for the preference

Throws:

`java.lang.SecurityException` - if the calling application is denied access to this preference.

### removeUserPreferenceChangeListener(UserPreferenceChangeListener)

```
public void removeUserPreferenceChangeListener(org.dvb.user.UserPreferenceChangeListener l)
```

Removes a listener for changes in user preferences.

Parameters:

`l` - the listener to remove.

### write(Preference)

```
public void write(org.dvb.user.Preference p)
throws UnsupportedPreferenceException, IOException
```

Saves the specified user preference. If this method succeeds then it will change the value of this preference for all future MHP applications.

Parameters:

`p` - the preference to save.

Throws:

`UnsupportedPreferenceException` - if the preference provided is not a standardized preference as defined for use with `GeneralPreference`.

`java.lang.SecurityException` - if the application does not have permission to call this method.

`java.io.IOException` - if saving the preference fails for other I/O reasons.

# org.dvb.user

# UserPreferencePermission

Declaration

```
public class UserPreferencePermission extends java.security.BasicPermission
```

```
java.lang.Object
  |
  +--java.security.Permission
        |
        +--java.security.BasicPermission
              |
              +--org.dvb.user.UserPreferencePermission
```

All Implemented Interfaces:

`java.security.Guard`, `java.io.Serializable`

Description

This class is for user preference and setting permissions. A UserPreferencePermission contains a name, but no actions list.

The permission name can either be "read" or "write". The "read" permission allows an application to read the user preferences and settings (using `UserPreferenceManager.read`) for which read access is not always granted. Access to the following settings/preferences is always granted: "User Language", "Parental Rating", "Default Font Size" and "Country Code".

The "write" permission allows an application to modify user preferences and settings (using `UserPreference-Manager.write`).

Constructors

UserPreferencePermission(String)

```
public UserPreferencePermission(java.lang.String name)
```

Creates a new UserPreferencePermission with the specified name. The name is the symbolic name of the User-PreferencePermission.

Parameters:

`name` - the name of the UserPreferencePermission.

UserPreferencePermission(String, String)

```
public UserPreferencePermission(java.lang.String name, java.lang.String actions)
```

Creates a new UserPreferencePermission object with the specified name. The name is the symbolic name of the User-PreferencePermission, and the actions String is unused and should be null. This constructor exists for use by the Policy object to instantiate new Permission objects.

Parameters:

`name` - the name of the UserPreferencePermission.

`actions` - should be null.

# Annex M (informative): SI access API

GEM does not specify a particular SI API (such as DVB SI), but nonetheless, a functionally equivalent "SI" is required for some profiles.

# Annex N (normative): Streamed media API extensions

References to "720 x 576" frames are to be read as referring to standard definition frames, as defined in the GEM terminal specification.

## N.1 Active Format Definition

The documentation for the method `getActiveFormatDefinition()` of the class `org.dvb.media.VideoFormatControl` specifies that the `active_format field` of the Active Format Descriptor shall be returned if it is present. GEM terminal specifications may specify signalling that is used to determine this value. A GEM terminal specification may specify a different descriptor to be signalled. If a different descriptor is signalled, the method shall in all cases return the corresponding value as defined in TS 101 154 [2], annex B.

NOTE: The "public final static int" values specified in the class `VideoFormatControl` may be inlined at the time an application's Java source is compiled. Therefore, it is essential that the same value be used in all GEM terminal specifications, e.g. the same numeric value of DAR_16_9 (as specified by TS 101 154 [2]) is returned, even if the underlying signalling used by the GEM terminal specification uses a different value to indicate the same thing.

### N.1.1 MHP Signalling for Active Format Definition

GEM terminals obeying a GEM terminal specification that adopts the MHP definition of the functional equivalent named "active format descriptor" shall accept the Active Format Descriptor as specified by the documentation for the method `getActiveFormatDefinition()` of the class `org.dvb.media.VideoFormatControl`.

### N.1.2 Drip-feed APIs

For GEM terminal specifications, which don't require MPEG-2 Video "drips", the corresponding APIs in `org.dvb.media.DripFeedPermission` and `org.dvb.media.DripFeedDataSource` are not required.

### N.1.3 VideoFormatControl

Signalling for the active format description as defined in annex B of TS 101 154 [2] is optional in GEM.

# Package org.dvb.media

Description

Provides DVB specific extensions to the Java Media Framework.

| Class Summary | |
|---|---|
| **Interfaces** | |
| `BackgroundVideoPresentationControl` | A control to support the setting and querying of the video presentation for background players. |
| `ComponentBasedPlayerControl` | The presence of this Control on a Player indicates that the Player can be used as either a background Player or a component-based Player. |
| `DVBMediaSelectControl` | DVBMediaSelectControl extends `MediaSelectControl` allowing the selection of different kinds of content in a running `Player`. |
| `SubtitleListener` | Report that a subtitle event has happened. |

| Class Summary | |
|---|---|
| SubtitlingEventControl | Allow applications to register and unregister their interest in events related to the availability and presentation of subtitles. |
| VideoFormatControl | This provides a means for applications to get information associated with the format and aspect ratio of the video being presented to the user. |
| VideoFormatListener | The listener used to receive video format events |
| VideoPresentationControl | A control to support setting and querying the video presentation. |
| **Classes** | |
| ActiveFormatDescriptionChangedEvent | Event signalling that the transmitted active format definition has changed |
| AspectRatioChangedEvent | Event signalling that the aspect ratio of the transmitted video has changed |
| CAStopEvent | This event is generated whenever access to a service is withdrawn by the CA system, e.g. at the end of a free preview period. |
| Codecs | Class providing access to information on video and audio formats that a GEM terminal can decode. |
| DFCChangedEvent | Event signalling that the decoder format conversion being used has changed |
| DripFeedDataSource | This class allows to create a source for a JMF player to be able to feed the decoder progressively with parts of a clip (e.g. |
| DripFeedPermission | This class represents a permission to access the drip feed mode. |
| NoComponentSelectedEvent | This event is generated whenever presentation of a stream stops because there are no selected components to present. |
| PresentationChangedEvent | This event is generated whenever the content being presented by a player changes for reasons outside the control of the application. |
| ServiceRemovedEvent | This event is generated whenever access to a service stops because the service concerned has been removed from the network. |
| StopByResourceLossEvent | This event is generated whenever presentation of a stream stops because the player has lost so many resources that it cannot continue. |
| SubtitleAvailableEvent | Report that subtitles are available to be presented having been unavailable. |
| SubtitleNotAvailableEvent | Inform an application that a subtitle stream has vanished from the network. |
| SubtitleNotSelectedEvent | Report that subtitles are not now selected. |
| SubtitleSelectedEvent | Report that subtitles are now selected. |
| VideoFormatEvent | The base class for all other events relating to changes in video format |
| VideoTransformation | VideoTransformation objects express video transformations, i.e. |
| **Exceptions** | |
| CAException | This exception is thrown when access to a media stream is denied by the CA system. |

# org.dvb.media

# ActiveFormatDescriptionChangedEvent

### Declaration

```
public class ActiveFormatDescriptionChangedEvent extends VideoFormatEvent

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.media.VideoFormatEvent
              |
              +--org.dvb.media.ActiveFormatDescriptionChangedEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Event signalling that the transmitted active format definition has changed.

### Constructors

#### ActiveFormatDescriptionChangedEvent(Object, int)

```
public ActiveFormatDescriptionChangedEvent(java.lang.Object source, int newFormat)
```

Construct the event.

#### Parameters:

`source` - the source of the event.

`newFormat` - the new active format description.

### Methods

#### getNewFormat()

```
public int getNewFormat()
```

Get the new active format description.

#### Returns:

the new active format description. The value of this is represented by one of the constants from `VideoFormatControl` and shall be the value passed into the constructor of the event.

# org.dvb.media

# AspectRatioChangedEvent

### Declaration

```
public class AspectRatioChangedEvent extends VideoFormatEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.media.VideoFormatEvent
              |
              +--org.dvb.media.AspectRatioChangedEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Event signalling that the aspect ratio of the transmitted video has changed.

### Constructors

#### AspectRatioChangedEvent(Object, int)

```
public AspectRatioChangedEvent(java.lang.Object source, int newRatio)
```

Construct the event.

#### Parameters:

`source` - the source of the event.

`newRatio` - the new aspect ratio of the transmitted video.

### Methods

#### getNewRatio()

```
public int getNewRatio()
```

Get the new aspect ratio of the transmitted video.

Returns:

the new aspect ratio of the video. The value of this is represented by one of the constants from the VideoFormatControl class and shall be the value passed into the constructor of the event.

# org.dvb.media BackgroundVideoPresentationControl

### Declaration

```
public interface BackgroundVideoPresentationControl extends VideoPresentationControl
```

### All Superinterfaces:

```
javax.media.Control, VideoPresentationControl
```

### Description

A control to support the setting and querying of the video presentation for background players.

### Methods

#### getClosestMatch(VideoTransformation)

```
public org.dvb.media.VideoTransformation getClosestMatch(org.dvb.media.VideoTransformation t)
```

This method takes a video transformation and returns the closest match of that video transformation that can be supported for the currently selected video. If the input video transformation can be supported, then the output video transformation will have the same parameters as the input video transformation. The definition of 'closest match' is implementation dependent.

Parameters:

`t` - the input video transformation.

Returns:

the closest match to the input video transformation. If the input video transformation is supported, then the input video transformation will be returned (the same instance), otherwise a newly created instance will be returned.

#### getVideoTransformation()

```
public org.dvb.media.VideoTransformation getVideoTransformation()
```

Return the current video transformation.

Returns:

the video transformation (clipping/scaling/positioning) that is currently used for displaying the video.

#### setVideoTransformation(VideoTransformation)

```
public boolean setVideoTransformation(org.dvb.media.VideoTransformation t)
```

Sets a new video transformation (clipping/scaling/positioning). If the new video transformation is not supported, then the video transformation will not be changed at all (no best effort attempt is made).

Parameters:

`t` - the new video transformation.

Returns:

true if the video transformation is supported and has been set, false otherwise.

# org.dvb.media

# CAException

## Declaration

```
public class CAException extends java.io.IOException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.media.CAException
```

## All Implemented Interfaces:

```
java.io.Serializable
```

## Description

This exception is thrown when access to a media stream is denied by the CA system. It will typically be thrown by calls to DataSource.start() when access to the stream accessed by the DataSource is denied.

## Constructors

### CAException()

```
public CAException()
```

Constructor without a reason.

### CAException(String)

```
public CAException(java.lang.String reason)
```

Constructor with a reason.

Parameters:

`reason` - the reason why access to the stream failed.

# org.dvb.media

# CAStopEvent

### Declaration

```
public class CAStopEvent extends javax.media.StopEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--javax.media.ControllerEvent
              |
              +--javax.media.TransitionEvent
                    |
                    +--javax.media.StopEvent
                          |
                          +--org.dvb.media.CAStopEvent
```

### All Implemented Interfaces:

```
javax.media.MediaEvent, java.io.Serializable
```

### Description

This event is generated whenever access to a service is withdrawn by the CA system, e.g. at the end of a free preview period. It is not generated when an attempt to construct a Player or DataSource fails due to CA restrictions, or when only some of the presented content is not available or alternate content is presented. Generation of this event informs the application that the Player is no longer presenting any content.

### Constructors

### CAStopEvent(Controller)

```
public CAStopEvent(javax.media.Controller source)
```

Construct an event.

### Parameters:

`source` - the controller which was presenting the service.

### CAStopEvent(Controller, int, int, int, MediaLocator)

```
public CAStopEvent(javax.media.Controller source, int previous, int current, int target,
javax.media.MediaLocator stream)
```

Construct an event.

### Parameters:

`source` - the controller which was presenting the service.

`stream` - the locator of the stream from which access has been withdrawn.

`previous` - the previous state of the controller.

`current` - the current state of the controller.

`target` - the target state of the controller.

### Methods

#### getStream()

```
public javax.media.MediaLocator getStream()
```

This method returns the stream from which access has been withdrawn.

Returns:

the locator for the stream concerned.

# org.dvb.media

# Codecs

### Declaration

```
public class Codecs

java.lang.Object
  |
  +--org.dvb.media.Codecs
```

### Description

Class providing access to information on video and audio formats that a GEM terminal can decode.

Since:

MHP 1.2.

### Methods

#### getAudioFormats()

```
public static java.lang.String[] getAudioFormats()
```

Return the list of audio formats that can be decoded. Formats are encoded as MPEG-7 term IDs as extended by the DVB IPI handbook. e.g. MPEG-1 layer 3 audio is encoded as "3.3".

Returns:

an array of audio formats.

#### getVideoFormats()

```
public static java.lang.String[] getVideoFormats()
```

Return the list of video formats that can be decoded. Formats are encoded as MPEG-7 term IDs as extended by the DVB IPI handbook. e.g. MPEG-2 video, main profile, main level is encoded as "2.2.2".

Returns:

an array of video formats.

#### isAudioFormatSupported(String)

```
public static boolean isAudioFormatSupported(java.lang.String term)
```

Query if a specific audio format can be decoded. Formats are encoded as MPEG-7 term IDs as extended by the DVB IPI handbook. e.g. MPEG-1 layer 3 audio is encoded as "3.3".

Parameters:

`term` - a audio format.

Returns:

true if the specified format can be decoded otherwise false.

### isVideoFormatSupported(String)

```
public static boolean isVideoFormatSupported(java.lang.String term)
```

Query if a specific video format can be decoded. Formats are encoded as MPEG-7 term IDs as extended by the DVB IPI handbook. e.g. MPEG-2 video, main profile, main level is encoded as "2.2.2".

Parameters:

`term` - a video format.

Returns:

true if the specified format can be decoded otherwise false.

# org.dvb.media ComponentBasedPlayerControl

### Declaration

```
public interface ComponentBasedPlayerControl extends javax.media.Control
```

### All Superinterfaces:

```
javax.media.Control
```

### Description

The presence of this Control on a Player indicates that the Player can be used as either a background Player or a component-based Player. Players for broadcast streaming formats that do not provide this control will always return null from their getVisualComponent() method.

A Player that provides this control must continue to provide an instance of it (not necessarily the same instance) after any transitions from background to component-based modes and vice-versa.

### Methods

### releaseVisualComponent()

```
public boolean releaseVisualComponent()
throws IllegalStateException
```

Transition the associated Player from a component-based Player to a background Player. Also releases the ownership of the visual component to allow other applications to acquire it.

The calling application must own the visual component - i.e. it must have obtained a non-null visual component through Player.getVisualComponent() on the Player associated with this control.

If the application has called paint() on that visual component, then this method transitions the Player from a component-based Player to a background Player, and releases the visual component. Otherwise, the Player is still a background Player so this method merely releases the visual component.

If this call transitions the Player from a component-based Player to a background Player, then the video is intially sized and positioned as close to the last location of the visual component as possible, given the limitations exposed after the transition by org.dvb.media.BackgroundVideoPresentationControl .getClosestMatch() or (equivalently) javax.tv.media.AWTVideoSizeControl.checkSize().

After the transition, applications shall re-acquire the list of JMF controls for the player. It is implementation dependent whether any previous JMF controls are re-used. Applications shall not use any previous JMF controls which are not in the new list of JMF controls.

The application should ensure that the visual component is removed from the AWT hierarchy before making this call. After this call, interoperable applications should not make any calls to the visual component aquired before this call. However, such calls will not affect the Player in any way. (For example, calling paint() on that visual component will not cause the Player to become a component-based player). The appearence of the visual component if it is displayed after this call is implementation-dependant.

Future calls to Player.getVisualComponent() will return a visual component that is different from the visual component returned before this call.

Returns:

true iff the Player has changed from component-based to background mode.

Throws:

`java.lang.IllegalStateException` - If the calling application does not own the visual component.

### swapVideoComponent(Player)

```
public void swapVideoComponent(javax.media.Player other)
```

Swap the presentation of the video for this JMF player and another Player between a component and the background.

Parameters:

`other` - the other player to use in the swap.

Throws:

`java.lang.IllegalArgumentException` - if both this player and the other player are background players or if both this player and the other player are component players or if the calling application does not have a reference to the java.awt.Component used in the swap or if either player is not presenting video

Since:

MHP 1.1.3.

# org.dvb.media

# DFCChangedEvent

### Declaration

```
public class DFCChangedEvent extends VideoFormatEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.media.VideoFormatEvent
              |
              +--org.dvb.media.DFCChangedEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Event signalling that the decoder format conversion being used has changed.

### Constructors

#### DFCChangedEvent(Object, int)

```
public DFCChangedEvent(java.lang.Object source, int newDFC)
```

Construct the event.

Parameters:

`source` - the source of the event.

`newDFC` - the new decoder format conversion being used.

### Methods

#### getNewDFC()

```
public int getNewDFC()
```

Get the new decoder format conversion.

Returns:

the new decoder format conversion. The value of this is represented by one of the constants from the VideoFormat-Control class and shall be the value passed into the constructor of the event.

# org.dvb.media

# DripFeedDataSource

### Declaration

```
public class DripFeedDataSource extends javax.media.protocol.DataSource
```

```
java.lang.Object
  |
  +--javax.media.protocol.DataSource
        |
        +--org.dvb.media.DripFeedDataSource
```

### All Implemented Interfaces:

`javax.media.protocol.Controls, javax.media.Duration`

### Description

This class allows to create a source for a JMF player to be able to feed the decoder progressively with parts of a clip (e.g. I or P MPEG-2 frame) according to the drip-fed mode format defined in the GEM content format clause.

To start using the drip-feed mode, the application needs to instantiate a player representing a MPEG-2 video decoder and have its source be a DripFeedDataSource instance.

A DripFeedDataSource instance can be obtained by calling the default constructor of the class.

A player that will be bound to a MPEG-2 video decoder (when realized) can be created with a locator of the following text representation: "dripfeed://".

After having the DripFeedDataSource connected to a Player representing a MPEG-2 video decoder, the following rules applies:

- If the feed method is called when the player is in the "prefetched" state the image will be stored so that when the player goes in the "started" state it will be automatically displayed.
- If the feed method is called when the player is in the "started" mode, the frame shall be displayed immediately. In particular it is not required to feed a second frame to the decoder to display the first frame.

- If the feed method is called when the player is in any other state (or if the DripFeedDataSource is not connected to a player), it will be ignored by the platform implementation.

### Constructors

#### DripFeedDataSource()

```
public DripFeedDataSource()
```

Constructor. A call to the constructor will throw a security exception if the application is not granted the right to use the drip feed mode. The constructor shall automatically set the MediaLocator for this DataSource to the only allowed value: dripfeed://. There is no need for applications to later call setLocator.

### Methods

#### connect()

```
public void connect()
throws IOException
```

This method shall not be used and has no effect. This source is considered as always connected.

#### Overrides:

`connect` in class `DataSource`.

#### Throws:

`java.io.IOException` - never thrown in this sub-class.

#### disconnect()

```
public void disconnect()
```

This method shall not be used and has no effect. This source is considered as always connected.

#### Overrides:

`disconnect` in class `DataSource`

#### feed(byte[])

```
public void feed(byte[] clip_part)
```

This method allows an application to feed the decoder progressively with parts of a clip (e.g. I or P MPEG-2 frame) according to the drip-fed mode format defined in the GEM content format clause.
The feed method shall not be called more often than every 500ms. If this rule is not respected, display is not guaranteed.

While in the prefetch state the drip feed data source is only required to correctly process a single invocation of this method where the data consists only of a single I frame. Possible additional invocations while in the prefetch state shall have implementation specific results.

#### Parameters:

`clip_part` - Chunk of bytes compliant with the drip-fed mode format defined in the GEM content format clause (i.e. one MPEG-2 frame with optional synctactic MPEG-2 elements).

#### getContentType()

```
public java.lang.String getContentType()
```

This method shall return the content type for mpeg-2 video "drips".

Overrides:

`getContentType` in class `DataSource`.

Returns:

the content type for MPEG-2 video drips.

### getControl(String)

```
public java.lang.Object getControl(java.lang.String controlType)
```

Obtain the object that implements the specified Class or Interface. The full class or interface name must be used. If the control is not supported then null is returned.

Overrides:

`getControl` in class `DataSource`.

Parameters:

`controlType` - the full class or interface name of the requested control.

Returns:

the object that implements the control, or null.

### getControls()

```
public java.lang.Object[] getControls()
```

Obtain the collection of objects that control this object. If no controls are supported, a zero length array is returned.

Overrides:

`getControls` in class `DataSource`.

Returns:

the collection of object controls.

### getDuration()

```
public javax.media.Time getDuration()
```

This method shall not be used and has no effect.

Overrides:

`getDuration` in class `DataSource`

Returns:

DURATION_UNKNOWN.

### start()

```
public void start()
throws IOException
```

This method shall not be used and has no effect. This source is considered as always started.

Overrides:

`start` in class `DataSource`

Throws:

`java.io.IOException` - never thrown in this sub-class.

### stop()

```
public void stop()
throws IOException
```

This method shall not be used and has no effect. This source is considered as always started.

#### Overrides:

`stop` in class `DataSource`

#### Throws:

`java.io.IOException` - never thrown in this sub-class.

# org.dvb.media

# DripFeedPermission

### Declaration

```
public class DripFeedPermission extends java.security.BasicPermission
```

```
java.lang.Object
  |
  +--java.security.Permission
        |
        +--java.security.BasicPermission
              |
              +--org.dvb.media.DripFeedPermission
```

### All Implemented Interfaces:

`java.security.Guard`, `java.io.Serializable`

### Description

This class represents a permission to access the drip feed mode.

### Constructors

### DripFeedPermission(String)

```
public DripFeedPermission(java.lang.String name)
```

Create a new DripFeedPermission.

#### Parameters:

`name` - the name string is currently unused and should be empty.

### DripFeedPermission(String, String)

```
public DripFeedPermission(java.lang.String name, java.lang.String actions)
```

Create a new DripFeedPermission. This constructor is used by the policy class to instantiate new permission objects.

#### Parameters:

`name` - The name string is currently unused and should be empty.

`actions` - The actions string is currently unused and should be null.

### Methods

#### implies(Permission)

```
public boolean implies(java.security.Permission p)
```

Checks if the specified permission is "implied" by this object.

Since name and actions are not used, the only check needed is whether p is also a DripFeedPermission.

Overrides:

`implies` in class `BasicPermission`.

Parameters:

`p` - the permission to check against.

Returns:

true if the passed permission is equal to or implied by this permission, false otherwise.

# org.dvb.media

# DVBMediaSelectControl

### Declaration

```
public interface DVBMediaSelectControl extends javax.tv.media.MediaSelectControl
```

All Superinterfaces:

`javax.media.Control`, `javax.tv.media.MediaSelectControl`

### Description

DVBMediaSelectControl extends `MediaSelectControl` allowing the selection of different kinds of content in a running `Player`. The extension is to allow the selection in a single operation of all the media service components in a service without needing knowledge about which media service components are present in that service.

Since:

MHP 1.0.2.

See Also:

`javax.tv.media.MediaSelectControl`

### Methods

#### add(Locator, StreamType)

```
public void add(javax.tv.locator.Locator component, javax.tv.service.navigation.StreamType type)
throws InvalidLocatorException, InvalidServiceComponentException, InsufficientResourcesException, Se
curityException
```

Adds a service component (for example, subtitles) to the presentation over-riding the stream type signalled for the component. This is an asynchronous operation that is completed on receipt of a MediaSelectEvent. Components whose addition would require Player resynchronization are not permitted. If the specified service component is already part of the presentation, this method does nothing.

Parameters:

`component` - The locator representing an individual service component to add to the presentation.

`type` - The stream type of the component.

Throws:

javax.tv.locator.InvalidLocatorException - If the specified locator does not reference a selectable service component.

javax.tv.service.selection.InvalidServiceComponentException - If the addition of the service component would require resynchronization of the Player, if the service component is not part of the Service to which the MediaSelectControl is restricted, or if the service component must be presented in conjunction with another service component that is not part of the current presentation.

javax.tv.service.selection.InsufficientResourcesException - - If the operation cannot be completed due to a lack of system resources.

java.lang.SecurityException - - If the caller does not have MediaSelectPermission(component) permission.

Since:

MHP 1.1.2.

### select(Locator[], StreamType[])

```
public void select(javax.tv.locator.Locator[] components,
javax.tv.service.navigation.StreamType[] types)
throws InvalidLocatorException, InvalidServiceComponentException, InsufficientResourcesException, Se
curityException
```

Selects one or more service components for presentation over-riding the stream type signalled for the component. If some content is currently playing, it is replaced in its entirety by the specified selection. This is an asynchronous operation that is completed on receipt of a MediaSelectEvent. Note that for certain selections that imply a different time base or otherwise change synchronization relationships, a RestartingEvent will be posted by the Player. If any of the components are successfully presented then a MediaSelectSucceededEvent is generated with the locator array containing only those components that were successfully presented. A MediaSelectFailedEvent is only generated if none of the components are successfully presented.

Parameters:

components - An array of locators representing a set of individual service components to present together.

types - The stream type corresponding to each locator.

Throws:

javax.tv.locator.InvalidLocatorException - If a locator provided does not reference a selectable service component.

javax.tv.service.selection.InvalidServiceComponentException - If a specified service component is not part of the Service to which the MediaSelectControl is restricted, if a specified service component must be presented in conjunction with another service component not contained in components, if the specified set of service components cannot be presented as a coherent whole, or if the service components are not all available simultaneously.

javax.tv.service.selection.InsufficientResourcesException - If the operation cannot be completed due to a lack of system resources.

java.lang.SecurityException - If the caller does not have MediaSelectPermission(components[i]) permission for any valid i.

java.lang.IllegalArgumentException - if the two arrays are not the same size.

Since:

MHP 1.1.2.

### select(Locator, StreamType)

```
public void select(javax.tv.locator.Locator component, javax.tv.service.navigation.StreamType type)
throws InvalidLocatorException, InvalidServiceComponentException, InsufficientResourcesException, Se
curityException
```

Selects a new service component for presentation over-riding the stream type signalled for the component. If some content is currently playing, it is replaced in its entirety by the specified selection. This is an asynchronous operation that is completed upon receipt of a MediaSelectEvent. Note that for certain selections that imply a different time base or otherwise change synchronization relationships, a RestartingEvent will be posted by the Player.

Parameters:

`component` - A locator representing an individual service component to present.

`type` - The stream type of the component.

Throws:

`javax.tv.locator.InvalidLocatorException` - If the locator does not reference a selectable service component.

`javax.tv.service.selection.InvalidServiceComponentException` - If the specified service component is not part of the Service to which the MediaSelectControl is restricted, or if it cannot be presented alone.

`javax.tv.service.selection.InsufficientResourcesException` - If the operation cannot be completed due to a lack of system resources.

`java.lang.SecurityException` - If the caller does not have MediaSelectPermission(component) permission.

Since:

MHP 1.1.2.

### selectServiceMediaComponents(Locator)

```
public void selectServiceMediaComponents(javax.tv.locator.Locator l)
throws InvalidLocatorException, InvalidServiceComponentException, InsufficientResourcesException
```

Selects for presentation the media service components from a service. If some content is currently playing, it is replaced in its entirety by the media service components from the specified service. This is an asynchronous operation that is completed upon receipt of a MediaSelectEvent. Note that for most selections that imply a different time base or otherwise change synchronization relationships, a RestartingEvent will be posted by the Player. The rules for deciding which media service components shall be presented are defined in the main body of the present document.

Parameters:

`l` - the locator for a service.

Throws:

`javax.tv.locator.InvalidLocatorException` - If the locator provided does not reference a service.

`javax.tv.service.selection.InvalidServiceComponentException` - If the locator provided does not reference a service which contains at least one media service component

`javax.tv.service.selection.InsufficientResourcesException` - If the operation cannot be completed due to a lack of system resources.

# org.dvb.media
# NoComponentSelectedEvent

### Declaration

```
public class NoComponentSelectedEvent extends javax.media.StopEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--javax.media.ControllerEvent
              |
              +--javax.media.TransitionEvent
                    |
                    +--javax.media.StopEvent
                          |
                          +--org.dvb.media.NoComponentSelectedEvent
```

### All Implemented Interfaces:

`javax.media.MediaEvent`, `java.io.Serializable`

### Description

This event is generated whenever presentation of a stream stops because there are no selected components to present. One example of this would be use of the `javax.tv.media.MediaSelectControl.remove` method to remove all components of a service. Generation of this event informs the application that the Player is no longer presenting any content.

### Since:

MHP 1.0.1.

### Constructors

### NoComponentSelectedEvent(Controller, int, int, int, MediaLocator)

```
public NoComponentSelectedEvent(javax.media.Controller source, int previous, int current,
int target, javax.media.MediaLocator stream)
```

Construct an event.

### Parameters:

`source` - the controller which was presenting the service.

`stream` - the locator of the stream whose presentation has stopped.

`previous` - the previous state of the controller.

`current` - the current state of the controller.

`target` - the target state of the controller.

### Methods

### getStream()

```
public javax.media.MediaLocator getStream()
```

This method returns the stream whose presentation has stopped

### Returns:

the locator for the stream concerned.

# org.dvb.media PresentationChangedEvent

### Declaration

```
public class PresentationChangedEvent extends javax.media.ControllerEvent

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--javax.media.ControllerEvent
              |
              +--org.dvb.media.PresentationChangedEvent
```

### All Implemented Interfaces:

`javax.media.MediaEvent, java.io.Serializable`

### Description

This event is generated whenever the content being presented by a player changes for reasons outside the control of the application. The state of the player does not change - only the content being presented.

### Fields

#### CA_FAILURE

```
public static final int CA_FAILURE
```

Presentation changed due an action by the CA subsystem. Alternate content is being played, not the content selected by the user (e.g. adverts in place of a scrambled service).

#### See Also:

`getReason()`

#### CA_RETURNED

```
public static final int CA_RETURNED
```

Presentation changed due to an action by the CA subsystem. Normal content is now being presented as requested by the user. This reason code is used when the CA subsystem commands the GEM terminal to switch back to the normal presentation after having previously selected an alternate content.

#### See Also:

`getReason()`

#### STREAM_UNAVAILABLE

```
public static final int STREAM_UNAVAILABLE
```

The stream being presented is no longer available in the transport stream.

#### See Also:

`getReason()`

### Constructors

#### PresentationChangedEvent(Controller, MediaLocator, int)

```
public PresentationChangedEvent(javax.media.Controller source, javax.media.MediaLocator stream,
int reason)
```

Constructor for the event.

Parameters:

`source` - the controller whose presentation changed.

`stream` - the stream now being presented.

`reason` - the reason for the change encoded as one of the constants in this class.

Methods

getReason()

```
public int getReason()
```

This method returns the reason why access has been withdrawn.

Returns:

the reason for the change specified when the event was constructed.

getStream()

```
public javax.media.MediaLocator getStream()
```

This method returns the locator for the stream now being presented.

Returns:

the locator for the stream now being presented.

# org.dvb.media

# ServiceRemovedEvent

Declaration

```
public class ServiceRemovedEvent extends javax.media.StopEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--javax.media.ControllerEvent
              |
              +--javax.media.TransitionEvent
                    |
                    +--javax.media.StopEvent
                          |
                          +--org.dvb.media.ServiceRemovedEvent
```

All Implemented Interfaces:

`javax.media.MediaEvent`, `java.io.Serializable`

Description

This event is generated whenever access to a service stops because the service concerned has been removed from the network. Generation of this event informs the application that the Player is no longer presenting any content.

Since:

MHP 1.0.1.

### Constructors

#### ServiceRemovedEvent(Controller)

```
public ServiceRemovedEvent(javax.media.Controller source)
```

Construct an event.

Parameters:

`source` - the controller which was presenting the service.

#### ServiceRemovedEvent(Controller, int, int, int, MediaLocator)

```
public ServiceRemovedEvent(javax.media.Controller source, int previous, int current, int target,
javax.media.MediaLocator stream)
```

Construct an event.

Parameters:

`source` - the controller which was presenting the service.

`stream` - the locator of the stream which was removed from the network.

`previous` - the previous state of the controller.

`current` - the current state of the controller.

`target` - the target state of the controller.

### Methods

#### getStream()

```
public javax.media.MediaLocator getStream()
```

This method returns the stream which was removed from the network.

Returns:

the stream concerned.

# org.dvb.media

# StopByResourceLossEvent

### Declaration

```
public class StopByResourceLossEvent extends javax.media.StopEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--javax.media.ControllerEvent
              |
              +--javax.media.TransitionEvent
                    |
                    +--javax.media.StopEvent
                          |
                          +--org.dvb.media.StopByResourceLossEvent
```

### All Implemented Interfaces:

```
javax.media.MediaEvent, java.io.Serializable
```

### Description

This event is generated whenever presentation of a stream stops because the player has lost so many resources that it cannot continue. Generation of this event informs the application that the Player is no longer presenting any content.

### Since:

MHP 1.0.1.

### Constructors

#### StopByResourceLossEvent(Controller, int, int, int, MediaLocator)

```
public StopByResourceLossEvent(javax.media.Controller source, int previous, int current, int target,
javax.media.MediaLocator stream)
```

Construct an event.

### Parameters:

`source` - the controller which was presenting the service.

`stream` - the locator of the stream which was being presented.

`previous` - the previous state of the controller.

`current` - the current state of the controller.

`target` - the target state of the controller.

### Methods

#### getStream()

```
public javax.media.MediaLocator getStream()
```

This method returns the stream which was being presented.

### Returns:

the locator for the stream concerned.

# org.dvb.media

# SubtitleAvailableEvent

### Declaration

```
public class SubtitleAvailableEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.media.SubtitleAvailableEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Report that subtitles are available to be presented having been unavailable. This event is not generated on service selection or other forms of 'zapping'. Its generation is restricted to changes in the composition of the subtitle aspects of the same broadcast stream.

### Constructors

#### SubtitleAvailableEvent(Object)

```
public SubtitleAvailableEvent(java.lang.Object source)
```

Constructor.

Parameters:

`source` - the source of the event. The platform shall always pass in the JMF player presenting the subtitles.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Return the JMF player which is the source of the event.

Overrides:

`getSource` in class `EventObject`.

Returns:

the source of the event. This shall be the JMF player passed in to the constructor of the event.

# org.dvb.media

# SubtitleListener

### Declaration

```
public interface SubtitleListener extends java.util.EventListener
```

### All Superinterfaces:

```
java.util.EventListener
```

### Description

Report that a subtitle event has happened.

### Methods

#### subtitleStatusChanged(EventObject)

```
public void subtitleStatusChanged(java.util.EventObject event)
```

Report a subtitle event has happened.

Parameters:

`event` - the event which happened.

# org.dvb.media SubtitleNotAvailableEvent

### Declaration

```
public class SubtitleNotAvailableEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.media.SubtitleNotAvailableEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Inform an application that a subtitle stream has vanished from the network. This event is not generated on service selection or other forms of 'zapping'. Its generation is restricted to changes in the composition of the subtitle aspects of the same broadcast stream.

### Constructors

#### SubtitleNotAvailableEvent(Object)

```
public SubtitleNotAvailableEvent(java.lang.Object source)
```

Constructor.

#### Parameters:

`source` - the source of the event. The platform shall always pass in the JMF player presenting the subtitles.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Return the source of the event.

#### Overrides:

`getSource` in class `EventObject`.

#### Returns:

the source of the event. This shall be the JMF player passed in to the constructor of the event.

# org.dvb.media SubtitleNotSelectedEvent

### Declaration

```
public class SubtitleNotSelectedEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.media.SubtitleNotSelectedEvent
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

Report that subtitles are not now selected. Even if subtitles are available in the network, they will not be presented. This event is generated when the combination of end user control of subtitles through the navigator and application control of subtitles through `SubtitlingLanguageControl.setSubtitling` changes whether subtitles are to be presented if they are available. It is not generated for changes in the underlying availability of subtitles even if those cause changes in whether subtitles are presented or not.

Constructors

SubtitleNotSelectedEvent(Object)

```
public SubtitleNotSelectedEvent(java.lang.Object source)
```

Constructor.

Parameters:

`source` - the source of the event. The platform shall always pass in the JMF player presenting the subtitles.

Methods

getSource()

```
public java.lang.Object getSource()
```

Return the source of the event

Overrides:

```
getSource in class EventObject
```

Returns:

the source of the event. This shall be the JMF player passed in to the constructor of the event.

# org.dvb.media

# SubtitleSelectedEvent

Declaration

```
public class SubtitleSelectedEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.media.SubtitleSelectedEvent
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

Report that subtitles are now selected. If subtitles are also available then they will be presented. This event is generated when the combination of end user control of subtitles through the navigator and application control of subtitles through `SubtitlingLanguageControl.setSubtitling` changes whether subtitles are to be presented if they are available. It is not generated for changes in the underlying availability of subtitles even if those cause changes in whether subtitles are presented or not.

### Constructors

#### SubtitleSelectedEvent(Object)

```
public SubtitleSelectedEvent(java.lang.Object source)
```

Constructor.

Parameters:

`source` - the source of the event. The platform shall always pass in the JMF player presenting the subtitles.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Return the source of the event.

Overrides:

`getSource in class EventObject`

Returns:

the source of the event. This shall be the JMF player passed in to the constructor of the event.

# org.dvb.media SubtitlingEventControl

### Declaration

```
public interface SubtitlingEventControl extends org.davic.media.SubtitlingLanguageControl
```

All Superinterfaces:

`javax.media.Control, org.davic.media.LanguageControl, org.davic.media.SubtitlingLanguageControl`

### Description

Allow applications to register and unregister their interest in events related to the availability and presentation of subtitles.

### Methods

#### addSubtitleListener(SubtitleListener)

```
public void addSubtitleListener(org.dvb.media.SubtitleListener l)
```

Add a listener for subtitle events.

Parameters:

`l` - the listener to report the events to.

#### removeSubtitleListener(SubtitleListener)

```
public void removeSubtitleListener(org.dvb.media.SubtitleListener l)
```

Remove a listener for subtitle events.

Parameters:

`l` - the listener to remove.

# org.dvb.media
# VideoFormatControl

### Declaration

```
public interface VideoFormatControl extends javax.media.Control
```

### All Superinterfaces:

```
javax.media.Control
```

### Description

This provides a means for applications to get information associated with the format and aspect ratio of the video being presented to the user. This control will only be available for Players presenting MPEG-2 video streams.

It is important to note that due to different video and display formats (and user preferences), not all of the full video frame may be displayed. Similarly, it may not always be possible to map video and graphics with perfect accuracy.

Signalling for the active format description shall be supported as defined in annex B of TS 101 154 [2].

### Fields

#### AFD_14_9

```
public static final int AFD_14_9
```

Constant representing an MPEG active format description of 14:9 (centre).

#### AFD_14_9_TOP

```
public static final int AFD_14_9_TOP
```

Constant representing an MPEG active format description of 14:9 (top).

#### AFD_16_9

```
public static final int AFD_16_9
```

Constant representing an MPEG active format description of 16:9 (centre).

#### AFD_16_9_SP_14_9

```
public static final int AFD_16_9_SP_14_9
```

Constant representing an MPEG active format description of 16:9 (with shoot and protect 14:9 centre).

#### AFD_16_9_SP_4_3

```
public static final int AFD_16_9_SP_4_3
```

Constant representing an MPEG active format description of 16:9 (with shoot and protect 4:3 centre).

#### AFD_16_9_TOP

```
public static final int AFD_16_9_TOP
```

Constant representing an MPEG active format description of 16:9 (top).

#### AFD_4_3

```
public static final int AFD_4_3
```

Constant representing an MPEG active format description of 4:3 (centre).

### AFD_4_3_SP_14_9

```
public static final int AFD_4_3_SP_14_9
```

Constant representing an MPEG active format description of 4:3 (with shoot & protect 14:9 centre).

### AFD_GT_16_9

```
public static final int AFD_GT_16_9
```

Constant representing an MPEG active format description of greater than 16:9 (centre).

### AFD_NOT_PRESENT

```
public static final int AFD_NOT_PRESENT
```

Constant showing an MPEG active format description is not present.

### AFD_SAME

```
public static final int AFD_SAME
```

Constant representing an MPEG active format description that is the same as the coded frame.

### ASPECT_RATIO_16_9

```
public static final int ASPECT_RATIO_16_9
```

Constant representing an aspect ratio of 16:9.

### ASPECT_RATIO_2_21_1

```
public static final int ASPECT_RATIO_2_21_1
```

Constant representing an aspect ratio of 2.21:1.

### ASPECT_RATIO_4_3

```
public static final int ASPECT_RATIO_4_3
```

Constant representing an aspect ratio of 4:3.

### ASPECT_RATIO_UNKNOWN

```
public static final int ASPECT_RATIO_UNKNOWN
```

Constant representing an unknown aspect ratio.

### DAR_16_9

```
public static final int DAR_16_9
```

Constant representing a display aspect ratio of 16:9.

### DAR_4_3

```
public static final int DAR_4_3
```

Constant representing a display aspect ratio of 4:3.

### DFC_PLATFORM

```
public static final int DFC_PLATFORM
```

Control over the decoder format conversions is returned to being managed by the platform. This is the same as the value used if no GEM application has set a video transformation. It is not required to correspond to a single decoder format conversion and may change over time as the video input format and signalling change. This constant can only be used to set the decoder format conversion. Reading the decoder format conversion shall always return the DFC used at the time concerned.

### DFC_PROCESSING_16_9_ZOOM

```
public static final int DFC_PROCESSING_16_9_ZOOM
```

The central 16:9 letterbox area of the 4:3 720 x 576 input grid is expanded to fill the 16:9 output frame.

### DFC_PROCESSING_CCO

```
public static final int DFC_PROCESSING_CCO
```

A 4:3 central part out of the 720 x 576 input 16:9 frame is transferred into a 720 x 576 4:3 output frame.

### DFC_PROCESSING_FULL

```
public static final int DFC_PROCESSING_FULL
```

The full 720 x 576 frame is transferred (this may be either 4:3 or 16:9; part of this may be black, e.g. in the "pillar box" cases).

### DFC_PROCESSING_LB_14_9

```
public static final int DFC_PROCESSING_LB_14_9
```

The 720 x 576 input grid is transferred into a 14:9 LB in a 4:3 frame.

### DFC_PROCESSING_LB_16_9

```
public static final int DFC_PROCESSING_LB_16_9
```

The 720 x 576 input grid is transferred into a 16:9 letterbox in a 4:3 frame.

### DFC_PROCESSING_LB_2_21_1_ON_16_9

```
public static final int DFC_PROCESSING_LB_2_21_1_ON_16_9
```

The 720 x 576 input grid is transferred into a 2.21:1 letterbox in a 16:9 frame.

### DFC_PROCESSING_LB_2_21_1_ON_4_3

```
public static final int DFC_PROCESSING_LB_2_21_1_ON_4_3
```

The 720 x 576 input grid is transferred into a 2.21:1 letterbox in a 4:3 frame.

### DFC_PROCESSING_NONE

```
public static final int DFC_PROCESSING_NONE
```

Decoder format conversion is inactive.

### DFC_PROCESSING_PAN_SCAN

```
public static final int DFC_PROCESSING_PAN_SCAN
```

A 4:3 part out of the 720 x 576 input 16:9 or 2.21:1 frame is transferred into a 720 x 576 4:3 output frame. The horizontal position of this part is determined by pan&scan vectors from the MPEG video stream.

### DFC_PROCESSING_UNKNOWN

```
public static final int DFC_PROCESSING_UNKNOWN
```

Constant representing an unknown format conversion being performed by the decoder.

## Methods

### addVideoFormatListener(VideoFormatListener)

```
public void addVideoFormatListener(org.dvb.media.VideoFormatListener l)
```

Add a listener for VideoFormatChangedEvents.

Parameters:

l - the listener to add.

getActiveFormatDefinition()

```
public int getActiveFormatDefinition()
```

Return the value of the active_format field of the MPEG Active Format Description of the video if it is transmitted (one of the constants AFD_* above). If this field is not available then AFD_NOT_PRESENT is returned. The constant values for the constants representing the active format description should be identical to the values specified in ETR154, annex B.

Returns:

the value of the active_format field of the MPEG Active Format Description of the video if it is transmitted. If this field is not available, or the video is not MPEG, then AFD_NOT_PRESENT is returned.

getAspectRatio()

```
public int getAspectRatio()
```

Return the aspect ratio of the video as it is transmitted. If the aspect ratio is not known, ASPECT_RATIO_UNKNOWN is returned.

Returns:

the aspect ratio of the video.

getDecoderFormatConversion()

```
public int getDecoderFormatConversion()
```

Return a value representing what format conversion is being done by the decoder in the platform (one of the constants DFC_* above). A receiver may implement only a subset of the available options. This decoder format conversion may be active or not depending upon the mode of operation.

Returns:

the decoder format conversion being performed or DFC_PROCESSING_UNKNOWN if this is not known.

getDisplayAspectRatio()

```
public int getDisplayAspectRatio()
```

Return the aspect ratio of the display device connected to this GEM decoder (one of the constants DAR_* above).

Returns:

the aspect ratio of the display device connected to the decoder.

getVideoTransformation(int)

```
public org.dvb.media.VideoTransformation getVideoTransformation(int dfc)
```

This method returns a VideoTransformation object that corresponds with the specified Decoder Format Conversion when applied to the currently selected video. If the specified Decoder Format Conversion is not supported for the currently selected video, then this method returns null.

Parameters:

dfc - the Decoder Format Conversion (one of the DFC_* constants specified in this interface).

Returns:

the video transformation, or null if the specified Decoder Format Conversion is not supported for the currently selected video.

### isPlatform()

```
public boolean isPlatform()
```

Test if control over the decoder format conversions is being managed by the platform as defined by `DFC_PLATFORM`.

#### Returns:

true if control over the decoder format conversions is being managed by the platform, false otherwise.

#### See Also:

`DFC_PLATFORM`

### removeVideoFormatListener(VideoFormatListener)

```
public void removeVideoFormatListener(org.dvb.media.VideoFormatListener l)
```

Remove a listener for VideoFormatChangedEvents.

#### Parameters:

`l` - the listener to remove.

# org.dvb.media

# VideoFormatEvent

#### Declaration

```
public abstract class VideoFormatEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.media.VideoFormatEvent
```

#### All Implemented Interfaces:

```
java.io.Serializable
```

#### Direct Known Subclasses:

```
ActiveFormatDescriptionChangedEvent, AspectRatioChangedEvent, DFCChangedEvent
```

#### Description

The base class for all other events relating to changes in video format.

#### Constructors

### VideoFormatEvent(Object)

```
public VideoFormatEvent(java.lang.Object source)
```

Constructor.

#### Parameters:

`source` - the source of the event. The platform shall always pass in the JMF Player presenting the video whose format changed.

# org.dvb.media VideoFormatListener

### Declaration

```
public interface VideoFormatListener extends java.util.EventListener
```

### All Superinterfaces:

```
java.util.EventListener
```

### Description

The listener used to receive video format events.

### Methods

#### receiveVideoFormatEvent(VideoFormatEvent)

```
public void receiveVideoFormatEvent(org.dvb.media.VideoFormatEvent anEvent)
```

receive a VideoFormatEvent.

#### Parameters:

`anEvent` - the VideoFormatEvent that has been received.

# org.dvb.media VideoPresentationControl

### Declaration

```
public interface VideoPresentationControl extends javax.media.Control
```

### All Superinterfaces:

```
javax.media.Control
```

### All Known Subinterfaces:

```
BackgroundVideoPresentationControl
```

### Description

A control to support setting and querying the video presentation.

Note: For a component-based player the scaling and positioning of the video is done by manipulating the corresponding AWT component. The VideoPresentationControl only allows for the setting of the clipping region.

Note: If the hardware supports the positioning of interlaced video on even lines only (when counting from 0), then a component-based player is allowed to position the top of the video one line below where it should be.

For a background player there is the BackgroundVideoPresentationControl that allows for the setting of the clipping region, the position and the scaling of the video in one atomic action.

### Fields

#### POS_CAP_FULL

```
public static final byte POS_CAP_FULL
```

Constant representing that the video can be positioned anywhere on the screen, even if a part of the video is off screen as a result of that.

#### POS_CAP_FULL_EVEN_LINES

```
public static final byte POS_CAP_FULL_EVEN_LINES
```

n Constant representing that the video can be positioned anywhere on the screen, even if a part of the video is off screen as a result of that, with the restriction that the field order is respected. This implies that interlaced video can be positioned on even lines only (when counting from 0).

#### POS_CAP_FULL_EVEN_LINES_IF_ENTIRE_VIDEO_ON_SCREEN

```
public static final byte POS_CAP_FULL_EVEN_LINES_IF_ENTIRE_VIDEO_ON_SCREEN
```

Constant representing that the video can be positioned anywhere on screen as long as all the video is on screen, with the restriction that the field order is respected. This implies that interlaced video can be positioned on even lines only (when counting from 0).

#### POS_CAP_FULL_IF_ENTIRE_VIDEO_ON_SCREEN

```
public static final byte POS_CAP_FULL_IF_ENTIRE_VIDEO_ON_SCREEN
```

Constant representing that the video can be positioned anywhere on screen as long as all the video is on screen.

#### POS_CAP_OTHER

```
public static final byte POS_CAP_OTHER
```

Constant representing that the video positioning capability cannot be expressed by another POS_CAP_* constant.

### Methods

#### getActiveVideoArea()

```
public org.havi.ui.HScreenRectangle getActiveVideoArea()
```

This method returns the size and location of the active video area. The active video area excludes any "bars" used for letterboxing or pillarboxing that the receiver knows about. Bars that are included in the broadcast stream and not signalled by active format descriptors are included in the active video area. The active video area may be larger/smaller than the screen, and may possibly be offset. The offsets will be negative if the origin of the active video area is above/left of the top, left corner of the screen. In case of pan&scan, the value returned may vary over time. This method only describes the relationship between the active video and the screen. It does not describe which portion of the screen is displaying the video.

NOTE: This method includes any video scaling.

Returns:

an HScreenRectangle representing the active video area in the normalized coordinate space.

#### getActiveVideoAreaOnScreen()

```
public org.havi.ui.HScreenRectangle getActiveVideoAreaOnScreen()
```

This method returns the size and location of the active video area on-screen. The active video area excludes any "bars" used for letterboxing or pillarboxing that the receiver knows about. Bars that are included in the broadcast stream and not signalled by active format descriptors are included in the active video area. The active video area on-screen may be smaller than the area of the screen, and may possibly be offset a positive amount. This method only describes the area on-screen where active video is being presented. It does not really describe which part of the video is being shown on-screen. This is especially true for pan&scan.

NOTE: This method includes any video scaling.

Returns:

an HScreenRectangle representing the active video area on-screen in the normalized coordinate space.

### getClipRegion()

```
public java.awt.Rectangle getClipRegion()
```

This method returns the area of the decoded video that will be displayed. If clipping is not supported, the dimensions of the bounding box will be the same as the displayed video. Note that when the GEM terminal is in `pan & scan mode`, the return value of this method will be out of date almost as soon as the method has returned.

Returns:

area of the decoded video that will be displayed. The coordinate space used to express the region is that of the decoded video after possible ETR154 up-sampling.

### getHorizontalScalingFactors()

```
public float[] getHorizontalScalingFactors()
```

This method gives information about the supported discrete horizontal scaling factors in case arbitrary horizontal scaling is not supported.

Returns:

an array with the supported discrete horizontal scaling factors (including the scaling factor 1), sorted in ascending order. null is returned when arbitrary horizontal scaling is supported.

### getInputVideoSize()

```
public java.awt.Dimension getInputVideoSize()
```

This method returns the dimensions of the video before any scaling has taken place (but after TS 101 154 [2] up-sampling). On 50 Hz standard definition systems this method always returns 720 x 576.

Returns:

the size of the decoded video before any scaling has taken place (but after TS 101 154 [2] up-sampling).

### getPositioningCapability()

```
public byte getPositioningCapability()
```

This method gives information about how the video can be positioned on screen.

Returns:

the positioning capability for the currently selected video as one of the POS_CAP_* constants.

### getTotalVideoArea()

```
public org.havi.ui.HScreenRectangle getTotalVideoArea()
```

This method returns a relative size and location of the total video area, including any "bars" used for letterboxing or pillarboxing that are included in the broadcast stream, but excluding any "bars" introduced as a result of video filtering. This may be larger or smaller than the size of the physical display device. This method only describes the relationship between the total video and the screen. It does not describe which portion of the screen is displaying the video.

NOTE: This method includes any video scaling.

Returns:

an HScreenRectangle representing the total video area in the normalized coordinate space.

### getTotalVideoAreaOnScreen()

```
public org.havi.ui.HScreenRectangle getTotalVideoAreaOnScreen()
```

This method returns a relative size and location of the total video area on-screen, including any "bars" used for letterboxing or pillarboxing that are included in the broadcast stream, but excluding any "bars" introduced as a result of video filtering. This method only describes the area on-screen where total video is being presented. This does not really describe which part of the video is being shown on-screen. This is especially true for pan&scan.

NOTE: This method includes any video scaling.

#### Returns:

an HScreenRectangle representing the total video area on-screen in the normalized coordinate space.

### getVerticalScalingFactors()

```
public float[] getVerticalScalingFactors()
```

This method gives information about the supported discrete vertical scaling factors in case arbitrary vertical scaling is not supported.

#### Returns:

an array with the supported discrete vertical scaling factors (including the scaling factor 1), sorted in ascending order. null is returned when arbitrary vertical scaling is supported.

### getVideoSize()

```
public java.awt.Dimension getVideoSize()
```

This method returns the size of the decoded video as it is being presented to the user. It takes scaling and clipping into account.

#### Returns:

the size of the decoded video as it is being presented to the user.

### setClipRegion(Rectangle)

```
public java.awt.Rectangle setClipRegion(java.awt.Rectangle clipRect)
```

Set the region of the decoded video that will be displayed. If clipping is not supported, this method has no effect. If the bounding box extends beyond the decoded video, the results are implementation dependent. By default, the clipping region is set to the dimensions of the decoded video. This method returns the bounding box of the clipping region that was actually set. Implementations may approximate the requested rectangle if they have restrictions on video clipping.

If the player is a component-based player (as opposed to a background player), then the top left corner of the clip region will be aligned with the top left corner of the java.awt.Component returned by the method javax.media.Player.get-VisualComponent(). Hence changing the position of the clip region within the video moves the video with respect to the coordinate space used by java.awt.

#### Parameters:

`clipRect` - the bounding box of the clipping region. The coordinate space used to express the region is that of the decoded video after possible TS 101 154 [2] up-sampling.

#### Returns:

the set clipping region. If the requested clipping region is supported exactly, then the input parameter clipRect is returned, otherwise a newly created object will be returned.

### supportsArbitraryHorizontalScaling()

```
public float[] supportsArbitraryHorizontalScaling()
```

This method gives information about whether arbitrary horizontal scaling is supported for the currently playing video. If arbitrary horizontal scaling is supported, then an array with two elements in returned. The first element returns the smallest allowed scaling factor (e.g. 0,5) and the second element returns the largest allowed scaling factor (e.g. 4). If arbitrary horizontal scaling is not supported, null is returned. In that case the method getHorizontalScalingFactors can be used to query which discrete scaling factors are supported.

#### Returns:

an array with the minimum and maximum allowed horizontal scaling factor, or null if arbitrary horizontal scaling is not supported.

### supportsArbitraryVerticalScaling()

```
public float[] supportsArbitraryVerticalScaling()
```

This method gives information about whether arbitrary vertical scaling is supported for the currently playing video. If arbitrary vertical scaling is supported, then an array with two elements in returned. The first element returns the smallest allowed scaling factor (e.g. 0,5) and the second element returns the largest allowed scaling factor (e.g. 2). If arbitrary vertical scaling is not supported, null is returned. In that case the method getVerticalScalingFactors can be used to query which discrete scaling factors are supported.

#### Returns:

an array with the minimum and maximum allowed vertical scaling factor, or null if arbitrary vertical scaling is not supported.

### supportsClipping()

```
public boolean supportsClipping()
```

Test if the decoder supports clipping.

#### Returns:

true if and only if the decoder supports clipping.

# org.dvb.media

# VideoTransformation

#### Declaration

```
public class VideoTransformation

java.lang.Object
  |
  +--org.dvb.media.VideoTransformation
```

#### Description

VideoTransformation objects express video transformations, i.e. the clipping, the horizontal and vertical scaling and the position of the video. All transformations are to be applied after possible TS 101 154 [2] up-sampling.

NOTE: Instances of VideoTransformation can represent pan and scan, but an application cannot create such instances itself. An application can get a VideoTransformation representing pan and scan, by calling the VideoFormatControl.getVideoTransformation() method with the pan and scan Decoder Format Conversion constant.

### Constructors

#### VideoTransformation()

```
public VideoTransformation()
```

Creates a VideoTransformation object with default parameters. Clipping is disabled, both the horizontal and the vertical scaling factors are 1, and the video position is (0,0) in the normalised coordinate space.

#### VideoTransformation(Rectangle, float, float, HScreenPoint)

```
public VideoTransformation(java.awt.Rectangle clipRect, float horizontalScalingFactor,
float verticalScalingFactor, org.havi.ui.HScreenPoint location)
```

Creates a VideoTransformation object with the supplied parameters.

Parameters:

`clipRect` - the bounding box of the clipping region. The coordinate space used to express the region is that of the decoded video after possible ETR154 up-sampling. A non-null ClipRect enables clipping. A null ClipRect disables it.

`horizontalScalingFactor` - the horizontal scaling factor.

`verticalScalingFactor` - the vertical scaling factor.

`location` - the location of the video on the screen in the normalised coordinate space.

### Methods

#### getClipRegion()

```
public java.awt.Rectangle getClipRegion()
```

Gets the clipping region.

Returns:

the bounding box of the clipping region. The coordinate space used to express the region is that of the decoded video after possible ETR154 up-sampling. null is returned if this video transformation represents pan and scan or if clipping is disabled.

#### getScalingFactors()

```
public float[] getScalingFactors()
```

Gets the horizontal and vertical scaling factors.

Returns:

an array with two elements. The first element contains the horizontal scaling factor, the second element the vertical scaling factor.

#### getVideoPosition()

```
public org.havi.ui.HScreenPoint getVideoPosition()
```

Returns the video position.

Returns:

the location of the video on the screen in the normalised coordinate space.

#### isPanAndScan()

```
public boolean isPanAndScan()
```

Returns whether this video transformation represents pan and scan.

Returns:

true is this video transformation represents pan and scan, false otherwise.

### setClipRegion(Rectangle)

```
public void setClipRegion(java.awt.Rectangle clipRect)
```

Sets the clipping region.

If this video transformation represents pan and scan, then it will no longer represent pan and scan when this method is called. A non-null ClipRect enables clipping. A null ClipRect disables it.

Parameters:

`clipRect` - the bounding box of the clipping region. The coordinate space used to express the region is that of the decoded video after possible ETR154 up-sampling.

### setScalingFactors(float, float)

```
public void setScalingFactors(float horizontalScalingFactor, float verticalScalingFactor)
```

Sets the horizontal and vertical scaling factors.

Parameters:

`horizontalScalingFactor` - the horizontal scaling factor.

`verticalScalingFactor` - the vertical scaling factor.

### setVideoPosition(HScreenPoint)

```
public void setVideoPosition(org.havi.ui.HScreenPoint location)
```

Sets the video position.

Parameters:

`location` - the location of the video on the screen in the normalised coordinate space.

# N.2 Streaming monitoring API

# Package

# org.dvb.media.monitoring

Description

Provides DVB specific extensions to the Java Media Framework related to stream monitoring.

| Class Summary | |
|---|---|
| **Interfaces** | |
| MonitoringControl | A control allows applications to get information associated with the Streamed CoD being presented to the user. |
| MonitoringListener | The listener used to receive MonitoringEvents. |
| **Classes** | |
| BufferingStatusEvent | Event signalling a change in the buffer status. |
| MonitoringEvent | The base class for all other events related to changes in associated CoD. |
| StreamQualityChangeEvent | Event signals that quality of the associated CoD has been changed. |
| StreamQualityInfo | Represents information about quality of a CoD stream. |
| VideoStreamQualityInfo | Represents information about CoD stream's video. |

# MonitoringControl

All Superinterfaces:

javax.media.Control

```
public interface MonitoringControl extends javax.media.Control
```

A MonitoringControl allows applications to get information associated with the Streamed CoD being presented to the user.

Additionally it allows applications to control quality of CoD by setting bitrate of downloaded CoD.

## addMonitoringListener

```
void addMonitoringListener(MonitoringListener l)
```

Add a listener for MonitoringEvent.

Parameters:

l - the listener to add

## getAudioFormat

```
java.lang.String getAudioFormat()
   throws java.lang.IllegalStateException
```

Get the audio format of the associated CoD.

Formats are encoded as MPEG-7 term IDs as extended by the DVB IPI handbook, e.g. MPEG-1 layer 3 audio is encoded as "3.3". Precognitions: Information is available after prefetching.

Returns:

audio format of the associated CoD

Throws:

`java.lang.IllegalStateException` - thrown if Player is neither in Prefetched nor Started state

## getAvailableStreamQualities

`StreamQualityInfo[] **getAvailableStreamQualities**()`

`throws java.lang.IllegalStateException`

Get all available qualities of streams in associated CoD. In case of video CoD an array of `VideoStreamQualityInfo`s shall be returned. Precognitions: Information is available after prefetching

Returns:

all available qualities of associated CoD

Throws:

`java.lang.IllegalStateException` - if Player is neither in Prefetched nor in Started state

## getCurrentLineRate

`long **getCurrentLineRate**()`

Get the current speed (in bits per second) with which the bits are sent onto the wire.

Returns:

current speed with which the bits are sent onto the wire

## getStreamQuality

`StreamQualityInfo **getStreamQuality**()`

`throws java.lang.IllegalStateException`

Get the current quality of associated CoD. It is the quality at which associated CoD is being downloaded and presented to the user. In case of video CoD an instance of `VideoStreamQualityInfo` shall be returned. Precognitions: Information is available after prefetching

Returns:

quality of stream at which associated CoD is being downloaded and presented to the user

Throws:

`java.lang.IllegalStateException` - if Player is neither in Prefetched nor Started state

## getVideoFormat

`java.lang.String **getVideoFormat**()`

`throws java.lang.IllegalStateException`

Get the video format of associated CoD.

Formats are encoded as MPEG-7 term IDs as extended by the DVB IPI handbook, e.g. MPEG-2 video, main profile, main level is encoded as "2.2.2". Precognitions: Information is available after prefetching

Returns:

video format of the associated CoD or null if CoD does not have video stream

Throws:

`java.lang.IllegalStateException` - thrown if Player is neither in Prefetched nor Started state

## removeMonitoringListener

`void` **`removeMonitoringListener`**`(MonitoringListener l)`

Remove a listener for MonitoringEvent.

Parameters:

`l` - the listener to remove

## setFixedStreamQuality

`void` **`setFixedStreamQuality`**`(StreamQualityInfo streamQuality)`

`throws java.lang.IllegalArgumentException,`

`java.lang.IllegalStateException`

Set fixed quality of associated CoD. It allows application to force downloading CoD with a given video resolution and stream bitrate, and thus disable adaptive streaming support for associated CoD.

Application can check CoD's available qualities using `getAvailableStreamQualities()` method.

Parameters:

`streamQuality` - the stream quality of CoD

Throws:

`java.lang.IllegalArgumentException` - thrown if CoD does not support given quality

`java.lang.IllegalStateException` - if Player is not in the Prefetched state

## setInitialStreamQuality

`void` **`setInitialStreamQuality`**`(StreamQualityInfo streamQuality)`

`throws java.lang.IllegalArgumentException,`

`java.lang.IllegalStateException`

Set initial quality of associated CoD. After Player starts, a quality could be changed by Adaptive Streaming.

Application can check CoD's available qualities using `getAvailableStreamQualities()` method.

Parameters:

`streamQuality` - the stream quality of CoD

Throws:

`java.lang.IllegalArgumentException` - thrown if CoD does not support given quality

`java.lang.IllegalStateException` - if Player is not in the Prefetched state

# StreamQualityInfo

`java.lang.Object`

└ **`org.dvb.media.monitoring.StreamQualityInfo`**

Direct Known Subclasses:

VideoStreamQualityInfo

```
public class StreamQualityInfo extends java.lang.Object
```

Represents information about quality of a CoD stream. It contains information about stream bitrate (bits per second).

## StreamQualityInfo

```
public StreamQualityInfo(long bitrate)
```

Construct a stream quality entity.

Parameters:

`bitrate` - stream bitrate

## getStreamBitrate

```
public long getStreamBitrate()
```

Get stream's bitrate (bits per second) of a CoD.

Returns:

Bitrate of a CoD

# VideoStreamQualityInfo

```
java.lang.Object
  └ org.dvb.media.monitoring.StreamQualityInfo
      └ org.dvb.media.monitoring.VideoStreamQualityInfo
```

```
public class VideoStreamQualityInfo extends StreamQualityInfo
```

Represent information about quality of a video CoD stream. It contains information about video resolution and stream bitrate (bits per second).

Instance of this class shall be returned by `MonitoringControl.getAvailableStreamQualities()` and `MonitoringControl.getStreamQuality()` in case when video stream is playing. Additionally, in this case an instance of this class shall be passed to `MonitoringControl.setFixedStreamQuality(StreamQualityInfo)` and `MonitoringControl.setFixedStreamQuality(StreamQualityInfo)` methods.

## VideoStreamQualityInfo

```
public VideoStreamQualityInfo(java.awt.Dimension videoResolution, long bitrate)
```

Construct an stream quality entity.

Parameters:

`videoResolution` - video resolution

`bitrate` - stream bitrate in bits per second

## getVideoResolution

```
public java.awt.Dimension getVideoResolution()
```

Get video resolution of a CoD.

Returns:

video resolution of a CoD

# MonitoringListener

```
public interface MonitoringListener
```

The listener used to receive MonitoringEvent.

## receiveMonitoringEvent

```
void receiveMonitoringEvent(MonitoringEvent anEvent)
```

Receive a MonitoringEvent.

Parameters:

`anEvent` - the MonitoringEvent that has been received

# MonitoringEvent

```
java.lang.Object
  └ java.util.EventObject
      └ org.dvb.media.monitoring.MonitoringEvent
```

All Implemented Interfaces:

java.io.Serializable

Direct Known Subclasses:

BufferingStatusEvent, StreamQualityChangeEvent

```
public class MonitoringEvent extends java.util.EventObject
```

The base class for all other events related to changes in associated CoD.

## MonitoringEvent

```
protected MonitoringEvent(java.lang.Object source)
```

Construct the event.

Parameters:

`source` - the source of the event. The platform shall always pass in the JMF Player presenting the media whose property has been changed

# StreamQualityChangeEvent

```
java.lang.Object
```

```
└ java.util.EventObject
    └ org.dvb.media.monitoring.MonitoringEvent
        └ org.dvb.media.monitoring.StreamQualityChangeEvent
```

All Implemented Interfaces:

java.io.Serializable

```
public class StreamQualityChangeEvent extends MonitoringEvent
```

Event is signalling that quality of the associated CoD has been changed.

This event shall occur when adaptive streaming changes played stream to another stream with a different video resolution and/or stream's bitrate because of changing network conditions.

## StreamQualityChangeEvent

```
public StreamQualityChangeEvent(java.lang.Object source, StreamQualityInfo newStreamQuality)
```

Construct the event.

Parameters:

`source` - the source of the event. The platform shall always pass in the JMF Player presenting the media whose property has been changed

`newStreamQuality` - the new value of quality of the stream

## getStreamQuality

```
public StreamQualityInfo getStreamQuality()
```

Get the new quality of associated CoD.

Returns:

the new stream's quality

# BufferingStatusEvent

```
java.lang.Object
└ java.util.EventObject
    └ org.dvb.media.monitoring.MonitoringEvent
        └ org.dvb.media.monitoring.BufferingStatusEvent
```

All Implemented Interfaces:

java.io.Serializable

```
public class BufferingStatusEvent extends MonitoringEvent
```

Event signalling a change in the buffer status.

Events are sent when a JMF Player is in the Started state.

To prevent implementations from being overloaded BUFFERING_STATUS_UNDERRUN and BUFFERING_STATUS_BACK_TO_NORMAL events should not be sent faster than one per seconds.

# BUFFERING_STATUS_BACK_TO_NORMAL

```
public static final int BUFFERING_STATUS_BACK_TO_NORMAL
```

Indicates that buffering status back to a normal state from BUFFERING_STATUS_UNDERRUN or BUFFERING_STATUS_STOPPED states.

# BUFFERING_STATUS_STOPPED

```
public static final int BUFFERING_STATUS_STOPPED
```

Indicates that buffering has been stopped. It means that network connection has been lost.

# BUFFERING_STATUS_UNDERRUN

```
public static final int BUFFERING_STATUS_UNDERRUN
```

Indicates that buffering status is underrun. It means that coding rate is too high compared with network line rate.

# BufferingStatusEvent

```
public BufferingStatusEvent(java.lang.Object source, int bufferingStatus)
   throws java.lang.IllegalArgumentException
```

Construct the event.

Parameters:

source - the source of the event. The platform shall always pass in the JMF Player presenting the media whose property has been changed

bufferingStatus - the status of the buffer

Throws:

java.lang.IllegalArgumentException - thrown if bufferingStatus is out of range

# getBufferingStatus

```
public int getBufferingStatus()
```

Get buffering status.

Returns:

status of the buffer

# N.3 Media Stream Synchronization API

# Package

# org.dvb.media

### Description

Provides DVB specific extensions to the Java Media Framework to enable synchronization of media streams.

| Class Summary | |
|---|---|
| **Interfaces** | |
| MasterSlaveSyncLinkageControl | MasterSlaveSyncLinkageControl handles Player-based Master/Slave relationship for synchronization of JMF Players. |
| SyncControl | SyncControl is a Control object for synchronization of a slave JMF Player. |
| SyncStateChangedEventListener | This listener interface is for receiving synchronization change events among JMF Players. |
| **Classes** | |
| SyncStateChangedEvent | An event which indicates change of status of synchronization between master/slave JMF Players. |
| **Exceptions** | |
| IncompatibleSynchronizableMediaTypeException | This exception is thrown when media type of each player is incompatible for synchronization. |

## MasterSlaveSyncLinkageControl

```
package org.dvb.media
public interface MasterSlaveSyncLinkageControl
```

MasterSlaveSyncLinkageControl handles Player-based Master/Slave relationship for synchronization of JMF Players. A JMF Player that provides master clock for synchronization is a *master Player*. MasterSlaveSyncLinkageControl allows a JMF Player from which this Control is obtained to be a master Player. At the time MasterSlaveSyncLinkageControl object is obtained from a Controller object of a JMF Player, the Player is a potential master Player. When a Controller object of a *slave Player* is added this control by `addSlave()` method, actual synchronization relationship is established. If synchronization between the Players is successfully established, a JMF Player to which MasterSlaveSyncLinkageControl object belongs becomes a master Player. When all the slave Players are removed from the MasterSlaveSyncLinkageControl object by `removeSlave()` method, a master Player is back to a potential master Player.

MasterSlaveSyncLinkageControl object monitors synchronization status between the master and slave Players. When synchronization status is changed, MasterSlaveSyncLinkageControl object notifies the situation by throwing SyncStateChangedEvent to the application. Before adding a slave Player, synchronization status is 'Unsynchronized'. At the time a slave Player is registered by `addSlave()` method, synchronization status is 'Synchronizing'. When synchronization is successfully established, the status is 'Synchronized'. In case that established synchronization cannot be kept within given tolerance, the status returns to 'Synchronizing' and related control objects try to re-synchronize automatically unless synchronization status is set to *OUT_OF_SYNC_UNSYNCHRONIZABLE*.

MasterSlaveSyncLinkageControl supports only simple Master/Slave relationship. That is, the object accepts relationship between a single master Player and slave Players. It is not allowed that a single JMF Player becomes a master Player and a slave Player at the same time. (A potential master Player can be a slave Player.) In addition, a slave Player can have only a single master Player. In case that an application tries to establish the relationship that violates these conditions, *IllegalArgumentException* is thrown when `addSlave()` is called.

Instance of MasterSlaveSyncLinkageControl can be obtained from a JMF Controller via the methods getControl(String) and getControls(). A single Controller object of a JMF Player can create at most one instance of MasterSlaveSyncLinkageControl.

### Synchronization Status and Status Constants

| Unsynchronized | Synchronizing | Synchronized |
|---|---|---|
| Master/Slave relationship is not established and Players can be running freely. | *MASTERPLAYER_STOOPED*, *SLAVEPLAYER_STOPPED*, *OUT_OF_SYNC_MASTER_CLK*, *OUT_OF_SYNC_SLAVE_CLK*, *OUT_OF_SYNC_OTHERSs* | *IN_SYNC* |
| | | Unsynchronizable<br>*OUT_OF_SYNC_UNSYNCHRONIZABLE* |

## IN_SYNC

```
static final int IN_SYNC
```
Status representing synchronized

## MASTERPLAYER_STOPPED

```
static final int MASTERPLAYER_STOPPED
```
Status representing stop() method is applied to a master Player

## OUT_OF_SYNC_MASTER_CLK

```
static final int OUT_OF_SYNC_MASTER_CLK
```
Status representing out of sync due to unstable clock of master Player

## OUT_OF_SYNC_OTHERS

```
static final int OUT_OF_SYNC_OTHERS
```
Status representing out of sync due to other reasons

## OUT_OF_SYNC_SLAVE_CLK

```
static final int OUT_OF_SYNC_SLAVE_CLK
```
Status representing out of sync due to unstable clock of slave Player

## OUT_OF_SYNC_UNSYNCHRONIZABLE

```
static final int OUT_OF_SYNC_UNSYNCHRONIZABLE
```
Status representing out of sync and Unsynchronzable state

## SLAVEPLAYER_STOPPED

```
static final int SLAVEPLAYER_STOPPED
```
Status representing stop() method is applied to a slave Player

## addSlave

```
void addSlave(Controller ctrl)
          throws    InsufficientResourcesException,
                    IncompatibleSynchronizableMediaTypeException,
                    java.lang.IllegalArgumentException
```

Adds a specified Player as a slave Player that synchronizes with a master Player. Offset value between master Player and a specified Player is implicitly given as zero in this method. If the specified Player already establishes synchronization, this method does nothing.

Parameters:

`ctrl` - Controller object of a potential slave Player to synchronize with this master Player.

Throws:

`InsufficientResourcesException` - If the operation cannot be completed due to a lack of system resources.

`IncompatibleSynchronizableMediaTypeException` - If media type Player handles is incompatible to establish synchronization.

`java.lang.IllegalArgumentException` - If a specified Controller object already forms a master or slave Player.

## addSlave

```
void addSlave(Controller ctrl, Time t)
          throws    InsufficientResourcesException,
                    IncompatibleSynchronizableMediaTypeException,
                    java.lang.IllegalArgumentException
```

Adds a specified Player as a slave Player that synchronizes with a master Player. Offset value given as a Time object states offset of time position of the specified slave Player and the master Player. If the specified Player already establishes synchronization, this method does nothing.

Parameters:

`ctrl` - Controller object of a potential slave Player to synchronize with this master Player.

`t` - Time object to represent offset between master Player's clock and the specified Player.

Throws:

`InsufficientResourcesException` - If the operation cannot be completed due to a lack of system resources.

`IncompatibleSynchronizableMediaTypeException` - If media type Player handles is incompatible to establish synchronization.

`java.lang.IllegalArgumentException` - If a specified Controller object already forms a master or slave Player.

## addSyncStateChangedEventListener

```
void addSyncStateChangedEventListener(SyncStateChangedEventListener l)
```
Adds the specified SyncStateChangedEventListener to receive the events. SyncStateChangedEvent occurs when synchronization status between the master Player and any slave Player is changed. If l is null, no exception is thrown and no action is performed.

Parameters:
l - SyncStateChangedEvent listener.

## getSlaves

Controller[] **getSlaves**()
Gets slave Players added by addSlave method. Actual synchronization status of each slave Player is not considered in return value of this method.
Returns:
An array of Controller objects which are added by addSlave method.

## getSyncStateChangedEventListeners

SyncStateChangedEventListener[] **getSyncStateChangedEventListeners**()
Returns an array of all the SyncStateChangedEvent listeners registered on this control.
Returns:
An array of registered SyncStateChangedEvent listeners.

## getSyncStatus

int **getSyncStatus**(Controller ctrl)
Gets currrent synchronization status of s specified slave Player.
Parameters:
ctrl - Controller objcet of a slave Player object to obtain current synchronization status.
Returns:
Synchronization status in field values.

## removeSlave

void **removeSlave**(Controller ctrl)
throws java.lang.IllegalArgumentException
Removes a specified slave Player. The specified Player turns self-running. This removal should not affect any progress of a master Player.
Parameters:
ctrl - Controller object of a potential slave Player to synchronize with this master Player.
Throws:
java.lang.IllegalArgumentException - If the specified Controller object is not registered as a slave Player.

## removeSyncStateChangedEventListener

void **removeSyncStateChangedEventListener**(SyncStateChangedEventListener l)
Removes a specified SyncStateChangedEventListener to receive the events. If l is null, no exception is thrown and no action is performed.
Parameters:
l - SyncStateChangedEvent listener.

# SyncControl

package org.dvb.media
public interface **SyncControl**

SyncControl is a Control object for synchronization of a slave JMF Player. SyncControl manipulates offset in time with a master Player and allowable torelance of synchronization. When SyncControl object is obtained from a Controller object of a JMF Player by getControl(String) or getControls() methods, the JMF Player is a potential slave Player. Such a JMF Player becomes a slave Player when the Player is registered as a slave Player by addSlave() methods of MasterSlaveSyncLinkageControl.

SyncControl supports only simple Master/Slave relationship. That is, the object accepts relationship between a single master Player and slave Players. It is not allowed that a single JMF Player becomes a master Player and a slave Player at the same time. In addition, a slave Player can have only a single master Player.

Instance of SyncControl can be obtained from a JMF Player via the methods getControl(String) and getControls(). A single JMF Player can create at most one instance of SyncControl.

## getMaster

MasterSlaveSyncLinkageControl **getMaster**()
Gets a MasterSlaveSinkLinkageControl object that this SyncControl is currently synchronizing with.
Returns:
MasterSlaveSinkLinkageControl object that this SyncControl is currently synchronizing with.

## getOffset

`Time` **`getOffset`**`()`
Gets offset value of a specified slave Player against master Player, by Time object.
Returns:
Offset value in terms of Time object.

## getTolerance

`Time` **`getTolerance`**`()`
Gets assigned tolerance value of synchronization for a specified slave Player.
Returns:
Value of tolerance for synchronization in terms of Time object.

## setOffset

`void` **`setOffset`**`(Time t)`
Sets offset value in time for a slave Player to the master Player. The value is given by Time object.
Parameters:
`t` - Offset value in terms of Time object.

## setTolerance

`void` **`setTolerance`**`(Time t)`
Sets threshold to determine whether or not a specified slave Player establishes synchronization. In other words, an application allows error of synchronization within a given parameter, t.
Parameters:
`t` - Allowable tolerance of synchronization in terms of Time object.

# SyncStateChangedEvent

```
java.lang.Object
  └ java.util.EventObject
      └ org.dvb.media.SyncStateChangedEvent
```

All Implemented Interfaces:
java.io.Serializable

```
package org.dvb.media
public class SyncStateChangedEvent extends java.util.EventObject
```

An event which indicates change of status of synchronization between master/slave JMF Players.

This event is generated by MasterSlaveSyncLinkageControl object when status of synchronization between any combination of the master Player and slave Players.

# SyncStateChangedEvent

`public` **`SyncStateChangedEvent`**`(java.lang.Object source, Controller ctrl,int status)`
Constructs a SyncStateChangedEvent object.
Parameters:
`ctrl` - Controller object of a slave Player that is related to this object.
`status` - current status of synchronization as a result of its change.

## getSourceController

`public Controller` **`getSourceController`**`()`
Gets Controller object of a slave JMF Player object that is related to this event.
Returns:
A slave Player object related to this event object.

## getSyncStatus

`public int` **`getSyncStatus`**`()`
Gets changed synchronization status.
Returns:
changed synchronization status.

# SyncStateChangedEventListener

All Superinterfaces:
java.util.EventListener

```
package org.dvb.media
public interface SyncStateChangedEventListener extends java.util.EventListener
```

This listener interface is for receiving synchronization change events among JMF Players. The class that is interested in management of synchronization among JMF Players is registered with MasterSlaveSyncLinkageControl object using addSyncStateChangedEventListener method. The event is generated when synchronization status among JMF Players is changed. The relevant method in the listener object is then invoked, and the SyncStateChangedEvent is passed to it.

## inSync

```
void inSync(SyncStateChangedEvent e)
```
Invoked when any combination of master and slave Players are synchronized.

## outOfSync

```
void outOfSync(SyncStateChangedEvent e)
```
Invoked when any combination of master and slave Players are desynchronized.

## unSynchronizable

```
void unSynchronizable(SyncStateChangedEvent e)
```
Invoked when synchronization is never established.

# IncompatibleSynchronizableMediaTypeException

```
java.lang.Object
  └ java.lang.Throwable
      └ java.lang.Exception
          └ org.dvb.media.IncompatibleSynchronizableMediaTypeException
```

All Implemented Interfaces:

java.io.Serializable

```
package org.dvb.media
public class IncompatibleSynchronizableMediaTypeException extends java.lang.Exception
```

This exception is thrown when media type of each player is incompatible for synchronization.

## IncompatibleSynchronizableMediaTypeException

```
public IncompatibleSynchronizableMediaTypeException()
```

Constructs IncompatibleSynchronizableMediaTypeException.

## IncompatibleSynchronizableMediaTypeException

```
public IncompatibleSynchronizableMediaTypeException(java.lang.String reason)
```

Constructs IncompatibleSynchronizableMediaTypeException with reason.

Parameters:

`reason` - String object describing the reason of this exception.

# Annex O (normative): Integration of the JavaTV SI API

GEM terminal specifications shall contain a mapping of the Java TV SI API to the network's underlying signalling. This mapping shall fulfill all of the semantic guarantees required by Java TV. The Java TV SI is formed by the classes in the package javax.tv.service and its subpackages, as defined for GEM, including any descriptive text accompanying those classes.

# Annex P (normative): Broadcast transport protocol access

## P.1 Overview

This clause includes a definition of APIs in the package org.dvb.dsmcc. The API is mapped to an *abstraction* of signalling based on DSMCC; no requirement to use DSMCC signalling is implied.

To benefit from the fact that most of the functionalities are already covered by the `java.io` package, this API inherits from `java.io` and only defines the extra functionalities pertaining to:

a) the nature of a broadcast filesystem and its latency (e.g. possibility to asynchronously load the objects);

b) the type of the objects that can be encapsulated in a carousel and that do not exist in a classical file structure. These are: ServiceGateway, Stream and StreamEvent.

An application can optionally use only the classes of `java.io`. Alternatively/additionally applications can use additional classes and methods adapted to the specific nature and latency of the network (such as for example, the asynchronous loading of objects).

The following, briefly explains the functionalities offered by this API.

The `ServiceDomain` class enables attaching to a Service domain.

When attached to a Service Domain, objects are available representing the types File, Directory, Stream description, Trigger object and Trigger event.

The class `DSMCCObject` is a common superclass for all of these types. It defines methods that deal with asynchronous or synchronous loading of Objects.

For the File and Directory Objects, their content is accessible as it would be for a classical file system, i.e. by using the `java.io package` (e.g. for listing the objects pointed to by a Directory object, you invoke the `list()` method of the `java.io.File` class, or to access the content of a File, you can instantiate a `FileInputStream` to read the `File`, etc.)

Additionally, the `DSMCCStream` and `DSMCCStreamEvent` classes define functionalities specific to the respective types of Objects (Stream description and Trigger object), enabling access to the attributes of these Objects. For the details of the attributes that can be accessed, refer to the documentation of these classes.

The `AsynchronousLoadingEvent` class and its subclasses represent events that are sent to a listener to notify it of the loading of an Object that had been activated by the application (asynchronous loading mode).

The `StreamEvent` class represents an abstraction of the real event that is generated, i.e. the Trigger event, which enables the broadcaster to synchronize the application with the stream. This class enables the access to the content of an event, as described in clause B.4.2.

Finally, the `StreamEventListener` and `AsynchronousLoadingEventListener` are interfaces that shall be implemented by the application, in order for it to receive the respective `StreamEvents` and `AsynchronousLoadingEvents`.

# P.2 The org.dvb.dsmcc package

## P.2.0 General

This package is derived from the `org.dvb.dsmcc` package as defined in clause P.4. In GEM these methods are bound to the more abstract signalling requirements of annex B.

The description of each class from the org.dvb.dsmcc package defined in clause P.4 is included in the present document, except as modified below.

In all cases, references to a "DSMCC object" in the signalling is to be interpreted as referring to the entity in the signalling represented by an object of type DSMCCObject in the present document.

## P.2.1 DSMCCObject

The first sentence of the class description should be interpreted as meaning that this class represents objects in a Service domain.

NOTE: See also the semantic limitations on methods of this class under certain conditions specified in clause P.1, "Overview".

### P.2.1.1 DSMCCObject.getSigners()

GEM terminal specifications may be written such that some applications are trusted without being authenticated by the MHP mechanism. For these applications, a GEM terminal specification may be written such that it is a valid implementation option that this method returns an outer array of size zero when the DSMCCObject is loaded, if the file is not signed using the required MHP mechanism.

GEM terminal specifications may define a functionally equivalent mechanism for indicating the signers of files, as described in clause 12.2, "Authentication of applications." If such a mechanism is defined, the signers of a file shall be returned by this method.

NOTE: In the case where the MHP signing mechanism is not included in a GEM terminal specification, clause 12.2, "Authentication of applications" requires the presence of a functionally equivalent mechanism to indicate signers of files.

### P.2.1.2 DSMCCObject.getSigners(boolean known_root)

NOTE: The semantics of the DSMCCObject.getSigners(boolean known_root) method are unchanged and require processing.

## P.2.2 DSMCCStream

References to BIOP::Stream message are to be interpreted as meaning the stream description as defined in clause B.3. References to the BIOP::StreamEvent message are to be interpreted as meaning the stream even description defined in clause B.4.1. References to elements of the BIOP messages are to be interpreted as referring to the corresponding element of the generic descriptions from annex B, as detailed below.

### P.2.2.1 isAudio() method

This shall return true if the audio_stream flag indicates that this stream contains audio.

### P.2.2.2 isData() method

This shall return true if the data_stream flag indicates that this stream contains data.

### P.2.2.3 isMPEGProgram() method

This shall return true if the is_mpeg_program flag indicates that this object represents an MPEG program.

### P.2.2.4 isVideo() method

This shall return true if the video_stream flag indicates that this stream contains video.

## P.2.3 DSMCCStreamEvent

References to the BIOP::StreamEvent message are to be interpreted as meaning the stream even description defined in clause B.4.1. References to elements of the BIOP messages are to be interpreted as referring to the corresponding element of the generic descriptions from annex B, as detailed below.

Throughout this class, references to a DSMCC StreamEvent in the signalling are to be read as referring to a trigger object, as defined in clause B.4.1.

### P.2.3.1 Lightweight binding of trigger API

GEM terminal specifications may define a lightweight binding to the trigger API, which consists of `DSMCCStreamEvent` and related classes. The lightweight binding is recommend where signalling is available that cannot reasonably support the full DSMCC trigger API. This may be in addition to or instead of a signalling binding to the full API, such as is provided by the DSMCC signalling.

For the lightweight binding, no signalling is required for the trigger object, described in clause B.4.1, "Trigger object." Instead, at a minimum a single DSMCCStreamEvent object named `"lightweight_triggers"` shall be available at the top level for appropriate instances of ServiceDomain.

NOTE 1: GEM terminal specification may specify signalling that allows the name of this object to be signalled, or to otherwise generalize the minimum requirements laid out here.

NOTE 2: It is recommended that all services that can carry triggers have a defined binding to the trigger API. Applications may subscribe to named trigger events (defined in clause B.4.2, "Trigger event") using this instance of `DSMCCStreamEvent`. The GEM terminal specification is required to define a binding to determine the source of the trigger events for this instance.

#### P.2.3.1.1 DSMCCStreamEvent.getEventList()

For the lightweight binding of the trigger API, this method may return null.

#### P.2.3.1.2 StreamEvent.getEventId()

For the lightweight binding of the trigger API, this method may always return -1 to indicate that there is no event ID.

#### P.2.3.1.3 DSMCCStreamEvent.unsubscribe(int, StreamEventListener)

For the lightweight binding of the trigger API, this method may always throw `UnknownEventException`.

## P.2.4 InvalidFormatException

This exception may be thrown when any inconsistency in the underlying signalling is received.

## P.2.5 ServiceDomain

The first paragraph of the class description is to be replaced by the following: A ServiceDomain represents the entity described in clause B.1.

Throughout this class, references to "service gateway" or "service domain" are to be interpreted as referring to service domain, as described in clause B.1.

## P.2.5.1 ServiceDomain.attach(byte[])

`ServiceDomain.attach(byte[])` is required to be present by the present document, however, signalling to support an NSAP address is not required. In GEM Terminal Specifications where such signalling is not defined, it is a valid implementation for this method to do nothing.

NOTE: The functional equivalent named "Carousel" defines signalling for the NSAP address. See also clause 15.6.1.1.1, "NSAP Address".

## P.2.5.2 ServiceDomain.attach

### P.2.5.2.1 ServiceDomain.attach(Locator)

The locator parameter is to be interpreted as any locator that refers to a service domain. Locator formats are discussed in clause 14.8.

### P.2.5.2.2 ServiceDomain.attach(Locator, int)

The locator parameter is to be interpreted as any locator that refers to a service. The integer is to be interpreted as a unique identification of a service domain within that service. Locator formats are discussed in clause 14.8. The method `ServiceDomain.attach(Locator,int)` is not applicable in IPTV and shall always fail with a ServiceXFRException.

NOTE: This method is intended for referencing multiple carousels in-band in a single service, e.g. separate carousels for applications and for locally inserted content. This concept is really only applicable in uni-directional networks.

## P.2.5.3 ServiceDomain.getLocator()

The description of this method is considered to read as follows:

- Return the locator for this service domain. If this ServiceDomain instance was last attached by specifying a locator then an equivalent locator shall be returned. If the attach was done with the `attach(locator, int)` signature, the locator is complemented with a representation of the integer.

NOTE: As specified by `javax.tv.locator.Locator.equals()`, an equivalent locator has the same external form.

## P.2.5.4 ServiceDomain.getNSAPAddress()

Signalling to support the ServiceDomain.getNSAPAddress() method is not required by GEM. In terminal specifications where no such signalling is defined, the behaviour of invoking this method may be undefined.

## P.2.5.5 ServiceDomain.getURL(Locator)

The description of this static method is considered to read as follows:

Returns a URL corresponding to a locator referring to a file or a directory, as specified in table 59 in clause 14.8, "Locators and content referencing." If the service domain corresponding to the locator is attached and the file or directory referenced in the locator exists then an instance of java.net.URL is returned which can be used to reference this object.

Parameters:

`l` - a locator referring to a file or directory, as specified in table 59.

Returns:

a java.net.URL which can be used to access the file or directory referenced by the locator.

Throws:

`InvalidLocatorException` - if the locator is not a valid locator or does not includes all elements leading to a file or directory.

`NotLoadedException` - is thrown if the locator is valid and includes enough information but it references a service domain which is not attached.

`FileNotFoundException` - if the service domain is attached but the file or directory referenced by the locator does not exist.

### P.2.5.6 ServiceDomain.isNetworkConnectionAvailable()

The specification for this method is considered to be replaced by:

- This method shall return true if and only if the source(s) of data for this filesystem are connected and physically available to the terminal.

## P.2.6 ServiceXFRErrorEvent

This class is required by GEM, however signalling that would cause this error to be generated is not required by GEM.

## P.2.7 ServiceXFRException

This class is required by GEM, however signalling that would cause this exception to be generated is not required by GEM.

## P.2.8 ServiceXFRReference

This class is required by GEM, however signalling that would cause an event containing an object of this type is not required by GEM.

## P.2.9 StreamEvent

Throughout this class, references to the DSMCC stream event descriptor are to be read as referring to the trigger event, as described in clause B.4.2. References to the event data refer to the payload defined in that clause. References to "NPT" and "eventNPT" refer to "timebase values" and "timebase_value", respectively.

# P.3 Support for Stored Applications

The API defined in this clause allows DVB-J applications direct access to information broadcast according to annex B of the present document. GEM terminal specifications may make other filesystems available via this API (e.g. for files stored in local storage within the GEM terminal). These other file systems may subset the semantics of this package as follows:

- `DSMCCObject.abort()` - shall always throw a `NothingToAbortException`;
- `DSMCCObject.asynchronousLoad(AsynchronousLoadingEventListener)` - if the file exists, succeeds immediately (with `SuccessEvent` being generated) otherwise fails with `InvalidPathNameException`;
- `DSMCCObject.isObjectKindKnown()` - always returns true;
- `DSMCCObject.isStream()` - always returns false;

- `DSMCCObject.isStreamEvent()` - always returns false;
- `DSMCCObject.loadDirectoryEntry(AsynchronousLoadingEventListener)` - always succeeds immediately with `SuccessEvent` being generated;
- `DSMCCObject.prefetch(both signatures)` - always returns false;
- `DSMCCObject.setRetrievalMode(int)` - silently ignored;
- `DSMCCObject.synchronousLoad()` - if the file exists, succeeds immediately otherwise fails with `InvalidPathnameException`.

Additionally

- `ObjectChangeEvent` instances shall not be generated.
- The `DSMCCObject.unload()` method shall not be considered as removing stored files from where they are stored.
- Access to files shall not fail for reasons of a service domain not being in an attached state even though a stored filesystem cannot be represented by a DSMCC Service Domain.

NOTE 1: These semantics are identical to an agreed solution for MHP 1.1.2.

NOTE 2: In MHP, the carousel from which an application is run is automatically mounted. Therefore, MHP applications that use org.dvb.dsmcc do not necessarily use the ServiceDomain class.

## P.4 Javadoc for org.dvb.dsmcc package

# Package org.dvb.dsmcc

### Description

Provides extended access to files carried in the broadcast stream. It includes some extensions to java.io which are generic to (possibly) long-latency file systems and some concepts which are specific to the DSMCC object carousel.

| Class Summary | |
|---|---|
| **Interfaces** | |
| AsynchronousLoadingEventListener | Listener for applications which perform asynchronous loading, in order to be informed if the loading is done or if an error has occurred. |
| NPTListener | Objects that implement the `NPTListener` interface can receive `NPTStatus-Event` and `NPTRateChangedEvent` events. |
| ObjectChangeEventListener | The objects that implements the ObjectChangeEventListener interface can receive ObjectChangeEvent event. |
| StreamEventListener | Objects that implement the StreamEventListener interface can receive Stream-Event event. |
| **Classes** | |
| AsynchronousLoadingEvent | This class described an Object event which is used to notify the loading of a DSMCC object. |
| DSMCCObject | A `DSMCCObject` extends the java.io.File class with a combination of features appropriate for any file system with a long access time and features specific to the DSMCC object carousel as one example of such a file system. |
| DSMCCStream | The DSMCCStream class is used to manage DSMCC Stream Objects. |
| DSMCCStreamEvent | The DSMCCStreamEvent class is used to manage DSMCC StreamEvent Objects. |
| InsufficientResourcesEvent | This event is generated if there are insufficient resources available to load a DSMCCObject. e.g. if there is not enough memory. |
| InvalidFormatEvent | This event is generated if the format of the data received is inconsistent. |

| Class Summary | |
|---|---|
| InvalidPathnameEvent | The pathname does not exist or the ServiceDomain has been detached. |
| LoadingAbortedEvent | This event will be sent to the AsynchronousEventListener when an asynchronous loading operation is aborted. |
| MPEGDeliveryErrorEvent | An MPEGDeliveryErrorEvent indicates that an error (for instance, a time out or accessing the data would require tuning) has occurred while loading data from an MPEG Stream. |
| NotEntitledEvent | This event is sent when an attempt to asynchronously load an object has failed because the elementary stream carrying the object is scrambled and the user is not entitled to access the content of the object. |
| NPTDiscontinuityEvent | Sent when an MHP terminal detects a permanent value discontinuity in a broadcast timebase as defined in the main body of the present document. |
| NPTPresentEvent | Sent to listeners on a DSMCCStream object when a broadcast timeline newly appears for that DSMCC stream when it was not previously present. |
| NPTRate | Represents the rate at which an broadcast timeline progresses. |
| NPTRateChangeEvent | Sent only when the rate of an broadcast timeline changes value. |
| NPTRemovedEvent | Sent to listeners on a DSMCCStream object when a broadcast timeline which was present for that DSMCC stream is removed. |
| NPTStatusEvent | Sent when an MHP terminal detects a change of status in the broadcast timeline of a stream. |
| ObjectChangeEvent | This class describes an object change event that is used to monitor the arrival of a new version of a `DSMCCObject`. |
| ServerDeliveryErrorEvent | The local machine can not communicate with the server. |
| ServiceDomain | A `ServiceDomain` represents a group of DSMCC objects. |
| ServiceXFRErrorEvent | The object requested is available in an alternate ServiceDomain. |
| ServiceXFRReference | A ServiceXFRReference object is used when a DSMCC Object can not be loaded in the current ServiceDomain but is available in an alternate Service-Domain. |
| StreamEvent | This class describes a Stream event which is used to synchronize an application with an MPEG Stream. |
| SuccessEvent | This event indicates that the asynchronous loading was successful. |
| Exceptions | |
| DSMCCException | The DSMCCException is the root class of all DSMCC related exceptions |
| IllegalObjectTypeException | This Exception is thrown when the application attempted to create a DSMCCStream or DSMCCStreamEvent object with an object or a path that did not correspond to a DSMCC Stream or DSMCC StreamEvent respectively |
| InsufficientResourcesException | This exception gets thrown when a request to perform a DSMCC related operation cannot be completed due to resource limitations. |
| InvalidAddressException | An InvalidAddressException is thrown when the format of an NSAP address is not recognized. |
| InvalidFormatException | An InvalidFormatException is thrown when an inconsistent DSMCC message is received. |
| InvalidPathNameException | The InvalidPathNameException is thrown when the path name to a DSMCCObject does not exist or if the service domain is not detached. |
| MPEGDeliveryException | An MPEGDEliveryException is thrown when an error (for instance, a time out or accessing the data would require tuning) occurs while loading data from an MPEG Stream. |
| NotEntitledException | This Exception is thrown when the user is not entitled to access the content of the object (the Elementary Stream is scrambled and the user is not entitled). |
| NothingToAbortException | A NothingToAbortException is thrown when the abort method is called and there is no loading in progress. |
| NotLoadedException | A NotLoadedException is thrown when an operation fails because information which is required to be loaded has not been. |
| ServerDeliveryException | A ServerDeliveryException is thrown when the local machine can not communicate with the server. |
| ServiceXFRException | A ServiceXFRException is thrown when a DSMCC Object can not be loaded in the current ServiceDomain but is available in an alternate ServiceDomain (i.e. for an object Carousel, the IOR of the object or one of its parent directories contains a Lite Option Profile Body). |
| UnknownEventException | The `UnknownEventException` is thrown when a method tries to access to an unknown event. |

# org.dvb.dsmcc

# AsynchronousLoadingEvent

### Declaration

```
public abstract class AsynchronousLoadingEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Direct Known Subclasses:

InsufficientResourcesEvent, InvalidFormatEvent, InvalidPathnameEvent, LoadingAbortedEvent, MPEGDeliveryErrorEvent, NotEntitledEvent, ServerDeliveryErrorEvent, ServiceXFRErrorEvent, SuccessEvent

### Description

This class described an Object event which is used to notify the loading of a DSMCC object.

### Constructors

#### AsynchronousLoadingEvent(DSMCCObject)

```
public AsynchronousLoadingEvent(org.dvb.dsmcc.DSMCCObject o)
```

Creates an AsynchronousLoadingEvent.

#### Parameters:

- o - the DSMCCObject that generated the event.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that generated the event.

#### Overrides:

```
getSource in class EventObject
```

##### Returns:

the DSMCCObject that generated the event.

# org.dvb.dsmcc AsynchronousLoadingEventListener

### Declaration

```
public interface AsynchronousLoadingEventListener extends java.util.EventListener
```

### All Superinterfaces:

```
java.util.EventListener
```

### Description

Listener for applications which perform asynchronous loading, in order to be informed if the loading is done or if an error has occurred.

### Methods

#### receiveEvent(AsynchronousLoadingEvent)

```
public void receiveEvent(org.dvb.dsmcc.AsynchronousLoadingEvent e)
```

Method called when an event is sent to the application.

### Parameters:

`e` - an AsynchronousLoadingEvent event.

# org.dvb.dsmcc DSMCCException

```
Declaration
public class DSMCCException extends java.io.IOException

java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Direct Known Subclasses:

IllegalObjectTypeException, InsufficientResourcesException, InvalidAddressException, InvalidFormatException, InvalidPathNameException, MPEGDeliveryException, NotEntitledException, NothingToAbortException, NotLoadedException, ServerDeliveryException, ServiceXFRException, UnknownEventException

### Description

The DSMCCException is the root class of all DSMCC related exceptions.

### Constructors

### DSMCCException()

```
public DSMCCException()
```

Construct a DSMCCException with no detail message.

### DSMCCException(String)

```
public DSMCCException(java.lang.String s)
```

Construct a DSMCCException with the specified detail message.

### Parameters:

`s` - the detail message.

# org.dvb.dsmcc

# DSMCCObject

### Declaration

```
public class DSMCCObject extends java.io.File
```

```
java.lang.Object
  |
  +--java.io.File
        |
        +--org.dvb.dsmcc.DSMCCObject
```

### All Implemented Interfaces:

```
java.lang.Comparable, java.io.Serializable
```

### Description

A `DSMCCObject` extends the java.io.File class with a combination of features appropriate for any file system with a long access time and features specific to the DSMCC object carousel as one example of such a file system. Files referenced by a `DSMCCObject` exist within a file system. This may be represented by an instance of `ServiceDomain` but need not be, particularly for the file system in which the initial file of the application itself is carried.

A DSMCCObject is specified by a pathname, which can either be an absolute pathname or a relative pathname. Relative paths shall work as defined in "Broadcast Transport Protocol Access API" in the main body of the present document. Path names must follow the naming conventions of the host platform. The constructors of this class shall accept the absolute paths returned by java.io.File.getAbsolutePath().

To access the content of the object:

- For a Directory, the method list of the java.io.File class has to be used to get the entries of the directory.
- For a Stream object, the class DSMCCStream has to be used.
- For a File, the java.io.FileInputStream class or the java.io.RandomAccessFile has to be used.

NOTE: Obviously, for the Object Carousel, the write mode of java.io.RandomAccessFile is not allowed.

DSMCCObjects exist in two states, loaded and unloaded as returned by the isLoaded method. Transitions from unloaded to loaded are triggered by applications calling the asynchronousLoad or synchronousLoad or get-Signers(boolean) methods. Transitions from loaded to unloaded are triggered by applications calling the unload method. Attempting to load an already loaded object does not cause it to be re-loaded.

The only state transitions for a DSMCCObject shall be only in response to these method calls. There shall be no implicit state transitions in either direction. When the application no longer has any references to an object in the loaded state, the system resources allocated should be freed by the system.

The state machine of DSMCCObject is disconnected from any state model of any cache for the file system that contains it. Objects may appear in that cache without any corresponding DSMCCObject being in the loaded state. Objects which are in that cache and where any corresponding DSMCCObject is not in the loaded state may disappear from that cache at any time. The contents of a object may be accessible to applications from the cache without the DSMCCObject ever being in the loaded state.

NOTE: DSMCCObjects in the loaded state will consume memory in the MHP receiver. If memory in the MHP receiver is short, this memory can only be recovered by the receiver killing the MHP application. Applications which can accept weaker guarantees about the data of a DSMCCObject being available should use the prefetch methods.

See Also:

`ServiceDomain`

Fields

### FORCED_STATIC_CACHING

```
public static final int FORCED_STATIC_CACHING
```

Constant to indicate that the data for an object shall be returned from the cache if present, disregarding the cache priority signalling. If the data is not in the cache, it shall be retrieved from the network.

This strategy is intended solely for situations where generic DVB object carousels (as opposed to MHP object carousels) are being accessed and only the accessing application knows which files are static and which are dynamic. It should not be used to access files in MHP object carousels where the Caching priority descriptor can and should be used.

Since:

MHP 1.1.2.

### FROM_CACHE

```
public static final int FROM_CACHE
```

Constant to indicate that the data for an object shall only be retrieved where it is already in cache and meets the requirements of cache priority signaling. Where data is not in the cache, or the contents do not meet the requirements of the of cache priority signaling (i.e. cache priority signalling indicates that an object re-acquisition is required), attempts to load a DSMCCObject shall fail with `MPEGDeliveryException` or `MPEGDeliveryErrorEvent` for synchronousLoad and asynchronousLoad respectively.

Since:

MHP 1.0.1.

### FROM_CACHE_OR_STREAM

```
public static final int FROM_CACHE_OR_STREAM
```

Constant to indicate that the data for an object shall be automatically be retrieved from the network where the data is not already cached. Note that this method does not modify the caching policy controlled by the signaling in the OC. So, if the data is signalled as requiring transparent caching then data will be retrieved from the network if required.

Since:

MHP 1.0.1.

### FROM_STREAM_ONLY

```
public static final int FROM_STREAM_ONLY
```

Constant to indicate that the data for an object shall always be retrieved from the network even if the data has already been cached.

NOTE: This functionality is introduced deliberately to bypass any receiver cache allowing an application to be synchronised with the broadcast carousel. It should not be used simply to ensure up-to-date content is retrieved. It should also be noted here that this feature is provided to fulfil very specific application requirements (where other provided methods may not be suitable) and misuse of this functionality may seriously affect application/receiver performance.

Since:

MHP 1.0.1.

Constructors

DSMCCObject(DSMCCObject, String)

```
public DSMCCObject(org.dvb.dsmcc.DSMCCObject dir, java.lang.String name)
```

Create a DSMCCObject object.

Parameters:

`dir` - the directory object.

`name` - the file pathname.

DSMCCObject(String)

```
public DSMCCObject(java.lang.String path)
```

Create a DSMCCObject object.

Parameters:

`path` - the path to the file.

DSMCCObject(String, String)

```
public DSMCCObject(java.lang.String path, java.lang.String name)
```

Create a DSMCCObject object.

Parameters:

`path` - the directory Path.

`name` - the file pathname.

Methods

abort()

```
public void abort()
throws NothingToAbortException
```

This method is used to abort a load in progress. It can be used to abort either a synchronousLoad or an asynchronous-Load.

Throws:

`NothingToAbortException` - There is no loading in progress.

addObjectChangeEventListener(ObjectChangeEventListener)

```
public void addObjectChangeEventListener(org.dvb.dsmcc.ObjectChangeEventListener listener)
throws InsufficientResourcesException
```

Subscribes an ObjectChangeEventListener to receive notifications of version changes of DSMCCObject.

This listener shall never be fired until after the object has successfully entered the loaded state for the first time. Hence objects which never successfully enter the loaded state (e.g. because the object cannot be found) shall never have this listener fire. Once an object has successfully entered the loaded state once, this event shall continue to be fired when changes are detected by the MHP regardless of further transitions in or out of the loaded state.

NOTE: The algorithm used for this change monitoring is implementation dependent. In some implementations, this exception will always be thrown. In other implementations, it will never be thrown. In other implementations, whether it is thrown or not will depend on the complexity and design of the object carousel in which the object is carried. Even where no exception is thrown, implementations are not required to detect all possible forms in which an object may change.

Parameters:

`listener` - the ObjectChangeEventListener to be notified.

Throws:

`InsufficientResourcesException` - if there are not sufficient resources to monitor the object for changes.

### asynchronousLoad(AsynchronousLoadingEventListener)

```
public void asynchronousLoad(org.dvb.dsmcc.AsynchronousLoadingEventListener l)
throws InvalidPathNameException
```

This method starts the asynchronous loading of the data for a DSMCCObject. This method can fail either asynchronously with an event or synchronously with an exception. When it fails synchronously with an exception, no event shall be sent to the listener. For each call to this method which returns without throwing an exception, one of the following events will be sent to the application (by a listener mechanism) as soon as the loading is done or if an error has occurred: SuccessEvent, InvalidFormatEvent, InvalidPathNameEvent, MPEGDeliveryErrorEvent, ServerDelivery-ErrorEvent, ServiceXFRErrorEvent, NotEntitledEvent, LoadingAbortedEvent, InsufficientResourcesEvent.

Parameters:

`l` - an AsynchronousLoadingEventListener to receive events related to asynchronous loading.

Throws:

`InvalidPathNameException` - the Object can not be found, or the service domain is not in an attached state.

### getSigners()

```
public java.security.cert.X509Certificate[][] getSigners()
```

This method shall attempt to validate all certificate chains found for this file in the network. Valid chains do not need to originate from root certificates known to the MHP terminal, e.g. self signing of data files. Applications should note that calls to this method may take some time. If the `DSMCCObject` is not loaded, this method will return null. If the `DSMCCObject` is loaded, this method returns the same as getSigners(false), except if getSigners(false) would throw an exception, this method will return an outer array of size zero.

NOTE: If the file in the network changes between when it was loaded and when the hash file(s), signature & certificate files are read and those files have been updated to match the new version of the file then the hash value of the data which was loaded will not match the hash value in the hash file in the network and hence no certificate chains will be valid.

Returns:

a two-dimensional array of X.509 certificates, where the first index of the array determines a certificate chain and the second index identifies the certificate within the chain. Within one certificate chain the leaf certificate is first followed by any intermediate certificate authorities in the order of the chain with the root CA certificate as the last item.

Since:

MHP 1.0.1.

### getSigners(boolean)

```
public java.security.cert.X509Certificate[][] getSigners(boolean known_root)
throws InvalidFormatException, InterruptedIOException, MPEGDeliveryException, ServerDeliveryExceptio
n, InvalidPathNameException, NotEntitledException, ServiceXFRException, InsufficientResourcesExcepti
on
```

This method shall attempt to validate all certificate chains found for this file in the network. The process of determining validity is the same as the process of authentication except that when known_root is false, checking whether the root certificate is known is not included. The known_root parameter to the method defines whether the MHP terminal shall check if the root certificate in each chain is known to it or not. If the root certificate is checked then chains with unknown root certificates shall not be considered to be valid. If root certificates are not checked then the MHP application is responsible for comparing them with some certificate which it provides (e.g. for self signing of data files). The hash file(s), signature & certificate files shall be shall be fetched from the network in compliance with the caching priority defined in the main body of the present document. If the object is in the loaded state then the data of the file which was loaded shall be used and no new file contents loaded. If the object is not in the loaded state then this method shall attempt to load it as if synchronousLoad had been called. Applications should note that calls to this method may take some time.

NOTE: If the file in the network changes between when it was loaded and when the hash file(s), signature and certificate files are read and those files have been updated to match the new version of the file then the hash value of the data which was loaded will not match the hash value in the hash file in the network and hence no certificate chains will be valid.

#### Parameters:

`known_root` - if true then valid certificate chains are only those where the root is known to the MHP terminal. If false, the validity of the chain shall be determined without considering whether the root is known to the MHP terminal or not.

#### Returns:

a two-dimensional array of X.509 certificates, where the first index of the array determines a certificate chain and the second index identifies the certificate within the chain. Within one certificate chain the leaf certificate is first followed by any intermediate certificate authorities in the order of the chain with the root CA certificate as the last item. If no certificate chains are found to be valid then an outer array of size zero shall be returned.

#### Throws:

`java.io.InterruptedIOException` - the loading has been aborted.

`InvalidPathNameException` - the Object can not be found, or the service domain is not in an attached state.

`NotEntitledException` - the stream carrying the object is scrambled and the user has no entitlements to descramble the stream.

`ServiceXFRException` - the file system containing the DSMCCObject is a DSMCC object carousel and the IOR of the object or one of its parent directories is a Lite Option Profile Body and the referenced carousel is not already mounted by the MHP terminal.

`InvalidFormatException` - an error occured in the format of the file system containing the DSMCCObject, for instance an inconsistent DSMCC message has been received.

`MPEGDeliveryException` - an error has occurred while loading data for the file, for instance a timeout when reading data from an MPEG stream

`ServerDeliveryException` - when an MHP terminal cannot communicate with the server for files delivered over a bi-directional IP connection.

`InsufficientResourcesException` - there is not enough memory to load the object

#### Since:

MHP 1.0.3.

### getURL()

```
public java.net.URL getURL()
```

Returns a URL identifying this carousel object. If the directory entry for the object has not been loaded then null shall be returned.

**Returns:**

a URL identifying the carousel object or null.

**Since:**

MHP 1.0.1.

### isLoaded()

```
public boolean isLoaded()
```

Returns a boolean indicating whether or not the DSMCCObject has been loaded.

**Returns:**

true if the file is already loaded, false otherwise.

### isObjectKindKnown()

```
public boolean isObjectKindKnown()
```

Returns a boolean indicating if the kind of the object is known. (The kind of an object is known if the directory containing it is loaded).

**Returns:**

true if the type of the object is known, false otherwise.

### isStream()

```
public boolean isStream()
```

Returns a boolean indicating whether or not the DSMCCObject is a DSMCC Stream object.

**Returns:**

true if the file is a stream, false if the object is not a stream or if the object kind is unknown.

### isStreamEvent()

```
public boolean isStreamEvent()
```

Returns a boolean indicating whether or not the DSMCCObject is a DSMCC StreamEvent object.

NOTE: If isStreamEvent is true then isStream is true also.

**Returns:**

true if the file is a stream event, false if the object is not a stream event or if the object kind is unknown.

### loadDirectoryEntry(AsynchronousLoadingEventListener)

```
public void loadDirectoryEntry(org.dvb.dsmcc.AsynchronousLoadingEventListener l)
throws InvalidPathNameException
```

Asynchronous loading of the directory entry information. Calling this is equivalent of calling the method `asynchronousLoad` on the parent directory of a `DSMCCObject`. This method can fail either asynchronously with an event or synchronously with an exception. When it fails synchronously with an exception, no event shall be sent to the listener.

Parameters:

`l` - a listener which will be called when the loading is done.

Throws:

`InvalidPathNameException` - if the object cannot be found.

### prefetch(DSMCCObject, String, byte)

```
public static boolean prefetch(org.dvb.dsmcc.DSMCCObject dir, java.lang.String path, byte priority)
```

Calling this method will issue a hint to the MHP for pre-fetching the object data for that DSMCC object into cache.

Parameters:

`dir` - the directory object in which to pre-fetch the data.

`path` - the relative path name of object to pre-fetch, starting from the directory object passes as parameter.

`priority` - the relative priority of this pre-fetch request (higher = more important)

Returns:

true if the MHP supports pre-fetching (i.e. will try to process the request) and false otherwise. Note that a return value of 'true' is only an indication that the MHP receiver supports pre-fetching. It is not a guarantee that the requested data will actually be loaded into cache as the receiver may decide to drop the request in order to make resources available for regular load requests.

### prefetch(String, byte)

```
public static boolean prefetch(java.lang.String path, byte priority)
```

Calling this method will issue a hint to the MHP for pre-fetching the object data for that DSMCC object into cache.

Parameters:

`path` - the absolute pathname of the object to pre-fetch.

`priority` - the relative priority of this pre-fetch request (higher = more important)

Returns:

true if the MHP supports pre-fetching (i.e. will try to process the request) and false otherwise. Note that a return value of 'true' is only an indication that the MHP receiver supports pre-fetching. It is not a guarantee that the requested data will actually be loaded into cache as the receiver may decide to drop the request in order to make resources available for regular load requests.

### removeObjectChangeEventListener(ObjectChangeEventListener)

```
public void removeObjectChangeEventListener(org.dvb.dsmcc.ObjectChangeEventListener listener)
```

Unsubscribes an ObjectChangeEventListener to receive notifications of version changes of DSMCCObject.

Parameters:

`listener` - a previously registered ObjectChangeEventListener.

### setRetrievalMode(int)

```
public void setRetrievalMode(int retrieval_mode)
```

Set the retrieval mode for a `DSMCCObject`. The default retrieval mode is FROM_CACHE_OR_STREAM. The retrieval mode state is sampled when the object is loaded (whether explicitly or as described in "Constraints on the java.io.File methods for broadcast carousels"). Changing the retrieval mode for a loaded object has no effect until the object is unloaded and loaded again.

Parameters:

retrieval_mode - the retrieval mode to be used for the object specified as one of the public static final constants in this class.

Throws:

java.lang.IllegalArgumentException - if the retrieval_mode specified is not one listed defined for use with this method.

Since:

MHP 1.0.1.

### synchronousLoad()

```
public void synchronousLoad()
throws InvalidFormatException, InterruptedIOException, MPEGDeliveryException, ServerDeliveryExceptio
n, InvalidPathNameException, NotEntitledException, ServiceXFRException, InsufficientResourcesExcepti
on
```

This method is used to load a DSMCCObject. This method blocks until the file is loaded. It can be aborted from another thread with the abort method. In this case the InterruptedIOException is thrown. For DSMCCObjects contained within a DSMCC object carousel, if the IOR of the object itself or one of its parent directories is a Lite Option Profile Body and the referenced carousel is not attached, the MHP implementation will not attempt to resolve it: a ServiceXFRException is thrown to indicate to the application where the DSMCCObject is actually located.

Throws:

java.io.InterruptedIOException - the loading has been aborted.

InvalidPathNameException - the Object can not be found, or the service domain is not in an attached state.

NotEntitledException - the stream carrying the object is scrambled and the user has no entitlements to descramble the stream.

ServiceXFRException - the file system containing the DSMCCObject is a DSMCC object carousel and the IOR of the object or one of its parent directories is a Lite Option Profile Body and the referenced carousel is not already mounted by the MHP terminal.

InvalidFormatException - an error occured in the format of the file system containing the DSMCCObject, for instance an inconsistent DSMCC message has been received.

MPEGDeliveryException - an error has occurred while loading data for the file, for instance a timeout when reading data from an MPEG stream

ServerDeliveryException - when an MHP terminal cannot communicate with the server for files delivered over a bi-directional IP connection.

InsufficientResourcesException - there is not enough memory to load the object

### unload()

```
public void unload()
throws NotLoadedException
```

When calling this method, the applications gives a hint to the MHP that if this object is not consumed by another application/thread, the system can free all the resources allocated to this object. It is worth noting that if other clients use this object (e.g. a file input stream is opened on this object or if the corresponding stream or stream event is being consumed) the system resources allocated to this object will not be freed. This method puts the DSMCCObject into the unloaded state.

Throws:

NotLoadedException - the carousel object is not loaded.

# org.dvb.dsmcc

# DSMCCStream

### Declaration

```
public class DSMCCStream

java.lang.Object
  |
  +--org.dvb.dsmcc.DSMCCStream
```

### Direct Known Subclasses:

DSMCCStreamEvent

### Description

The DSMCCStream class is used to manage DSMCC Stream Objects. The BIOP::Stream message shall be read from the network once only, before the constructor of this class returns. Hence methods which return information from that message shall not be affected by any subsequent changes to that information.

The methods in this class relating to the NPT mechanism shall also be mapped onto the DVB timeline as defined in the main part of the present document.

NOTE: The NPT mechanism and scheduled stream events that depend on it are known to be vulnerable to disruption in many digital TV distribution networks. Existing deployed network equipment that re-generates the STC is unlikely to be aware of NPT and hence will not make the necessary corresponding modification to STC values inside NPT reference descriptors. Applications should only use NPT where they are confident that the network where they are to be used does not have this problem.

### See Also:

DSMCCObject

### Constructors

#### DSMCCStream(DSMCCObject)

```
public DSMCCStream(org.dvb.dsmcc.DSMCCObject aDSMCCObject)
throws NotLoadedException, IllegalObjectTypeException
```

Creates a Stream Object from a DSMCC Object. The BIOP message referenced by the DSMCCObject has to be a Stream or StreamEvent BIOP message.

#### Parameters:

`aDSMCCObject` - the DSMCC object which describes the stream

#### Throws:

`NotLoadedException` - the DSMCCObject is not loaded.

`IllegalObjectTypeException` - the DSMCCObject is neither a DSMCC Stream nor a DSMCCStream-Event.

#### DSMCCStream(String)

```
public DSMCCStream(java.lang.String path)
throws IOException, IllegalObjectTypeException
```

Create a Stream Object from its pathname. For an object Carousel, this method will block until the module which contains the object is loaded. The BIOP message referenced by the DSMCCObject pointed to by the parameter path has to be a Stream or StreamEvent BIOP message.

Parameters:

path - the pathname of the DSMCCStream Object.

Throws:

java.io.IOException - If an IO error occurred.

IllegalObjectTypeException - the DSMCCObject is neither a DSMCC Stream nor a DSMCCStream-Event.

### DSMCCStream(String, String)

```
public DSMCCStream(java.lang.String path, java.lang.String name)
throws IOException, IllegalObjectTypeException
```

Create a DSMCCStream from its pathname. For an object Carousel, this method will block until the module which contains the object is loaded. The BIOP message referenced by the DSMCCObject pointed to be the parameters path and name has to be a Stream or StreamEvent BIOP message.

Parameters:

path - the directory path.

name - the name of the DSMCCStream Object.

Throws:

java.io.IOException - If an IO error occurred.

IllegalObjectTypeException - the DSMCCObject is neither a DSMCC Stream nor a DSMCCStream-Event.

## Methods

### addNPTListener(NPTListener)

```
public void addNPTListener(org.dvb.dsmcc.NPTListener l)
```

Add a listener for timeline events on the DSMCCStream object. Adding the same listener a second time has no effect.

Parameters:

l - the listener.

Since:

MHP 1.0.1.

### getDuration()

```
public long getDuration()
```

This function returns the duration in milliseconds of the DSMCC Stream. If the DSMCCStream BIOP message does not specify duration, zero will be returned.

Returns:

The duration in milliseconds of the DSMCC Stream.

### getMediaTimeFromStream(long, Clock)

```
public javax.media.Time getMediaTimeFromStream(long time, javax.media.Clock c)
```

Convert a time measured according to a timeline in the stream into a JMF media time measured in a specified clock.

Parameters:

time - the time measured according to the timeline in the stream.

c - the clock to express the media time.

Returns:

the JMF media time corresponding to the specified time.

Throws:

`java.lang.IllegalArgumentException` - if the Clock is not a Player which is presenting at least one of the streams of this collection (as returned by getStreamLocator).

### getNPT()

```
public long getNPT()
throws MPEGDeliveryException
```

This function is used to get value of the current timeline in milliseconds. Implementations are not required to continuously monitor for the timeline descriptors. In implementations which do not continuously monitor, this method will block while the current timeline is retrieved from the stream.

Returns:

the current time in milliseconds or zero if the DSMCC Stream object BIOP message does not contain any taps pointing to a timeline.

Throws:

`MPEGDeliveryException` - if there is an error in retrieving the timeline descriptors.

### getNPTRate()

```
public org.dvb.dsmcc.NPTRate getNPTRate()
throws MPEGDeliveryException
```

Get the rate for the timeline referenced in the DSMCCStream object. Where the timeline is NPT, the NPT rate is returned. Where the timeline is the DVB timeline, 0/1 is returned if it is paused and 1/1 is returned if it is advancing. The method returns null if no timeline is referenced by the DSMCCStream object, (i.e. for NPT, no STR_NPT_USE tap in the list of taps).

Returns:

the NPT rate or null.

Throws:

throws - MPEGDeliveryException if there is an error in retrieving NPT reference descriptors.

`MPEGDeliveryException`

Since:

MHP 1.0.1.

### getStreamLocator()

```
public org.davic.net.Locator getStreamLocator()
```

This function returned a Locator referencing the streams of this collection. The interpretation of the return value is determined by the `isMPEGProgram` method.

Returns:

a locator.

### isAudio()

```
public boolean isAudio()
```

This function returns a boolean indicating if the Stream Object refers to an audio stream. This is the case if the audio field in the Stream(Event) BIOP message has a value different from zero.

Returns:

true only if the Stream object refers to an audio stream.

### isData()

```
public boolean isData()
```

This function returns a boolean indicating if the Stream Object refers to a data stream. This is the case if the data field in the Stream(Event) BIOP message has a value different from zero.

Returns:

true only if the Stream object refers to a data stream.

### isMPEGProgram()

```
public boolean isMPEGProgram()
```

This method will return true if the Stream(Event) BIOP message contains a tap with use field BIOP_PROGRAM_USE, otherwise it will return false.

Returns:

true only if the Stream(Event) BIOP message is as described above.

### isVideo()

```
public boolean isVideo()
```

This function returns a boolean indicating if the Stream Object refers to an video stream. This is the case if the `video' field in the Stream(Event) BIOP message has a value different from zero otherwise false is returned.

Returns:

true only if the Stream object refers to an video stream.

### removeNPTListener(NPTListener)

```
public void removeNPTListener(org.dvb.dsmcc.NPTListener l)
```

Remove a listener for timeline events on the `DSMCCStream` object. Removing a non-subscribed listener has no effect.

Parameters:

`l` - the listener to remove.

Since:

MHP 1.0.1.

# org.dvb.dsmcc

# DSMCCStreamEvent

### Declaration

```
public class DSMCCStreamEvent extends DSMCCStream
```

```
java.lang.Object
  |
  +--org.dvb.dsmcc.DSMCCStream
        |
        +--org.dvb.dsmcc.DSMCCStreamEvent
```

### Description

The DSMCCStreamEvent class is used to manage DSMCC StreamEvent Objects. Applications wishing to monitor changes in the list of events which are part of this stream event should use `DSMCCObject.addObjectChangeEvent-Listener` on the `DSMCCObject` representing which describes this set of stream events. The BIOP::StreamEvent message shall be read from the network once only, before the constructor of this class returns. Hence methods which return information from that message shall not be affected by any subsequent changes to that information.

The subscribe method only verifies that the event name can be bound to an eventId but it does not require that the stream event descriptors for that event id can be received at that moment. While the event listener is registered the MHP terminal shall filter the stream event descriptors as specified in the "Monitoring broadcast timebases and events" clause in the main body of the specification.

NOTE: The NPT mechanism and scheduled stream events that depend on it are known to be vulnerable to disruption in many digital TV distribution networks. Existing deployed network equipment that re-generates the STC is unlikely to be aware of NPT and hence will not make the necessary corresponding modification to STC values inside NPT reference descriptors. Applications should only use NPT where they are confident that the network where they are to be used does not have this problem.

### Constructors

#### DSMCCStreamEvent(DSMCCObject)

```
public DSMCCStreamEvent(org.dvb.dsmcc.DSMCCObject aDSMCCObject)
throws NotLoadedException, IllegalObjectTypeException
```

Create a `DSMCCStreamEvent` from a `DSMCCObject`. The Object has to be a DSMCC StreamEvent.

#### Parameters:

`aDSMCCObject` - the DSMCC object which describes the stream.

#### Throws:

`NotLoadedException` - the DSMCCObject is not loaded.

`IllegalObjectTypeException` - the DSMCCObject does not lead to a DSMCC StreamEvent.

#### DSMCCStreamEvent(String)

```
public DSMCCStreamEvent(java.lang.String path)
throws IOException, IllegalObjectTypeException
```

Create a DSMCCStreamEvent Object from its pathname. The path has to lead to a DSMCCStreamEvent.

#### Parameters:

`path` - the pathname of the DSMCCStreamEvent object.

#### Throws:

`java.io.IOException` - An IO error has occurred.

`IllegalObjectTypeException` - the path does not lead to a DSMCC StreamEvent.

#### DSMCCStreamEvent(String, String)

```
public DSMCCStreamEvent(java.lang.String path, java.lang.String name)
throws IOException, IllegalObjectTypeException
```

Create a DSMCCStreamEvent from its pathname. For an object Carousel, this method will block until the module which contains the object is loaded. The path has to lead to a DSMCC Stream Event.

#### Parameters:

`path` - the directory path.

`name` - the name of the DSMCCStreamEvent Object.

Throws:

java.io.IOException - If an IO error occurred.

IllegalObjectTypeException - the path does not lead to a DSMCC StreamEvent.

Methods

getEventList()

```
public java.lang.String[] getEventList()
```

This function is used to get the list of the events of the DSMCCStreamEvent object.

Returns:

The list of the eventName.

subscribe(String, StreamEventListener)

```
public int subscribe(java.lang.String eventName, org.dvb.dsmcc.StreamEventListener l)
throws UnknownEventException, InsufficientResourcesException
```

This function is used to subscribe to an event of a DSMCC StreamEvent object.

Parameters:

eventName - the name of the event.

l - an object that implements the StreamEventListener Interface.

Returns:

The event Identifier.

Throws:

UnknownEventException - the event cannot be found at this time.

InsufficientResourcesException - if resources needed to perform the subscription are not available.

unsubscribe(int, StreamEventListener)

```
public void unsubscribe(int eventId, org.dvb.dsmcc.StreamEventListener l)
throws UnknownEventException
```

This function is used to cancel the subscription to an event of a DSMCCEvent object.

Parameters:

eventId - Identifier of the event.

l - an object that implements the StreamEventListener Interface.

Throws:

UnknownEventException - The event can not be found.

unsubscribe(String, StreamEventListener)

```
public void unsubscribe(java.lang.String eventName, org.dvb.dsmcc.StreamEventListener l)
throws UnknownEventException
```

This function is used to cancel the subscription to an event of a DSMCCEvent object.

Parameters:

eventName - the name of the event.

l - an object that implements the StreamEventListener Interface.

Throws:

UnknownEventException - The event can not be found.

# org.dvb.dsmcc IllegalObjectTypeException

## Declaration

```
public class IllegalObjectTypeException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.IllegalObjectTypeException
```

## All Implemented Interfaces:

```
java.io.Serializable
```

## Description

This Exception is thrown when the application attempted to create a DSMCCStream or DSMCCStreamEvent object with an object or a path that did not correspond to a DSMCC Stream or DSMCC StreamEvent respectively.

## Constructors

### IllegalObjectTypeException()

```
public IllegalObjectTypeException()
```

Constructor of the exception with no detail message.

### IllegalObjectTypeException(String)

```
public IllegalObjectTypeException(java.lang.String s)
```

Constructor of the exception.

## Parameters:

s - detail message.

# org.dvb.dsmcc InsufficientResourcesEvent

## Declaration

```
public class InsufficientResourcesEvent extends AsynchronousLoadingEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
              |
              +--org.dvb.dsmcc.InsufficientResourcesEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This event is generated if there are insufficient resources available to load a DSMCCObject. e.g. if there is not enough memory.

### Constructors

#### InsufficientResourcesEvent(DSMCCObject)

```
public InsufficientResourcesEvent(org.dvb.dsmcc.DSMCCObject o)
```

Create an InsufficientResourcesException object.

### Parameters:

- o - the DSMCCObject that generated the event.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that generated the event.

### Overrides:

getSource in class AsynchronousLoadingEvent

### Returns:

the DSMCCObject that generated the event.

# org.dvb.dsmcc

# InsufficientResourcesException

### Declaration

```
public class InsufficientResourcesException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.InsufficientResourcesException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This exception gets thrown when a request to perform a DSMCC related operation cannot be completed due to resource limitations. For example, no section filters or system timers may be available.

Since:

MHP 1.0.1.

Constructors

InsufficientResourcesException()

```
public InsufficientResourcesException()
```

Construct an InsufficientResourcesException with no detail message.

InsufficientResourcesException(String)

```
public InsufficientResourcesException(java.lang.String message)
```

Construct an InsufficientResourcesException with the specified detail message.

Parameters:

`message` - the message for the exception.

# org.dvb.dsmcc InvalidAddressException

Declaration

```
public class InvalidAddressException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.InvalidAddressException
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

An InvalidAddressException is thrown when the format of an NSAP address is not recognized.

Constructors

InvalidAddressException()

```
public InvalidAddressException()
```

Construct a InvalidAddressException with no detail message.

InvalidAddressException(String)

```
public InvalidAddressException(java.lang.String s)
```

Construct a InvalidAddressException with the specified detail message.

Parameters:

`s` - the detail message.

# org.dvb.dsmcc
# InvalidFormatEvent

### Declaration

```
public class InvalidFormatEvent extends AsynchronousLoadingEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
              |
              +--org.dvb.dsmcc.InvalidFormatEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This event is generated if the format of the data received is inconsistent.

### Constructors

#### InvalidFormatEvent(DSMCCObject)

```
public InvalidFormatEvent(org.dvb.dsmcc.DSMCCObject o)
```

Create an InvalidFormatException object.

#### Parameters:

- o - the DSMCCObject that generated the event.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that generated the event.

#### Overrides:

getSource in class AsynchronousLoadingEvent

#### Returns:

the DSMCCObject that generated the event.

# org.dvb.dsmcc InvalidFormatException

### Declaration

```
public class InvalidFormatException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.InvalidFormatException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

An InvalidFormatException is thrown when an inconsistent DSMCC message is received.

### Constructors

#### InvalidFormatException()

```
public InvalidFormatException()
```

Construct an InvalidFormatException with no detail message.

#### InvalidFormatException(String)

```
public InvalidFormatException(java.lang.String s)
```

Construct an InvalidFormatException with the specified detail message.

### Parameters:

`s` - the detail message.

# org.dvb.dsmcc InvalidPathnameEvent

### Declaration

```
public class InvalidPathnameEvent extends AsynchronousLoadingEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
              |
              +--org.dvb.dsmcc.InvalidPathnameEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

The pathname does not exist or the ServiceDomain has been detached.

### Constructors

#### InvalidPathnameEvent(DSMCCObject)

```
public InvalidPathnameEvent(org.dvb.dsmcc.DSMCCObject o)
```

Create an InvalidPathnameEvent.

#### Parameters:

- o - the DSCMCCObject that generated this event.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that generated the event.

#### Overrides:

getSource in class AsynchronousLoadingEvent

#### Returns:

the DSMCCObject that generated the event.

# org.dvb.dsmcc

# InvalidPathNameException

### Declaration

```
public class InvalidPathNameException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.InvalidPathNameException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

The InvalidPathNameException is thrown when the path name to a DSMCCObject does not exist or if the service domain is not detached.

### Constructors

#### InvalidPathNameException()

```
public InvalidPathNameException()
```

Construct an InvalidPathNameException with no detail message.

### InvalidPathNameException(String)

```
public InvalidPathNameException(java.lang.String s)
```

Construct an InvalidPathNameException with the specified detail message.

Parameters:

s - the detail message.

# org.dvb.dsmcc

# LoadingAbortedEvent

### Declaration

```
public class LoadingAbortedEvent extends AsynchronousLoadingEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
              |
              +--org.dvb.dsmcc.LoadingAbortedEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This event will be sent to the AsynchronousEventListener when an asynchronous loading operation is aborted.

Since:

MHP 1.0.1.

### Constructors

### LoadingAbortedEvent(DSMCCObject)

```
public LoadingAbortedEvent(org.dvb.dsmcc.DSMCCObject aDSMCCObject)
```

Creates a LoadingAbortedEvent object.

Parameters:

aDSMCCObject - the DSMCCObject that generated the event.

### Methods

### getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that generated the event.

Overrides:

getSource in class AsynchronousLoadingEvent.

Returns:

the DSMCCObject whose loading was aborted.

# org.dvb.dsmcc

# MPEGDeliveryErrorEvent

### Declaration

```
public class MPEGDeliveryErrorEvent extends AsynchronousLoadingEvent

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
              |
              +--org.dvb.dsmcc.MPEGDeliveryErrorEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

An MPEGDeliveryErrorEvent indicates that an error (for instance, a time out or accessing the data would require tuning) has occurred while loading data from an MPEG Stream.

### Constructors

#### MPEGDeliveryErrorEvent(DSMCCObject)

```
public MPEGDeliveryErrorEvent(org.dvb.dsmcc.DSMCCObject o)
```

Creates an MPEGDeliveryEvent.

#### Parameters:

- o - the DSMCCObject that generated the event.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that generated the event.

#### Overrides:

getSource in class AsynchronousLoadingEvent

#### Returns:

the DSMCCObject that generated the event.

# org.dvb.dsmcc MPEGDeliveryException

### Declaration

```
public class MPEGDeliveryException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.MPEGDeliveryException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

An MPEGDEliveryException is thrown when an error (for instance, a time out or accessing the data would require tuning) occurs while loading data from an MPEG Stream.

### Constructors

#### MPEGDeliveryException()

```
public MPEGDeliveryException()
```

Construct an MPEGDeliveryException with no detail message.

#### MPEGDeliveryException(String)

```
public MPEGDeliveryException(java.lang.String s)
```

Construct an MPEGDeliveryException with the specified detail message.

### Parameters:

`s` - the detail message.

# org.dvb.dsmcc NotEntitledEvent

### Declaration

```
public class NotEntitledEvent extends AsynchronousLoadingEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
              |
              +--org.dvb.dsmcc.NotEntitledEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This event is sent when an attempt to asynchronously load an object has failed because the elementary stream carrying the object is scrambled and the user is not entitled to access the content of the object.

### Constructors

#### NotEntitledEvent(DSMCCObject)

```
public NotEntitledEvent(org.dvb.dsmcc.DSMCCObject o)
```

Creates a NotEntitledEvent object.

#### Parameters:

- o - the DSMCCObject that generated the event.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that generated the event.

#### Overrides:

getSource in class AsynchronousLoadingEvent

#### Returns:

the DSMCCObject that generated the event.

# org.dvb.dsmcc

# NotEntitledException

### Declaration

```
public class NotEntitledException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.NotEntitledException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This Exception is thrown when the user is not entitled to access the content of the object (the Elementary Stream is scrambled and the user is not entitled).

### Constructors

#### NotEntitledException()

```
public NotEntitledException()
```

Construct a NotEntitledException with no detail message.

#### NotEntitledException(String)

```
public NotEntitledException(java.lang.String s)
```

Construct a NotEntitledException with a detail message.

#### Parameters:

`s` - detail message.

# org.dvb.dsmcc

# NothingToAbortException

### Declaration

```
public class NothingToAbortException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.NothingToAbortException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

A NothingToAbortException is thrown when the abort method is called and there is no loading in progress.

### Constructors

#### NothingToAbortException()

```
public NothingToAbortException()
```

Construct a NothingToAbortException with no detail message.

#### NothingToAbortException(String)

```
public NothingToAbortException(java.lang.String s)
```

Construct a NothingToAbortException with the specified detail message.

#### Parameters:

`s` - the detail message.

# org.dvb.dsmcc NotLoadedException

### Declaration

```
public class NotLoadedException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.NotLoadedException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

A NotLoadedException is thrown when an operation fails because information which is required to be loaded has not been.

### Constructors

#### NotLoadedException()

```
public NotLoadedException()
```

Construct a NotLoadedException with no detail message.

#### NotLoadedException(String)

```
public NotLoadedException(java.lang.String s)
```

Construct a NotLoadedException with the specified detail message.

### Parameters:

`s` - the detail message.

# org.dvb.dsmcc NPTDiscontinuityEvent

### Declaration

```
public class NPTDiscontinuityEvent extends NPTStatusEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.NPTStatusEvent
              |
              +--org.dvb.dsmcc.NPTDiscontinuityEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Sent when an MHP terminal detects a permanent value discontinuity in a broadcast timebase as defined in the main body of the present document. This may represent an error condition in the incoming broadcast.

Where the timebase is delivered using NPT, this event shall be sent following a PCR discontinuity when the MHP terminal has enough information to determine that there will be an NPT discontinuity. If the `NPTDiscontinuity-Event` is because of invalid data in a new NPTReferenceDescriptor then the event will be generated when that new NPTReferenceDescriptor is detected by the MHP terminal. If the `NPTDiscontinuityEvent` is because no new NPTReferenceDescriptor is detected within the time allowed by the main body of the present document then it will be generated when that time interval has elapsed.

### Since:

MHP 1.0.1.

### Constructors

### NPTDiscontinuityEvent(DSMCCStream, long, long)

```
public NPTDiscontinuityEvent(org.dvb.dsmcc.DSMCCStream source, long before, long after)
```

Construct an event. The `before` and `after` values used shall be the values at the time when the receiver determined that a NPT discontinuity has happened. If the `NPTDiscontinuityEvent` is because of invalid data in a new NPTReference-Descriptor then this is the time when that new descriptor was known to be invalid. If `NPTDiscontinuityEvent` is because of the absence of a new NPTReferenceDescriptor then this will be when the MHP terminal detects that the time interval allowed by the present document for such new descriptors has elapsed. Where an NPT value is unknown or cannot be computed, -1 shall be used.

### Parameters:

`source` - the stream whose NPT suffered a discontinuity.

`before` - the last NPT value detected before the discontinuity or -1 for DVB timelines.

`after` - the first NPT value detected after the discontinuity or -1 for DVB timelines.

### Methods

### getFirstNPT()

```
public long getFirstNPT()
```

Return the first known stable value of NPT after the discontinuity.

### Returns:

an NPT value or -1 for DVB timelines.

### getLastNPT()

```
public long getLastNPT()
```

Return the last known stable value of NPT before the discontinuity.

### Returns:

an NPT value or -1 for DVB timelines.

# org.dvb.dsmcc
# NPTListener

### Declaration

```
public interface NPTListener extends java.util.EventListener
```

### All Superinterfaces:

```
java.util.EventListener
```

### Description

Objects that implement the `NPTListener` interface can receive `NPTStatusEvent` and `NPTRateChangedEvent` events.

### Since:

MHP 1.0.1.

### Methods

#### receiveNPTStatusEvent(NPTStatusEvent)

```
public void receiveNPTStatusEvent(org.dvb.dsmcc.NPTStatusEvent e)
```

Send a `NPTStatusEvent` to a registered listener.

### Parameters:

`e` - a `NPTStatusEvent` describing the status change.

#### receiveRateChangedEvent(NPTRateChangeEvent)

```
public void receiveRateChangedEvent(org.dvb.dsmcc.NPTRateChangeEvent e)
```

Send a `NPTRateChangeEvent` to a registered listener.

### Parameters:

`e` - the `NPTRateChangeEvent` event.

# org.dvb.dsmcc
# NPTPresentEvent

### Declaration

```
public class NPTPresentEvent extends NPTStatusEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.NPTStatusEvent
              |
              +--org.dvb.dsmcc.NPTPresentEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Sent to listeners on a DSMCCStream object when a broadcast timeline newly appears for that DSMCC stream when it was not previously present. This is specific to the particular timebase for this stream.

### Since:

MHP 1.0.1.

### Constructors

#### NPTPresentEvent(DSMCCStream)

```
public NPTPresentEvent(org.dvb.dsmcc.DSMCCStream source)
```

Construct an event.

### Parameters:

`source` - the DSMCCStream for which the broadcast timeline appeared.

# org.dvb.dsmcc

# NPTRate

### Declaration

```
public class NPTRate

java.lang.Object
  |
  +--org.dvb.dsmcc.NPTRate
```

### Description

Represents the rate at which an broadcast timeline progresses. Rates are expressed as the combination of a numerator and a denominator. Instances of this class are constructed by the platform and returned to applications. A rate of 1/1 shall indicate "the standard presentation rate" as defined in the NPT specification.

### Since:

MHP 1.0.1.

### Methods

#### getDenominator()

```
public int getDenominator()
```

Get the NPT rate's denominator.

### Returns:

the denominator.

#### getNumerator()

```
public int getNumerator()
```

Get the NPT rate's numerator. A value of zero indicates that the NPT is not progressing.

### Returns:

the numerator.

# org.dvb.dsmcc
# NPTRateChangeEvent

### Declaration

```
public class NPTRateChangeEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.NPTRateChangeEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Sent only when the rate of an broadcast timeline changes value.

### Since:

MHP 1.0.1.

### Constructors

#### NPTRateChangeEvent(DSMCCStream, NPTRate)

```
public NPTRateChangeEvent(org.dvb.dsmcc.DSMCCStream source, org.dvb.dsmcc.NPTRate rate)
```

Construct an event.

#### Parameters:

`source` - the stream whose rate changed.

`rate` - the new rate of that stream immediately following the change.

### Methods

#### getRate()

```
public org.dvb.dsmcc.NPTRate getRate()
```

Return the new rate of the stream immediately after the change.

#### Returns:

a `NPTRate` object encapsulating the new rate.

#### getSource()

```
public java.lang.Object getSource()
```

Return the stream whose rate changed.

#### Overrides:

`getSource` in class `EventObject`

#### Returns:

the `DSMCCStream` object on which the rate change has occurred.

# org.dvb.dsmcc
# NPTRemovedEvent

### Declaration

```
public class NPTRemovedEvent extends NPTStatusEvent

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.NPTStatusEvent
              |
              +--org.dvb.dsmcc.NPTRemovedEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Sent to listeners on a DSMCCStream object when a broadcast timeline which was present for that DSMCC stream is removed. This is specific to the particular timebase for this stream.

### Since:

MHP 1.0.1.

### Constructors

#### NPTRemovedEvent(DSMCCStream)

```
public NPTRemovedEvent(org.dvb.dsmcc.DSMCCStream source)
```

Construct an event.

### Parameters:

`source` - the DSMCCStream from which the broadcast timeline was removed.

# org.dvb.dsmcc
# NPTStatusEvent

### Declaration

```
public abstract class NPTStatusEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.NPTStatusEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Direct Known Subclasses:

```
NPTDiscontinuityEvent, NPTPresentEvent, NPTRemovedEvent
```

### Description

Sent when an MHP terminal detects a change of status in the broadcast timeline of a stream.

Since:

MHP 1.0.1.

Constructors

NPTStatusEvent(DSMCCStream)

```
public NPTStatusEvent(org.dvb.dsmcc.DSMCCStream source)
```

Construct an event.

Parameters:

`source` - the stream whose broadcast timeline status changed.

Methods

getSource()

```
public java.lang.Object getSource()
```

Return the stream whose broadcast timeline status changed.

Overrides:

`getSource` in class `EventObject`

Returns:

the `DSMCCStream` whose status changed.

# org.dvb.dsmcc

# ObjectChangeEvent

Declaration

```
public class ObjectChangeEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.ObjectChangeEvent
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

This class describes an object change event that is used to monitor the arrival of a new version of a `DSMCCObject`. For files carried in a DSMCC object carousel, when a change in a module is detected, this event shall be sent to all registered listeners for all objects carried in that module.

Constructors

ObjectChangeEvent(DSMCCObject, int)

```
public ObjectChangeEvent(org.dvb.dsmcc.DSMCCObject source, int aVersionNumber)
```

Creates an ObjectChangeEvent indicating that a new version of the monitored DSMCC Object has been detected. It is up to the application to reload the new version of the object.

Parameters:

`source` - the DSMCCObject whose version has changed.

`aVersionNumber` - the new version number.

Methods

getNewVersionNumber()

```
public int getNewVersionNumber()
```

This method is used to get the new version number of the monitored `DSMCCObject`. For files carried in a DSMCC object carousel, this method shall return the version number of the module carrying the file.

Returns:

the new version number.

getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that has changed.

Overrides:

`getSource` in class `EventObject`

Returns:

the DSMCCObject that has changed.

# org.dvb.dsmcc

# ObjectChangeEventListener

Declaration

```
public interface ObjectChangeEventListener extends java.util.EventListener
```

All Superinterfaces:

```
java.util.EventListener
```

Description

The objects that implements the ObjectChangeEventListener interface can receive ObjectChangeEvent event.

Methods

receiveObjectChangeEvent(ObjectChangeEvent)

```
public void receiveObjectChangeEvent(org.dvb.dsmcc.ObjectChangeEvent e)
```

Send a ObjectChangeEvent to the ObjectChangeEventListener.

Parameters:

`e` - the ObjectChangeEvent event.

# org.dvb.dsmcc
# ServerDeliveryErrorEvent

### Declaration

```
public class ServerDeliveryErrorEvent extends AsynchronousLoadingEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
              |
              +--org.dvb.dsmcc.ServerDeliveryErrorEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

The local machine can not communicate with the server.  This event is only used with files implemented by delivery over bi-directional IP connections. For the object carousel the `MPEGDeliveryErrorEvent` is used instead.

### Constructors

#### ServerDeliveryErrorEvent(DSMCCObject)

```
public ServerDeliveryErrorEvent(org.dvb.dsmcc.DSMCCObject o)
```

Creates a ServerDeliveryEvent object.

### Parameters:

- o - the DSMCCObject that generated the event.

### Methods

#### getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that generated the event.

### Overrides:

`getSource` in class `AsynchronousLoadingEvent`

### Returns:

the DSMCCObject that generated the event.

# org.dvb.dsmcc ServerDeliveryException

### Declaration

```
public class ServerDeliveryException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.ServerDeliveryException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

A ServerDeliveryException is thrown when the local machine can not communicate with the server. This exception is only used with files implemented by delivery over a bi-directional IP connection. For the object carousel the `MPEGDeliveryException` is used instead.

### Constructors

### ServerDeliveryException()

```
public ServerDeliveryException()
```

Construct a ServerDeliveryException with no detail message.

### ServerDeliveryException(String)

```
public ServerDeliveryException(java.lang.String s)
```

Construct a ServerDeliveryException with the specified detail message.

### Parameters:

`s` - the detail message.

# org.dvb.dsmcc ServiceDomain

### Declaration

```
public class ServiceDomain
```

```
java.lang.Object
  |
  +--org.dvb.dsmcc.ServiceDomain
```

### Description

A `ServiceDomain` represents a group of DSMCC objects. The objects are sent either using the object carousel for a broadcast network or with the DSM-CC User-to-User protocol for an interactive network.

To access the objects of a `ServiceDomain`, it has to be attached to the file system name space of the MHP terminal. To access the content of an object, the application has four ways:

- It can instantiate the class that is used to read the object (java.io.FileInputStream or java.io.RandomAccessFile for a File or DSMCCStream for a stream) from its pathname. The loading of the object is implicit but the application has no way to abort it. NB: Obviously, for the Object Carousel, the write mode of java.io.RandomAccessFile is not allowed.
- It can instantiate a DSMCCObject and carry out a Synchronous loading. The loading can be aborted by the abort method of the DSMCCObject class. When the object is loaded, the application will instantiate the class used to read the object.
- It can instantiate a DSMCCObject and carry out an Asynchronous loading. So several loading can be started in parallel from the same thread.
- It is also possible to create directly a java.io.File for a DSMCC object.

Instances of `ServiceDomain` exist in two states, attached and detached. Newly created instances are always in the detached state. They become attached when a call to the `attach` method succeeds. They become detached following a call to the `detach` method.

When service domains in the attached state temporarily lose their network connection (e.g. if the MHP terminal tunes away from the transport stream where they are carried), the behaviour of DSMCC objects which are part of the service domain is specified in the main body of the present document. If such a network connection becomes available again then the service domain shall resume normal behaviour.

A service domain which is temporarily lost its network connection may be forced into the detached state by the implementation if the loss of the network connection becomes irrecoverable. The precise details of when this happens are implementation dependent. This is the only situation when shall be forced into the detached state. Once a ServiceDomain is detached, it will never be automatically attached.

### Constructors

#### ServiceDomain()

```
public ServiceDomain()
```

Creates a ServiceDomain object.

### Methods

#### attach(byte[])

```
public void attach(byte[] NSAPAddress)
throws DSMCCException, InterruptedIOException, InvalidAddressException, MPEGDeliveryException
```

This function is used to attach a `ServiceDomain` from either an object carousel or from an interactive network. This call will block until the attachment is done.

Calling this method on a `ServiceDomain` object already in the attached state shall imply a detach of the `ServiceDomain` object before the attach operation unless the `ServiceDomain` is already attached to the correct location. Hence if the attach operation fails, the appropriate exception for the failure mode shall be thrown and the `ServiceDomain` is left in a detached state and not attached to the former object carousel / service domain. If the `ServiceDomain` is already attached to the correct location then the method call shall have no effect.

#### Parameters:

`NSAPAddress` - The NSAP Address of a ServiceDomain as defined in in ISO/IEC 13818-6 [38].

#### Throws:

`java.io.InterruptedIOException` - The attachment has been aborted.

`InvalidAddressException` - The NSAP Address is invalid.

`DSMCCException` - An error has occurred during the attachment.

MPEGDeliveryException - attaching to this domain would require tuning.

### attach(Locator)

```
public void attach(org.davic.net.Locator l)
throws DSMCCException, InterruptedIOException, MPEGDeliveryException
```

This function is used to attach a ServiceDomain from an object carousel. It loads the module which contains the service gateway object and mounts the ServiceDomain volume in the file system hierarchy. This call will block until the service gateway is loaded. It can be aborted by another thread with the method detach. In this case an InterruptedIOException is thrown.

Calling this method on a ServiceDomain object already in the attached state shall imply a detach of the ServiceDomain object before the attach operation unless the ServiceDomain is already attached to the correct location. Hence if the attach operation fails, the appropriate exception for the failure mode shall be thrown and the ServiceDomain is left in a detached state and not attached to the former object carousel/service domain. If the ServiceDomain is already attached to the correct location then the method call shall have no effect.

#### Parameters:

l - The locator pointing to the elementary stream carrying the DSI of the object carousel, or to a DVB service that carries one and only one object carousel.

#### Throws:

DSMCCException - An error has occurred during the attachment. For example, the locator does not point to a component carrying a DSI of an Object Carousel or to a service containing a single carousel.

java.io.InterruptedIOException - The attachment has been aborted.

MPEGDeliveryException - attaching to this domain would require tuning.

### attach(Locator, int)

```
public void attach(org.davic.net.Locator aDVBService, int aCarouselId)
throws ServiceXFRException, InterruptedIOException, MPEGDeliveryException
```

This function is used to attach a ServiceDomain from an object carousel. It loads the module which contains the service gateway object and mounts the ServiceDomain volume in the file system hierarchy. This call will block until the service gateway is loaded. It can be aborted by another thread with the method detach. In this case an InterruptedIOException is thrown.

Calling this method on a ServiceDomain object already in the attached state shall imply a detach of the ServiceDomain object before the attach operation unless the ServiceDomain is already attached to the correct location. Hence if the attach operation fails, the appropriate exception for the failure mode shall be thrown and the ServiceDomain is left in a detached state and not attached to the former object carousel/service domain. If the ServiceDomain is already attached to the correct location then the method call shall have no effect.

#### Parameters:

aDVBService - The coordinates of the DVB service which contains the object carousel. This locator has to point to a DVB service.

aCarouselId - The identifier of the carousel.

#### Throws:

java.io.InterruptedIOException - The attachment has been aborted.

MPEGDeliveryException - An MPEG error occurred (such as time-out).

ServiceXFRException - The service gateway cannot be loaded in the current service domain. This exception shall not be thrown in this version of the specification.

### detach()

```
public void detach()
throws NotLoadedException
```

A call to this method is a hint that the applications gives to the MHP to unmount the volume and delete the objects of the service domain. When another application is using objects of the same service domain the method has no effects. When there are no other application using objects of the service domain, a call to this method is a hint that the MHP can free all the resources allocated to this service domain.

After this, the `ServiceDomain` will be in a non-attached state and will behave as if it had just been constructed. Subsequent calls to `detach` shall throw `NotLoadedException`.

#### Throws:

`NotLoadedException` - is thrown if the ServiceDomain is not attached or if there is no call to `attach` in progress.

### getContextData(int)

```
public byte[] getContextData(int context_id)
throws NotLoadedException
```

Returns the context_data_bytes from the service context list of the ServiceDomain's service gateway object.

#### Parameters:

`context_id` - the context_id whose context_data is to be returned.

#### Returns:

an array containing the context_data_bytes or null if the context_id specified is not present in the service context list.

#### Throws:

`NotLoadedException` - if the service domain is not loaded.

### getLocator()

```
public org.davic.net.Locator getLocator()
```

Return the locator for this service domain. If this ServiceDomain instance was last attached by specifying a locator then an equivalent locator shall be returned except if the original locator contained extra information that is not necessary to identify the service domain in which case this extra information is removed. If the attach was done with the `attach(locator, int)` signature, the locator is complemented with the component_tag value that the platform has identified during attaching the ServiceDomain. If this ServiceDomain instance was last attached by specifying an NSAP address then the locator shall be generated from that address. If this ServiceDomain has never been attached then null shall be returned.

The syntax of the NSAP address is defined in section titled "LiteOptionsProfileBody" in annex B of the MHP specification. It contains the same fields as the locator syntax specified in the System integration aspects clause. The locator is constructed by taking the fields out of the NSAP address and encoding them in the locator syntax together with the component_tag value that the platform has identified during attaching the ServiceDomain.

#### Returns:

a locator for this service domain.

#### Since:

MHP 1.0.1.

### getMountPoint()

```
public org.dvb.dsmcc.DSMCCObject getMountPoint()
```

Returns a `DSMCCObject` object describing the top level directory of this `ServiceDomain`. If the ServiceDomain object is not attached then null is returned.

Returns:

an instance of org.dvb.dsmcc.DSMCCObject if attached or null otherwise.

Since:

MHP 1.0.1.

### getNSAPAddress()

```
public byte[] getNSAPAddress()
throws NotLoadedException
```

This method returns the NSAP address of the `ServiceDomain`.

Returns:

the NSAP address of the `ServiceDomain`.

Throws:

`NotLoadedException` - is thrown if the `ServiceDomain` is not attached.

### getURL(Locator)

```
public static java.net.URL getURL(org.davic.net.Locator l)
throws NotLoadedException, InvalidLocatorException, FileNotFoundException
```

Obtain a java.net.URL corresponding to a 'dvb:' locator. If the service domain corresponding to the locator is attached and the file referenced in the locator exists then an instance of `java.net.URL` is returned which can be used to reference this file.

Parameters:

`l` - a locator object encapsulating a 'dvb:' locator which includes a 'dvb_abs_path' element.

Returns:

a `java.net.URL` which can be used to access the file referenced by the 'dvb:' locator.

Throws:

`org.davic.net.InvalidLocatorException` - if the locator is not a valid 'dvb:' locator or does not includes all elements including 'dvb_abs_path' element.

`NotLoadedException` - is thrown if the locator is valid and includes enough information but it references a service domain which is not attached.

`java.io.FileNotFoundException` - if the service domain is attached but the file referenced by the locator does not exist.

### isAttached()

```
public boolean isAttached()
```

Return whether this service domain is in the attached or detached state.

Returns:

true if this service domain is in the attached state, otherwise false.

Since:

MHP 1.0.1.

### isNetworkConnectionAvailable()

```
public boolean isNetworkConnectionAvailable()
```

Return whether the network connection for this service domain is available. This return value is independent of whether the service domain is attached or not. If a service domain is distributed across multiple network connections (e.g. using the optional support for DSMCC over IIOP) then this will reflect the availability of the network connection carrying the object mounted to the  mount point.

Returns:

true if the network connection for this service domain is available otherwise false.

Since:

MHP 1.0.1.

# org.dvb.dsmcc

# ServiceXFRErrorEvent

## Declaration

```
public class ServiceXFRErrorEvent extends AsynchronousLoadingEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
              |
              +--org.dvb.dsmcc.ServiceXFRErrorEvent
```

## All Implemented Interfaces:

```
java.io.Serializable
```

## Description

The object requested is available in an alternate ServiceDomain. When an application attempts to asynchronously load an object that has itself a LiteOptionProfileBody IOR or that has a parent directory that has a LiteOptionProfileBody IOR, this event shall be sent to the application.  There is no implicit mounting by the implementation of the carousel that actually contains the object. This event is not sent if the Service Domain that actually contains the DSMCCObject is already mounted.

## Constructors

### ServiceXFRErrorEvent(DSMCCObject, ServiceXFRReference)

```
public ServiceXFRErrorEvent(org.dvb.dsmcc.DSMCCObject o, org.dvb.dsmcc.ServiceXFRReference ref)
```

Creates a ServiceXFRErrorEvent object.

Parameters:

`o` - the DSMCCObject that generated the event.

`ref` - the address of an alternate ServiceDomain where the object can be found.

## Methods

### getServiceXFR()

```
public org.dvb.dsmcc.ServiceXFRReference getServiceXFR()
```

This method is used to get a reference to the service domain that contains the requested object.

Returns:

The address of an alternate ServiceDomain where the object can be found.

### getSource()

```
public java.lang.Object getSource()
```

Returns the DSMCCObject that generated the event.

Overrides:

getSource in class AsynchronousLoadingEvent

Returns:

the DSMCCObject that generated the event.

# org.dvb.dsmcc ServiceXFRException

### Declaration

```
public class ServiceXFRException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.ServiceXFRException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

A ServiceXFRException is thrown when a DSMCC Object can not be loaded in the current ServiceDomain but is available in an alternate ServiceDomain (i.e. for an object Carousel, the IOR of the object or one of its parent directories contains a Lite Option Profile Body). There is no implicit mounting by the implementation of the carousel that actually contain the object. This exception is not thrown if the Service Domain that actually contains the DSMCCObject is already mounted.

### Constructors

#### ServiceXFRException(byte[], String)

```
public ServiceXFRException(byte[] NSAPAddress, java.lang.String pathName)
```

Creates a ServiceXFRException object.

Parameters:

NSAPAddress - The NSAP Address of a ServiceDomain as defined in ISO/IEC 13818-6 [38].

pathName - pathName of the object in the alternate ServiceDomain.

#### ServiceXFRException(Locator, int, String)

```
public ServiceXFRException(org.davic.net.Locator aService, int carouselId,
java.lang.String pathName)
```

Creates a ServiceXFRException object.

Parameters:

aService - Locator of the Service.

carouselId - Carousel Identifier.

pathName - pathName of the object in the alternate ServiceDomain.

Methods

getServiceXFR()

```
public org.dvb.dsmcc.ServiceXFRReference getServiceXFR()
```

This method is used to get the alternate ServiceDomain which contains the object requested.

Returns:

the address of an alternate ServiceDomain where the object can be found.

# org.dvb.dsmcc ServiceXFRReference

```
Declaration
public class ServiceXFRReference

java.lang.Object
  |
  +--org.dvb.dsmcc.ServiceXFRReference
```

Description

A ServiceXFRReference object is used when a DSMCC Object can not be loaded in the current ServiceDomain but is available in an alternate ServiceDomain. Instances of this class are just containers. The parameters passed are merely stored and returned by the access methods. It is the responsibility of the platform when generating instances to use correct values.

Constructors

ServiceXFRReference(byte[], String)

```
public ServiceXFRReference(byte[] nsapAddress, java.lang.String pathName)
```

Creates a ServiceXFRReference object.

Parameters:

nsapAddress - The NSAP Address of a ServiceDomain as defined in ISO/IEC 13818-6 [38].

pathName - pathName of the object in the alternate ServiceDomain.

ServiceXFRReference(Locator, int, String)

```
public ServiceXFRReference(org.davic.net.Locator serviceLocator, int carouselId,
java.lang.String pathName)
```

Creates a ServiceXFRReference object.

Parameters:

serviceLocator - Locator of the Service.

carouselId - Carousel Identifier.

pathName - pathName of the object in the alternate ServiceDomain.

### Methods

#### getCarouselId()

```
public int getCarouselId()
```

This method returns the carousel identifier. If the object was constructed using the constructor which includes a carousel ID or if it was constructed using the constructor which includes an NSAP address and that NSAP address contains a carouselID then this method shall return that carousel ID otherwise this method shall return -1.

Returns:

the carousel identifier or -1.

#### getLocator()

```
public org.davic.net.Locator getLocator()
```

This method returns the Locator of the Service for an Object Carousel.

Returns:

the Locator of the Service for an Object Carousel. This method returns null, if the ServiceDomain is not associated with an Object Carousel. In this case the NSAP address must be used instead.

#### getNSAPAddress()

```
public byte[] getNSAPAddress()
```

This method returns the NSAP Address of a ServiceDomain as defined in ISO/IEC 13818-6 [38]. If the object was constructed using an NSAP address then this method shall return the NSAP address passed into the constructor. If the object was constructed with a locator and a carouselID then this method shall return an NSAP address derived from this information when locator is an instance of org.davic.net.dvb.DVBLocator. Otherwise this method shall return null.

Returns:

the NSAP Address of a ServiceDomain as defined in ISO/IEC 13818-6 [38] or null.

#### getPathName()

```
public java.lang.String getPathName()
```

This method returns the pathname of the object in the alternate ServiceDomain.

Returns:

the pathname of the object in the alternate ServiceDomain.

# org.dvb.dsmcc

# StreamEvent

### Declaration

```
public class StreamEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.StreamEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This class describes a Stream event which is used to synchronize an application with an MPEG Stream.

NOTE: The NPT mechanism and scheduled stream events that depend on it are known to be vulnerable to disruption in many digital TV distribution networks. Existing deployed network equipment that re-generates the STC is unlikely to be aware of NPT and hence will not make the necessary corresponding modification to STC values inside NPT reference descriptors. This may cause stream events scheduled against NPT to fire at the wrong time or to never fire at all. Applications should only use scheduled stream events with NPT where they are confident that the network where they are to be used does not have this problem. DVB Timeline, DSM-CC "do it now" events and events carried within DVB synchronised auxiliary data offer more reliable alternatives to NPT.

### Constructors

#### StreamEvent(DSMCCStreamEvent, long, String, int, byte[])

```
public StreamEvent(org.dvb.dsmcc.DSMCCStreamEvent source, long npt, java.lang.String name,
int eventId, byte[] eventData)
```

Creates a StreamEvent object.

#### Parameters:

`source` - The `DSMCCStreamEvent` that has generated the event.

`npt` - The value of the timeline when the event is triggered. For NPT based timelines, this value is equal to the field eventNPT in the DSMCC StreamEventDescriptor except where the event is a "do it now" event in which case the value -1 is returned (as the value of NPT may not be meaningful). For DVB timelines, this value is the to-be-completed

`name` - The name of this event. The list of event names is located in the DSMCC StreamEvent object. This list is returned by the method `DSMCCStreamEvent.getEventList`.

`eventId` - The eventId of this event. The list of event IDs is located in the DSMCC StreamEvent object.

`eventData` - The application specific data found in the DSMCC StreamEventDescriptor.

### Methods

#### getEventData()

```
public byte[] getEventData()
```

This method is used to retrieve the private data associated with the event. For events signalled by a DVB synchronised event descriptor, these are the bytes carried in the synchronised_event_data_byte field. For events signalled by a DSMCC stream event descriptor, these are the contents of the privateDataByte field.

#### Returns:

The private data associated with the event.

#### getEventId()

```
public int getEventId()
```

This method is used to get the identifier of the StreamEvent.

#### Returns:

The identifier of the StreamEvent.

#### getEventName()

```
public java.lang.String getEventName()
```

This method is used to get the name of the StreamEvent.

Returns:

the name of the StreamEvent.

getEventNPT()

```
public long getEventNPT()
```

This method is used to get the value of the timeline when the event was triggered.

Returns:

The value of the timeline in milliseconds.

getSource()

```
public java.lang.Object getSource()
```

This method returns the DSMCCStreamEvent that generated the event.

Overrides:

`getSource` in class `EventObject`

Returns:

the DSMCCStreamEvent that generated the event.

# org.dvb.dsmcc StreamEventListener

Declaration

```
public interface StreamEventListener extends java.util.EventListener
```

All Superinterfaces:

```
java.util.EventListener
```

Description

Objects that implement the StreamEventListener interface can receive StreamEvent event.

Methods

receiveStreamEvent(StreamEvent)

```
public void receiveStreamEvent(org.dvb.dsmcc.StreamEvent e)
```

Send a StreamEvent to the StreamEventListener.

Parameters:

`e` - the StreamEvent event.

# org.dvb.dsmcc SuccessEvent

Declaration

```
public class SuccessEvent extends AsynchronousLoadingEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.dsmcc.AsynchronousLoadingEvent
              |
              +--org.dvb.dsmcc.SuccessEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This event indicates that the asynchronous loading was successful.

### Constructors

### SuccessEvent(DSMCCObject)

```
public SuccessEvent(org.dvb.dsmcc.DSMCCObject o)
```

Creates a `SuccessEvent` object.

### Parameters:

`o` - the `DSMCCObject` which was successfully loaded.

### Methods

### getSource()

```
public java.lang.Object getSource()
```

Returns the `DSMCCObject` which was successfully loaded.

### Overrides:

`getSource` in class `AsynchronousLoadingEvent`

### Returns:

the loaded `DSMCCObject`

# org.dvb.dsmcc

# UnknownEventException

### Declaration

```
public class UnknownEventException extends DSMCCException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--java.io.IOException
                    |
                    +--org.dvb.dsmcc.DSMCCException
                          |
                          +--org.dvb.dsmcc.UnknownEventException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

The `UnknownEventException` is thrown when a method tries to access to an unknown event. This exception may get thrown because the event in question is not being signalled yet. It does not indicate that the event is permanently unavailable. Applications may choose to attempt to subscribe to the event again at a later point in time in the expectation that the event has become available since the previous attempt.

### Constructors

#### UnknownEventException()

```
public UnknownEventException()
```

Construct an UnknownEventException with no detail message.

#### UnknownEventException(String)

```
public UnknownEventException(java.lang.String s)
```

Construct an UnknownEventException with the specified detail message.

### Parameters:

`s` - the detail message.

# Annex Q (normative): Datagram socket buffer control

# Package

# org.dvb.net

### Description

Provides general networking features not found elsewhere.

| Class Summary | |
|---|---|
| **Classes** | |
| DatagramSocketBufferControl | This class provides additional control over buffering for DatagramSockets. |

# org.dvb.net

# DatagramSocketBufferControl

### Declaration

```
public class DatagramSocketBufferControl

java.lang.Object
  |
  +--org.dvb.net.DatagramSocketBufferControl
```

### Description

This class provides additional control over buffering for DatagramSockets.

### Methods

#### getReceiveBufferSize(DatagramSocket)

```
public static int getReceiveBufferSize(java.net.DatagramSocket d)
throws SocketException
```

Get value of the SO_RCVBUF option for this socket, that is the buffer size used by the platform for input on the this Socket. The value returned need not be the value previously set by setReceiveBufferSize (if any).

Parameters:

d - The DatagramSocket for which to query the receive buffer size.

Returns:

The size of the receive buffer, in bytes.

Throws:

java.net.SocketException - - If there is an error when querying the SO_RCVBUF option.

### setReceiveBufferSize(DatagramSocket, int)

```
public static void setReceiveBufferSize(java.net.DatagramSocket d, int size)
throws SocketException
```

Sets the SO_RCVBUF option to the specified value for this DatagramSocket. The SO_RCVBUF option is used by the platform's networking code as a hint for the size to use when allocating the underlying network I/O buffers.

Increasing buffer size can increase the performance of network I/O for high-volume connection, while decreasing it can help reduce the backlog of incoming data. For UDP, this sets the buffer size for received packets.

Because SO_RCVBUF is a hint, applications that want to verify what size the buffers were set to should call get-ReceiveBufferSize. This method shall throw IllegalArgumentException - if size is 0 or is negative.

Parameters:

`d` - The DatagramSocket for which to change the receive buffer size.

`size` - The requested size of the receive buffer, in bytes.

Throws:

`java.net.SocketException` - - If there is an error when setting the SO_RCVBUF option.

# Annex R (normative):
# DVB-J return channel connection management API

# Package

# org.dvb.net.rc

Description

Provides session management for bi-directional IP connections which are session based from the point of view of an application. The best example of this is a conventional modem.

| Class Summary | |
|---|---|
| **Interfaces** | |
| `ConnectionListener` | This interface should be implemented by objects wishing to be notified about the connection status of a `ConnectionRCInterface`. |
| **Classes** | |
| `ConnectionDroppedEvent` | ConnectionDroppedEvent - An event generated after an attempt to setup a connection for a `ConnectionRCInterface` fails due to the connection being dropped by the target. |
| `ConnectionEstablishedEvent` | ConnectionEstablishedEvent - An event generated after a connection is established for a `ConnectionRCInterface`. |
| `ConnectionFailedEvent` | ConnectionFailedEvent - An event generated after an attempt to setup a connection for a `ConnectionRCInterface` fails. |
| `ConnectionParameters` | This class encapsulates the parameters needed to specify the target of a connection. |
| `ConnectionRCEvent` | ConnectionRCEvent - the base class for events related to connection oriented return channels. |
| `ConnectionRCInterface` | This class models a connection based return channel network interface for use in receiving and transmitting IP packets over a return channel. |
| `ConnectionTerminatedEvent` | ConnectionTerminatedEvent - An event generated after a connected `ConnectionRCInterface` is disconnected. |
| `NoDialToneEvent` | NoDialToneEvent - An event generated after an attempt to setup a connection for a `ConnectionRCInterface` of `TYPE_PSTN` fails due to there not being a dial tone on the return channel concerned. |
| `RCInterface` | This class models a return channel network interface for use in receiving and transmitting IP packets over a logical return channel. |
| `RCInterfaceManager` | This class is the factory and manager for all return channel interfaces in the system. |
| `RCInterfaceReleasedEvent` | This event informs an application that a `RCInterface` has been released by an application or other entity in the system. |
| `RCInterfaceReservedEvent` | This event informs an application that a `RCInterface` has been reserved by an application or other entity in the system. |
| `RCPermission` | This class is for return channel set-up permissions. |
| `TargetBusyEvent` | TargetBusyEvent - An event generated after an attempt to setup a connection for a `ConnectionRCInterface` fails due to the target of the connection being busy. |
| **Exceptions** | |
| `IncompleteTargetException` | Thrown when the target for a connection is incompletely specified. |
| `PermissionDeniedException` | Thrown when an application calls a method which it does not have permission to call at that time. |

# org.dvb.net.rc ConnectionDroppedEvent

### Declaration

```
public class ConnectionDroppedEvent extends ConnectionFailedEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.net.rc.ConnectionRCEvent
              |
              +--org.dvb.net.rc.ConnectionFailedEvent
                    |
                    +--org.dvb.net.rc.ConnectionDroppedEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

ConnectionDroppedEvent - An event generated after an attempt to setup a connection for a `ConnectionRCInterface` fails due to the connection being dropped by the target. For a ConnectionRCInterface of type `TYPE_PSTN`, this includes the following conditions:

- Connection was dropped after ringing but before it was answered.
- Connection rang but was never answered and timed-out.
- Connection was dropped after answering but before the IP connection was established.
- Connection was answered but the IP connection was never established and timed-out.

### Constructors

#### ConnectionDroppedEvent(Object)

```
public ConnectionDroppedEvent(java.lang.Object source)
```

Construct an event.

### Parameters:

`source` - the `ConnectionRCInterface` whose connection attempt failed.

# org.dvb.net.rc ConnectionEstablishedEvent

### Declaration

```
public class ConnectionEstablishedEvent extends ConnectionRCEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.net.rc.ConnectionRCEvent
              |
              +--org.dvb.net.rc.ConnectionEstablishedEvent
```

### All Implemented Interfaces:

`java.io.Serializable`

### Description

ConnectionEstablishedEvent - An event generated after a connection is established for a `ConnectionRCInterface`.

### Constructors

### ConnectionEstablishedEvent(Object)

```
public ConnectionEstablishedEvent(java.lang.Object source)
```

Construct an event.

### Parameters:

`source` - the `ConnectionRCInterface` whose connection was established.

# org.dvb.net.rc

# ConnectionFailedEvent

### Declaration

```
public class ConnectionFailedEvent extends ConnectionRCEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.net.rc.ConnectionRCEvent
              |
              +--org.dvb.net.rc.ConnectionFailedEvent
```

### All Implemented Interfaces:

`java.io.Serializable`

### Direct Known Subclasses:

`ConnectionDroppedEvent, NoDialToneEvent, TargetBusyEvent`

### Description

ConnectionFailedEvent - An event generated after an attempt to setup a connection for a `ConnectionRCInterface` fails.

### Constructors

### ConnectionFailedEvent(Object)

```
public ConnectionFailedEvent(java.lang.Object source)
```

Construct an event.

### Parameters:

`source` - the `ConnectionRCInterface` whose connection attempt failed.

# org.dvb.net.rc ConnectionListener

### Declaration

```
public interface ConnectionListener extends java.util.EventListener
```

### All Superinterfaces:

```
java.util.EventListener
```

### Description

This interface should be implemented by objects wishing to be notified about the connection status of a `ConnectionRCInterface`.

### Methods

#### connectionChanged(ConnectionRCEvent)

```
public void connectionChanged(org.dvb.net.rc.ConnectionRCEvent e)
```

This method is called to report events related to the setup and termination of return channel interface connections.

### Parameters:

`e` - the event which happened.

# org.dvb.net.rc ConnectionParameters

### Declaration

```
public class ConnectionParameters
```

```
java.lang.Object
  |
  +--org.dvb.net.rc.ConnectionParameters
```

### Description

This class encapsulates the parameters needed to specify the target of a connection.

### Constructors

#### ConnectionParameters(String, String, String)

```
public ConnectionParameters(java.lang.String number, java.lang.String username,
java.lang.String password)
```

Construct a set of connection parameters. Details of the DNS server to use are supplied by the server.

### Parameters:

`number` - the target of the connection, e.g. a phone number.

`username` - the username to use in connection setup.

`password` - the password to use in connection setup.

### ConnectionParameters(String, String, String, InetAddress[])

```
public ConnectionParameters(java.lang.String number, java.lang.String username,
java.lang.String password, java.net.InetAddress[] dns)
```

Construct a set of connection parameters.

Parameters:

`number` - the target of the connection, e.g. a phone number.

`username` - the username to use in connection setup.

`password` - the password to use in connection setup.

`dns` - the list of DNS servers to try before reporting failure. The order in which they are interrogated is not specified. Once one result has been obtained, there is no requirement to try others.

### Methods

### getDNSServer()

```
public java.net.InetAddress[] getDNSServer()
```

Return the addresses of the DNS servers to use for the connection.

Returns:

return the addresses of the DNS servers passed into the constructor of the instance or null if none was provided.

### getPassword()

```
public java.lang.String getPassword()
```

Return the password used in establishing this connection The value returned shall be the one passed into the constructor of this instance.

Returns:

the password used in establishing this connection.

### getTarget()

```
public java.lang.String getTarget()
```

Return the target of this connection for example a phone number. The value returned shall be the one passed into the constructor of this instance.

Returns:

the target of the connection.

### getUsername()

```
public java.lang.String getUsername()
```

Return the username used in establishing this connection The value returned shall be the one passed into the constructor of this instance.

Returns:

the username used in establishing the connection.

# org.dvb.net.rc
# ConnectionRCEvent

### Declaration

```
public class ConnectionRCEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.net.rc.ConnectionRCEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Direct Known Subclasses:

```
ConnectionEstablishedEvent, ConnectionFailedEvent, ConnectionTerminatedEvent
```

### Description

ConnectionRCEvent - the base class for events related to connection oriented return channels.

### Constructors

### ConnectionRCEvent(Object)

```
public ConnectionRCEvent(java.lang.Object source)
```

Construct an event.

### Parameters:

`source` - the `ConnectionRCInterface` for which the event was generated.

# org.dvb.net.rc
# ConnectionRCInterface

### Declaration

```
public class ConnectionRCInterface extends RCInterface implements org.davic.resources.ResourceProxy

java.lang.Object
  |
  +--org.dvb.net.rc.RCInterface
        |
        +--org.dvb.net.rc.ConnectionRCInterface
```

### All Implemented Interfaces:

```
org.davic.resources.ResourceProxy
```

### Description

This class models a connection based return channel network interface for use in receiving and transmitting IP packets over a return channel. Targets for connections are specified as strings including the number to dial. These strings can only include either numbers or a "+" character (as the first character only).

When a ConnectionRCInterface instance is first obtained by an application, the current target shall be set to the default. Applications which wish to use a non-default target need to set this target before attempting to reserve the ConnectionRCInterface. This is because if the application does not have the permission to use the default target, the reserve() method is required throw a SecurityException.

### Constructors

#### ConnectionRCInterface()

```
protected ConnectionRCInterface()
```

Constructor for instances of this class. This constructor is provided for the use of implementations and specifications which extend the present document. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

### Methods

#### addConnectionListener(ConnectionListener)

```
public void addConnectionListener(org.dvb.net.rc.ConnectionListener l)
```

Add a listener for events related to connections of this interface.

Parameters:

l - the listener for the connection related events.

#### connect()

```
public void connect()
throws IOException, PermissionDeniedException
```

Connect this return channel to the current target. If this ResourceProxy does not have the underlying resource reserved then a PermissionDeniedException will be thrown. Where the underlying resource is reserved but at the time the method is called, it is known that connection is impossible then an IOException will be thrown. Apart from this, this method is asynchronous and completion or failure is reported through the event listener on this class. If a connection is already established when this method is called then the method call shall have no effect.

Throws:

PermissionDeniedException - if this application does not own the resource.

java.io.IOException - if connection is known to be impossible at the time when the method is called.

#### disconnect()

```
public void disconnect()
throws PermissionDeniedException
```

Disconnect this return channel from the current target. This method is asynchronous and completion is reported through the event listener on this class. This method does not release the underlying resource from the ResourceProxy. If no connection is established then this method shall have no effect.

Throws:

PermissionDeniedException - if this application does not own the resource.

#### getClient()

```
public org.davic.resources.ResourceClient getClient()
```

Return the object which asked to be notified about withdrawal of the underlying resource. This is the object provided as the first parameter to the last call to the reserve method on this object. If this object does not have the underlying resource reserved then null is returned.

Specified By:

getClient in interface ResourceProxy

Returns:

the object which asked to be notified about withdrawal of the underlying physical resource from this resource proxy or null.

### getConnectedTime()

```
public int getConnectedTime()
```

Return the time an interface has been connected.

Returns:

the time in seconds for which this interface has been connected or -1 if the device is not connected.

### getCurrentTarget()

```
public org.dvb.net.rc.ConnectionParameters getCurrentTarget()
throws IncompleteTargetException
```

Get the current target for connections.

If this ConnectionRCInterface is connected then this method shall return the target to which the connection was made. If this ConnectionRCInterface is not connected then this method shall return the last target set by the setTarget method (if any) otherwise the default.

This returns either the default target or the last target set by this application calling the setTarget method on this instance before the connection was established. This applies regardless of whether the connection was established by another GEM application or if some of the connection parameters have been supplied by the server.

Returns:

the current set of connection target parameters.

Throws:

`IncompleteTargetException` - if the current target is not completely configured.

`java.lang.SecurityException` - if the application is not allowed to read the current target as defined by the security policy of the platform.

### getSetupTimeEstimate()

```
public float getSetupTimeEstimate()
```

Obtain an estimate of the setup time for a successful connection for this interface in seconds.

Returns:

an estimate of the setup time for a successful connection for this interface in seconds.

### isConnected()

```
public boolean isConnected()
```

Check if this interface is connected. Connected means able to receive and transmit packets.

Returns:

true if the interface is connected, otherwise false.

### release()

```
public void release()
```

Release the right to control this return channel interface. If this object does not have the right to control this return channel interface then this method shall have no effect.

### removeConnectionListener(ConnectionListener)

```
public void removeConnectionListener(org.dvb.net.rc.ConnectionListener l)
```

Remove a listener for events related to connections of this interface. If the listener specified is not currently receiving these events then this method has no effect.

Parameters:

`l` - the listener for the connection related events.

### reserve(ResourceClient, Object)

```
public void reserve(org.davic.resources.ResourceClient c, java.lang.Object requestData)
throws PermissionDeniedException
```

Request the right to control this return channel interface. If the right to control the return channel interface has already been reserved then this method shall have no effect.

The details of the current connection target shall be obtained from the `ConnectionParameters` instance which is the current target during the call to this method. Hence changes to that `ConnectionParameters` instance before a call to this method shall be taken account of during the method call. Changes after the call to this method shall have no effect on that connection.

Parameters:

`c` - the object to be notified when resources are removed.

`requestData` - Used by the Resource Notification API in the requestRelease method of the ResourceClient interface. The usage of this parameter is optional and a null reference may be supplied.

Throws:

`PermissionDeniedException` - if this interface cannot be reserved.

`java.lang.SecurityException` - if the application is denied access to the resource by security policy.

### setTarget(ConnectionParameters)

```
public void setTarget(org.dvb.net.rc.ConnectionParameters target)
throws IncompleteTargetException, PermissionDeniedException
```

Set a non-default target for connections.

If this method is called for a ConnectionRCInterface which is connected then successful calls to this method shall not interrupt that connection. The newly set target shall just be stored until either the next time a connection is established with that ConnectionRCInterface instance or until a subsequent call to setTarget on that ConnectionRCInterface.

The details of the current connection target shall be obtained from the newly set target during the call to this method. Changes to that instance after the call to this method shall have no effect on the ConnectionRCInterface unless/until set-Target is called again for that instance or it is released and reserved again.

Parameters:

`target` - the new set of connection target parameters.

Throws:

`IncompleteTargetException` - if the application owns the resource but the target is not completely specified.

`PermissionDeniedException` - this exception shall never be thrown.

`java.lang.SecurityException` - if the application is not allowed to modify the target as defined by the security policy of the platform.

### setTargetToDefault()

```
public void setTargetToDefault()
throws PermissionDeniedException
```

Set the target for connections to the default.

Throws:

`PermissionDeniedException` - if this application does not own the resource.

`java.lang.SecurityException` - if the application is not allowed to connect to the default target.

# org.dvb.net.rc ConnectionTerminatedEvent

### Declaration

```
public class ConnectionTerminatedEvent extends ConnectionRCEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.net.rc.ConnectionRCEvent
              |
              +--org.dvb.net.rc.ConnectionTerminatedEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

ConnectionTerminatedEvent - An event generated after a connected `ConnectionRCInterface` is disconnected.

### Constructors

### ConnectionTerminatedEvent(Object)

```
public ConnectionTerminatedEvent(java.lang.Object source)
```

Construct an event.

Parameters:

`source` - the `ConnectionRCInterface` whose status changed.

# org.dvb.net.rc IncompleteTargetException

### Declaration

```
public class IncompleteTargetException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.net.rc.IncompleteTargetException
```

### All Implemented Interfaces:

`java.io.Serializable`

### Description

Thrown when the target for a connection is incompletely specified. This is thrown either when the default connection target is incompletely defined in the device or when an application provides an incompletely defined connection target to the device or when the connection target is badly formed, e.g. includes illegal characters in a number parameter.

### Constructors

### IncompleteTargetException()

```
public IncompleteTargetException()
```

Default constructor for the exception

### IncompleteTargetException(String)

```
public IncompleteTargetException(java.lang.String reason)
```

Constructor for the exception with a specified reason.

### Parameters:

`reason` - the reason why the exception was raised.

# org.dvb.net.rc

# NoDialToneEvent

### Declaration

```
public class NoDialToneEvent extends ConnectionFailedEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.net.rc.ConnectionRCEvent
              |
              +--org.dvb.net.rc.ConnectionFailedEvent
                    |
                    +--org.dvb.net.rc.NoDialToneEvent
```

### All Implemented Interfaces:

`java.io.Serializable`

### Description

NoDialToneEvent - An event generated after an attempt to setup a connection for a `ConnectionRCInterface` of `TYPE_PSTN` fails due to there not being a dial tone on the return channel concerned.

### Constructors

### NoDialToneEvent(Object)

```
public NoDialToneEvent(java.lang.Object source)
```

Construct an event.

### Parameters:

`source` - the `ConnectionRCInterface` whose connection attempt failed.

# org.dvb.net.rc PermissionDeniedException

### Declaration

```
public class PermissionDeniedException extends java.lang.Exception

java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.net.rc.PermissionDeniedException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown when an application calls a method which it does not have permission to call at that time.

### Constructors

#### PermissionDeniedException()

```
public PermissionDeniedException()
```

Default constructor for the exception.

#### PermissionDeniedException(String)

```
public PermissionDeniedException(java.lang.String reason)
```

Constructor for the exception with a specified reason.

### Parameters:

`reason` - the reason why the exception was raised.

# org.dvb.net.rc RCInterface

### Declaration

```
public class RCInterface

java.lang.Object
  |
  +--org.dvb.net.rc.RCInterface
```

### Direct Known Subclasses:

```
ConnectionRCInterface
```

### Description

This class models a return channel network interface for use in receiving and transmitting IP packets over a logical return channel. This can include real analog modems, cable return channel and all the other options allowed by the relevant DVB specification. This class does not model any concept of connection. Hence interfaces represented by this class and not by a sub-class of it are permanently connected.

### Fields

#### TYPE_CATV

```
public static final int TYPE_CATV
```

Constant to indicate a CATV return channel.

#### TYPE_DECT

```
public static final int TYPE_DECT
```

Constant to indicate a DECT return channel.

#### TYPE_ISDN

```
public static final int TYPE_ISDN
```

Constant to indicate an ISDN return channel.

#### TYPE_LMDS

```
public static final int TYPE_LMDS
```

Constant to indicate a LMDS return channel.

#### TYPE_MATV

```
public static final int TYPE_MATV
```

Constant to indicate a MATV return channel.

#### TYPE_OTHER

```
public static final int TYPE_OTHER
```

Constant to indicate all other return channel technologies not having a suitable defined constant in this class.

NOTE: DVB does not intend to add future constants to this list for future return channel technologies. These should be represented as TYPE_OTHER.

#### TYPE_PSTN

```
public static final int TYPE_PSTN
```

Constant to indicate a PSTN return channel.

#### TYPE_RCS

```
public static final int TYPE_RCS
```

Constant to indicate a DVB-RCS return channel.

#### TYPE_UNKNOWN

```
public static final int TYPE_UNKNOWN
```

Constant to indicate an unknown return channel technology. There is an intermediate physical interface between the GEM terminal and the return channel device. This return value gives no information about whether the return channel is connection oriented or connectionless.

### Constructors

#### RCInterface()

```
protected RCInterface()
```

Constructor for instances of this class. This constructor is provided for the use of implementations and specifications which extend the present document. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

### Methods

#### getDataRate()

```
public int getDataRate()
```

Return the maximum data rate of the connection over the immediate access network to which this network interface is connected. For asymmetric connections, the data rate coming into the GEM terminal shall be returned. For connection oriented interfaces which are not currently connected, the value returned shall be that of the last connection established where that information is available. Where that information is not available, (e.g. where no connection has been established since a GEM terminal was power cycled), -1 shall be returned.

Returns:

a data rate in KBaud or -1 where this is not available.

Since:

MHP 1.0.1.

#### getType()

```
public int getType()
```

Return the type of return channel represented by this object. Note, applications wishing to discover whether a return channel interface is connection oriented or not are recommended to test whether an object is an instance of `ConnectionRCInterface` or not. A non-connection oriented interface really means a permanently connected return channel.

Returns:

the type of return channel represented by this object encoded as one of the constants defined in this class.

# org.dvb.net.rc

# RCInterfaceManager

### Declaration

```
public class RCInterfaceManager implements org.davic.resources.ResourceServer
```

```
java.lang.Object
  |
  +--org.dvb.net.rc.RCInterfaceManager
```

### All Implemented Interfaces:

```
org.davic.resources.ResourceServer
```

### Description

This class is the factory and manager for all return channel interfaces in the system. The methods on this class which return instances of the `RCInterface` will only return new instances of that class under the following conditions:

- on the first occasion an instance needs to be returned to a particular application for a particular interface;
- when new return channel interfaces are added to the system.

### Methods

#### addResourceStatusEventListener(ResourceStatusListener)

```
public void addResourceStatusEventListener(org.davic.resources.ResourceStatusListener listener)
```

This method informs a resource server that a particular object should be informed of changes in the state of the resources managed by that server.

Specified By:

addResourceStatusEventListener in interface ResourceServer

Parameters:

listener - the object to be informed of state changes.

getInstance()

```
public static org.dvb.net.rc.RCInterfaceManager getInstance()
```

Factory method to obtain a manager. The RCInterfaceManager is either a singleton for each GEM application or a singleton for the GEM terminal.

Returns:

an instance of an RCInterfaceManager.

getInterface(InetAddress)

```
public org.dvb.net.rc.RCInterface getInterface(java.net.InetAddress addr)
```

Return the interface which will be used when connecting to a particular host. Null is returned if this is not known when the method is called.

Parameters:

addr - the IP address of the host to connect to.

Returns:

the interface which will be used or null if this is not known.

getInterface(Socket)

```
public org.dvb.net.rc.RCInterface getInterface(java.net.Socket s)
```

Return the interface which is used for a particular socket.

Parameters:

s - the socket to use.

Returns:

the interface which is used or null if the socket is not connected.

getInterface(URLConnection)

```
public org.dvb.net.rc.RCInterface getInterface(java.net.URLConnection u)
```

Return the interface which is used for a particular URLConnection.

Parameters:

u - the URLConnection to use.

Returns:

the interface which is used or null if the URLConnection is not connected.

getInterfaces()

```
public org.dvb.net.rc.RCInterface[] getInterfaces()
```

Factory method to return a list of all return channel interfaces visible to this application. The number of entries in the array will exactly match the number of return channel interfaces visible to the application. Null is returned if no interfaces are visible to this application.

Returns:

an array of available return channel interfaces.

removeResourceStatusEventListener(ResourceStatusListener)

```
public void removeResourceStatusEventListener(org.davic.resources.ResourceStatusListener listener)
```

This method informs a resource server that a particular object is no longer interested in being informed about changes in state of resources managed by that server. If the object had not registered its interest initially then this method has no effect.

Specified By:

`removeResourceStatusEventListener` in interface `ResourceServer`

Parameters:

`listener` - the object which is no longer interested.

# org.dvb.net.rc

# RCInterfaceReleasedEvent

Declaration

```
public class RCInterfaceReleasedEvent extends org.davic.resources.ResourceStatusEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.davic.resources.ResourceStatusEvent
              |
              +--org.dvb.net.rc.RCInterfaceReleasedEvent
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

This event informs an application that a `RCInterface` has been released by an application or other entity in the system. It is generated when an application which had successfully reserved a `RCInterface` calls the `Connection-RCInterface.release` method. It will also be generated if any other entities in the system own such an interface and then release that interface in such a way that it could then become available to applications using this API.

Constructors

RCInterfaceReleasedEvent(Object)

```
public RCInterfaceReleasedEvent(java.lang.Object bg)
```

Constructor for the event.

Parameters:

`bg` - the `RCInterface` which has been released.

Methods

getSource()

```
public java.lang.Object getSource()
```

Returns the device that has been released.

Overrides:

`getSource` in class `ResourceStatusEvent`

Returns:

the `RCInterface` object representing the interface that has been released.

# org.dvb.net.rc

# RCInterfaceReservedEvent

Declaration

```
public class RCInterfaceReservedEvent extends org.davic.resources.ResourceStatusEvent

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.davic.resources.ResourceStatusEvent
              |
              +--org.dvb.net.rc.RCInterfaceReservedEvent
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

This event informs an application that a `RCInterface` has been reserved by an application or other entity in the system. It is generated when an application successfully reserves a `RCInterface`. It will also be generated if any other entities in the system reserve such an interface with the effect of something which was visible to applications using this API becoming unavailable.

Constructors

RCInterfaceReservedEvent(Object)

```
public RCInterfaceReservedEvent(java.lang.Object bg)
```

Constructor for the event.

Parameters:

`bg` - the `RCInterface` representing the device which has been reserved.

Methods

getSource()

```
public java.lang.Object getSource()
```

Returns the device that has been reserved.

Overrides:

`getSource` in class `ResourceStatusEvent`

Returns:

an `RCInterface` representing the device that has been reserved.

# org.dvb.net.rc

# RCPermission

### Declaration

```
public class RCPermission extends java.security.BasicPermission

java.lang.Object
  |
  +--java.security.Permission
        |
        +--java.security.BasicPermission
              |
              +--org.dvb.net.rc.RCPermission
```

### All Implemented Interfaces:

```
java.security.Guard, java.io.Serializable
```

### Description

This class is for return channel set-up permissions. An RCPermission contains a name, but no actions list.

The permission name can be "target:default", which indicates the permission to use the default connection parameters.

The permission name can also be "target:<phone number>", which indicates the permission to use the specified phone number in the connection set-up (ConnectionRCInterface.setTarget(ConnectionParameters) method).

A wildcard may be used at the end of the permission name. In that case, all phone numbers starting with the number before the wildcard are included in the permission. A "+" may be used at the start of the phone number to indicate a phone number including the international country code.

EXAMPLES:

- target:0206234342 (Permission to dial the specified phone number).
- target:020* (Permission to dial phone numbers starting with 020).
- target:* (Permission to dial all phone numbers, including the default).

NOTE: ConnectionRCInterface.reserve(ResourceClient, Object) will throw a SecurityException if the application is not allowed to set-up a connection over the return channel at all (i.e., there is no valid target allowed).

### Constructors

#### RCPermission(String)

```
public RCPermission(java.lang.String name)
```

Creates a new RCPermission with the specified name. The name is the symbolic name of the RCPermission.

#### Parameters:

`name` - the name of the RCPermission.

#### RCPermission(String, String)

```
public RCPermission(java.lang.String name, java.lang.String actions)
```

Creates a new RCPermission object with the specified name. The name is the symbolic name of the RCPermission, and the actions String is unused and should be null. This constructor exists for use by the Policy object to instantiate new Permission objects.

Parameters:

`name` - the name of the RCPermission.

`actions` - should be null.

Methods

implies(Permission)

```
public boolean implies(java.security.Permission p)
```

Checks if this RCPermission "implies" the specified Permission.

More specifically, this returns true if and only if:

- p is an instance of RCPermission; and
- p's name is implied by the name of this permission, as described by the wildcarding rules specified in the the description of this class.

Overrides:

`implies` in class `BasicPermission`.

Parameters:

`p` - The Permission to check against.

Returns:

true if the specified Permission is implied by this object; false otherwise.

# org.dvb.net.rc

# TargetBusyEvent

Declaration

```
public class TargetBusyEvent extends ConnectionFailedEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.net.rc.ConnectionRCEvent
              |
              +--org.dvb.net.rc.ConnectionFailedEvent
                    |
                    +--org.dvb.net.rc.TargetBusyEvent
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

TargetBusyEvent - An event generated after an attempt to setup a connection for a `ConnectionRCInterface` fails due to the target of the connection being busy.

Constructors

TargetBusyEvent(Object)

```
public TargetBusyEvent(java.lang.Object source)
```

Construct an event.

Parameters:

`source` - the `ConnectionRCInterface` whose connection attempt failed.

# Annex S (normative): Application listing and launching

# Package

# org.dvb.application

Description

Provides access to lists of applications which are available in this context and the ability to launch those applications.

| Class Summary | |
|---|---|
| **Interfaces** | |
| AppAttributes | The AppAttributes class is a mapping of various information about a registered application. |
| AppProxy | An AppProxy Object is a proxy to an application. |
| AppsDatabaseEventListener | The AppsDatabaseListener class allows an application to monitor the application database so that it can keep an up to date interface without polling the state. |
| AppStateChangeEventListener | The AppStateChangeEventListener class allows a launcher application to keep track of applications it launches or other applications running as part of the same service. |
| DVBJProxy | A DVBJProxy Object is a proxy to a DVBJ application. |
| **Classes** | |
| AppIcon | The AppIcon encapsulates the information concerning the icon attached to the application |
| AppID | The AppID is a representation of the unique identifier for applications. |
| AppsControlPermission | This class represents a Permission to control the lifecycle of another application. |
| AppsDatabase | The AppsDatabase is an abstract view of the currently available applications. |
| AppsDatabaseEvent | The AppsDatabaseEvent class indicates either an entry in the application database has changed, or so many changes have occurred. |
| AppsDatabaseFilter | Abstract class for the filters. |
| AppStateChangeEvent | The AppStateChangeEvent class indicates a state transition of the application. |
| CurrentServiceFilter | Instances of CurrentServiceFilter are used to set a filter on the list of applications that are retrieved from the AppsDatabase (See methods getApps-Attributes and getAppsIDs). |
| RunningApplicationsFilter | Instances of RunningApplicationsFilter are used to set a filter on the list of applications that are retrieved from the AppsDatabase (See methods get-AppsAttributes and getAppsIDs). |
| **Exceptions** | |
| IllegalProfileParameterException | The IllegalProfileParameter exception is thrown if the application attempts to ask for a version number for a profile not specified for the application. |
| LanguageNotAvailableException | The LanguageNotAvailableException exception is thrown if the application asks for the name of an application in a language not signalled in the AIT. |

# org.dvb.application

# AppAttributes

### Declaration

```
public interface AppAttributes
```

### All Known Subinterfaces:

```
org.dvb.application.storage.ExtendedAppAttributes
```

### Description

The `AppAttributes` class is a mapping of various information about a registered application. For applications which are signalled in an AIT, the mapping between the values returned by methods in this class and the fields and descriptors of the AIT shall be as specified in the main body of the present document.

Instances of objects implementing this interface are immutable and populated before the instance is first returned to an application.

### Since:

MHP1.0.

### Fields

### DVB_HTML_application

```
public static final int DVB_HTML_application
```

The DVB registered value for all DVB-HTML applications.

### DVB_J_application

```
public static final int DVB_J_application
```

The DVB registered value for all DVB-J applications.

### Methods

### getAppIcon()

```
public org.dvb.application.AppIcon getAppIcon()
```

This method returns an object encapsulating the information about the icon(s) for the application.

### Returns:

the information related to the icons that are attached to the application or null if no icon information is available.

### Since:

MHP1.0.

### getIdentifier()

```
public org.dvb.application.AppID getIdentifier()
```

This method returns the application identifier.

### Returns:

the application identifier.

Since:

MHP1.0.

### getIsServiceBound()

```
public boolean getIsServiceBound()
```

This method determines whether the application is bound to a single service.

Returns:

true if the application is bound to a single service, false otherwise.

Since:

MHP1.0.

### getName()

```
public java.lang.String getName()
```

This method returns the name of the application. If the default language (as specified in user preferences) is in the set of available language / name pairs then the name in that language shall be returned. Otherwise this method will return a name which appears in that set on a "best-effort basis". If no application names are signalled, an empty string shall be returned.

Returns:

the name of the application.

Since:

MHP1.0.

### getName(String)

```
public java.lang.String getName(java.lang.String iso639code)
throws LanguageNotAvailableException
```

This method returns the name of the application in the language which is specified by the parameter passed as an argument. If the language specified is not in the set of available language /name pairs then an exception shall be thrown.

Parameters:

`iso639code` - the specified language, encoded as per ISO 639 [84].

Returns:

returns the name of the application in the specified language.

Throws:

`LanguageNotAvailableException` - if the name is not available in the language specified or if the parameter passed is null.

Since:

MHP1.0.

### getNames()

```
public java.lang.String[][] getNames()
```

This method returns all the available names for the application together with their ISO 639 [84] language code. If no application names are signalled, an array of length zero shall be returned.

Returns:

the possible names of the application, along with their ISO 639 [84] language code. The first string in each sub-array is the ISO 639 [84] language code. The second string in each sub-array is the corresponding application name.

Since:

MHP1.0.

getPriority()

```
public int getPriority()
```

This method returns the priority of the application.

Returns:

the priority of the application.

Since:

MHP1.0.

getProfiles()

```
public java.lang.String[] getProfiles()
```

This method returns those minimum profiles required for the application to execute. Profile names shall be encoded using the same encoding specified elsewhere in this specification as input for use with the `java.lang.System.get-Property` method to query if a profile is supported by this platform.

For example, for implementations conforming to the first version of the specification, the translation from AIT signaling values to strings shall be as follows:

- '1' in the signaling will be translated into 'mhp.profile.enhanced_broadcast';
- '2' in the signaling will be translated into 'mhp.profile.interactive_broadcast'.

Only profiles supported by this particular GEM terminal shall be returned. Hence the method can return an array of size zero where all the profiles on which an application can execute are unknown.

Returns:

an array of Strings, each String describing a profile.

Since:

MHP1.0.

getProperty(String)

```
public java.lang.Object getProperty(java.lang.String index)
```

The following method is included for properties that do not have explicit property accessors. The naming of properties and their return values are described in the main body of the present document.

Parameters:

`index` - a property name.

Returns:

either the return value corresponding to the property name or null if the property name is unknown or null.

Since:

MHP1.0.

### getServiceLocator()

```
public org.davic.net.Locator getServiceLocator()
```

This method returns the locator of the Service describing the application. For an application transmitted on a remote connection, the returned locator shall be the service for that remote connection. For applications not transmitted on a remote connection, the service returned shall be the currently selected service of the service context within which the application calling the method is running.

Returns:

the locator of the Service describing the application.

Since:

MHP1.0.

### getType()

```
public int getType()
```

This method returns the type of the application (as registered by DVB).

Returns:

the type of the application (as registered by DVB).

Since:

MHP1.0.

### getVersions(String)

```
public int[] getVersions(java.lang.String profile)
throws IllegalProfileParameterException
```

This method returns an array of integers containing the version number of the specification required to run this application at the specified profile.

Parameters:

`profile` - a profile encoded as defined in the clause "Profile and version properties" in the main body of the present document. e.g. mhp.profile.interactive_broadcast for the interactive broadcast profile.

Returns:

an array of integers, containing the major, minor and micro values (in that order) required for the specified profile.

Throws:

`IllegalProfileParameterException` - thrown if the profile specified is not one of the minimum profiles required for the application to execute or if the parameter passed in is null.

Since:

MHP1.0.

### isStartable()

```
public boolean isStartable()
```

This method determines whether the application is startable or not. An Application is not startable if any of the following apply.

- The application is transmitted on a remote connection, and either does not have an application storage descriptor, is not cached, or is not signalled as launchable completely from cache.

- The application is signalled as not_launchable_from_broadcast and is not stored at the time the AppAttributes instance is first returned to an application.
- The caller of the method does not have the Permissions to start it.
- if the application is signalled with a control code which is neither AUTOSTART nor PRESENT.

If none of the above apply, then the application is startable.

The value returned by this method does not depend on whether the application is actually running or not.

Returns:

true if an application is startable, false otherwise.

Since:

MHP1.0.

### isVisible()

```
public boolean isVisible()
```

This method determines whether the application is marked as being visible to users. An inter-operable application shall honour this visibility setting. Thus a generic launching application shall list applications that are marked as visible and shall not list applications that are not marked as visible.

Returns:

`true` if this application is marked as being visible to users, `false` otherwise.

Since:

MHP1.0.3.

# org.dvb.application

# AppIcon

### Declaration

```
public class AppIcon

java.lang.Object
  |
  +--org.dvb.application.AppIcon
```

### Description

The `AppIcon` encapsulates the information concerning the icon attached to the application.

### Constructors

### AppIcon()

```
protected AppIcon()
```

The constructor for the class. This constructor is intended for implementation convenience and evolution of the specification and not for use by GEM applications. Applications should obtain instances of this class from `App-Attributes.getAppIcon`.

See Also:

```
AppAttributes.getAppIcon()
```

### Methods

#### getIconFlags()

```
public java.util.BitSet getIconFlags()
```

This method returns the flags identifying which icons are provided for the application.

Returns:

the icon flags encoded as a BitSet.

Since:

MHP1.0.

#### getLocator()

```
public org.davic.net.Locator getLocator()
```

This method returns the location of the directory containing the application icons.

Returns:

the location of the directory containing the application icons.

Since:

MHP1.0.

# org.dvb.application

# AppID

### Declaration

```
public class AppID

java.lang.Object
  |
  +--org.dvb.application.AppID
```

### Description

The `AppID` is a representation of the unique identifier for applications.

Its string form is the Hex representation of the 48 bit number.

### Constructors

#### AppID(int, int)

```
public AppID(int oid, int aid)
```

Create a new AppID based on the given integers. There is no range checking on these numbers.

Parameters:

`oid` - the globally unique organization number.

`aid` - the unique count within the organization.

Since:

MHP1.0.

### Methods

#### equals(Object)

```
public boolean equals(java.lang.Object obj)
```

Compares two AppIDs for equality.

Overrides:

`equals` in class `Object`

Parameters:

`obj` - the reference object with which to compare.

Returns:

`true` if this obj is an AppID and its Organisation ID and its Application ID match the IDs for this AppID; `false` otherwise.

#### getAID()

```
public int getAID()
```

This method returns the integer value of the application count supplied in the constructor.

Returns:

the integer value of the application count supplied in the constructor.

Since:

MHP1.0.

#### getOID()

```
public int getOID()
```

This method returns the integer value of the organization number supplied in the constructor.

Returns:

the integer value of the organization number supplied in the constructor.

Since:

MHP1.0.

#### hashCode()

```
public int hashCode()
```

Returns a hash code value for this AppID. The hashcode for two AppIDs with the same Organisation ID and Application ID are equal.

Overrides:

`hashCode` in class `Object`

Returns:

a hash code value for this AppID.

#### toString()

```
public java.lang.String toString()
```

This method returns a string containing the Hex representation of the 48 bit number. The string shall be formatted as specified in the clause on "Text encoding of application identifiers" in the System Integration clause of the GEM specification.

Overrides:

`toString` in class `Object`

Returns:

a string containing the Hex representation of the 48 bit number.

Since:

MHP1.0.

# org.dvb.application

# AppProxy

Declaration

```
public interface AppProxy
```

All Known Subinterfaces:

`DVBJProxy`

Description

An `AppProxy` Object is a proxy to an application. A call to the start, stop or pause will cause the resident Application Manager to respectively start, stop or pause the application bound to this AppProxy object. Each of these three method calls can throw a Security Exception if the calling application is not entitled to do so.

Each of these method calls are asynchronous and will result in exactly one AppStateChangeEvent to be generated whether the method call was successful or not. If the method call was not successful, any call to the `hasFailed` method of the corresponding `AppStateChangeEvent` will return true.

Some of the methods here allow the AppProxy to transition through several states before the final state is reached. If this compound state transition is unsuccessful at any point, the resulting AppStateChangeEvent shall have a fromstate which is the last state in this transition which the AppProxy successfully entered and a toState which would have been the next state in the compound state transition.

For instance, if an application were to call start on an AppProxy for a DVB-J application in the NOT_LOADED state and that DVB-J application was to throw a XletStateChangeException from its startXlet method, the getFromState will return PAUSED and getToState will return STARTED. If an application were to call start on an AppProxy for a DVB-J application in the NOT_LOADED state and that DVB-J application was to throw a XletStateChangeException from its initXlet method, the getFromState will return NOT_LOADED and getToState will return PAUSED. Calling the start method for an application which is already running shall fail and generate an AppStateChangeEvent with hasFailed returning true and both fromstate and tostate being STARTED.

See the definition of `AppStateChangeEvent` for more information. Having more than one application attempt to control the lifecycle of the same (other) application at the same time is not recommended. The ordering of AppState-ChangeEvents if this is done will be implementation dependent.

See Also:

`AppStateChangeEvent`

Fields

DESTROYED

```
public static final int DESTROYED
```

The application is in the destroyed state. 1 The application instance is in the destroyed state and remains in this state. An AppProxy for a new instance of this application may be obtained from the AppsDatabase and started.

### INVALID

```
public static final int INVALID
```

The application is not signalled in the currently selected service and hence not listed in the applications database. Once an AppProxy is in this state, all attempts to cause further state transitions on that AppProxy instance shall fail with an AppStateChangeEvent from INVALID to INVALID, even if a service is selected where that application is signalled.

### NOT_LOADED

```
public static final int NOT_LOADED
```

The application has not yet been loaded from the network at all.

### PAUSED

```
public static final int PAUSED
```

The application is in the paused state.

### STARTED

```
public static final int STARTED
```

The application is in the active state.

### Methods

### addAppStateChangeEventListener(AppStateChangeEventListener)

```
public void addAppStateChangeEventListener(org.dvb.application.AppStateChangeEventListener listener)
```

Add a listener to the application proxy so that an application can be informed if the application changes state.

**Parameters:**

`listener` - the listener to be added.

**Since:**

MHP1.0.

### getState()

```
public int getState()
```

Return the current state of the application.

**Returns:**

the state of the application.

### pause()

```
public void pause()
```

Request that the application manager pause the application bound to this information structure.

The `application` will be paused. Calls to this method shall fail if the application is not in the active state. If the application represented by this AppProxy is a DVB-J application, calling this method will, if successful, result in the `pauseXlet` method being called on the Xlet making up the DVB-J application.

**Throws:**

`java.lang.SecurityException` - if the application is not entitled to pause this application. Note that if an application is entitled to stop an application, it is also entitled to pause it: having the right to stop an application is logically equivalent to having the right to pause it.

Since:

MHP1.0.

### removeAppStateChangeEventListener(AppStateChangeEventListener)

```
public void removeAppStateChangeEventListener(org.dvb.application.AppStateChangeEventListener
listener)
```

Remove a listener on the database.

Parameters:

`listener` - the listener to be removed.

Since:

MHP1.0.

### resume()

```
public void resume()
```

Request that the application manager resume the execution of the application. The `application` will be started. This method will throw a security exception if the application does not have the authority to resume the application. Calls to this method shall fail if the application is not in the paused state.

This method is asynchronous and its completion will be notified by an AppStateChangeEvent. In case of failure, the hasFailed method of the `AppStateChangeEvent` will return true. If the application represented by this AppProxy is a DVB-J application, calling this method will, if successful, result in the `startXlet` method being called on the Xlet making up the DVB-J application.

Throws:

`java.lang.SecurityException` - if the application is not entitled to resume this application.

Since:

MHP1.0.

### start()

```
public void start()
```

Request that the application manager start the application bound to this information structure.

The `application` will be started. This method will throw a security exception if the application does not have the authority to start applications. Calls to this method shall only succeed if the application if the application is signalled with a control code which is either AUTOSTART or PRESENT and any one of the following applies:

- if the application (DVB-J or DVB-HTML) is in the not loaded or paused states;
- if a DVB-J application is in the "loaded" state;
- if a DVB-HTML application is in the "loading" state.

If the application was not loaded at the moment of this call, then the application will be started. In the case of a DVB-J application, it will be initialized and then started by the Application Manager, hence causing the Xlet to go from Not-Loaded to Paused and then from Paused to Active. If the application was in the Paused state at the moment of the call and had never been in the Active state, then the application will be started. If the application represented by this App-Proxy is a DVB-J application, calling this method will, if successful, result in the `startXlet` method being called on the Xlet making up the DVB-J application.

This method is asynchronous and its completion will be notified by an AppStateChangeEvent. In case of failure, the hasFailed method of the `AppStateChangeEvent` will return true.

Throws:

`java.lang.SecurityException` - if the application is not entitled to start this application.

Since:

MHP1.0.

### start(String[])

```
public void start(java.lang.String[] args)
```

Request that the application manager start the application bound to this information structure passing to that application the specified parameters.

The `application` will be started. This method will throw a security exception if the application does not have the authority to start applications. Calls to this method shall only succeed if the application if the application is signalled with a control code which is either AUTOSTART or PRESENT and any one of the following applies:

- if the application (DVB-J or DVB-HTML) is in the not loaded or paused states;
- if a DVB-J application is in the "loaded" state;
- if a DVB-HTML application is in the "loading" state.

If the application was not loaded at the moment of this call, then the application will be started. In the case of a DVB-J application, it will be initialized and then started by the Application Manager, hence causing the Xlet to go from Not-Loaded to Paused and then from Paused to Active. If the application was in the Paused state at the moment of the call and had never been in the Active state, then the application will be started. If the application represented by this App-Proxy is a DVB-J application, calling this method will, if successful, result in the `startXlet` method being called on the Xlet making up the DVB-J application.

This method is asynchronous and its completion will be notified by an AppStateChangeEvent. In case of failure, the hasFailed method of the `AppStateChangeEvent` will return true. If this AppProxy is a DVBJProxy and if init(String[]) has previously been called for this application instance, then the parameters are not passed into the application but are silently discarded. Those passed into init take precedence.

Parameters:

`args` - the parameters to be passed into the application being started.

Throws:

`java.lang.SecurityException` - if the application is not entitled to start this application.

Since:

MHP1.0.1.

### stop(boolean)

```
public void stop(boolean forced)
```

Request that the application manager stop the application bound to this information structure.

The `application` will be stopped. A call to this method shall fail if the application was already in the destroyed state. This method call will stop the application if it was in any other state before the call. If the application is in the NOT_LOADED state then it shall move directly to the DESTROYED state with no other action being taken. If the application represented by this AppProxy is a DVB-J application and is not in the DESTROYED state then calling this method will, if successful, result in the `destroyXlet` method being called on the Xlet making up the DVB-J application with the same value for the parameter as passed to this method.

This method is asynchronous and its completion will be notified by an AppStateChangeEvent. In case of failure, the hasFailed method of the `AppStateChangeEvent` will return true.

Parameters:

`forced` - if true then do not ask the application but forcibly terminate it, if false give the application an opportunity to refuse.

Throws:

`java.lang.SecurityException` - if the application is not entitled to stop this application.

Since:

MHP1.0.

# org.dvb.application

# AppsControlPermission

## Declaration

```
public final class AppsControlPermission extends java.security.BasicPermission
```

```
java.lang.Object
  |
  +--java.security.Permission
        |
        +--java.security.BasicPermission
              |
              +--org.dvb.application.AppsControlPermission
```

## All Implemented Interfaces:

```
java.security.Guard, java.io.Serializable
```

## Description

This class represents a Permission to control the lifecycle of another application.

## Constructors

### AppsControlPermission()

```
public AppsControlPermission()
```

Creates a new AppsControlPermission. There is a simple mapping between the Application control Permission requests and the way the AppsControlPermission are granted. This mapping is defined in the main body of the present document.

### AppsControlPermission(String, String)

```
public AppsControlPermission(java.lang.String name, java.lang.String actions)
```

Creates a new AppsControlPermission. There is a simple mapping between the Application control Permission requests and the way the AppsControlPermission are granted. This mapping is defined in the main body of the present document. The actions string is currently unused and should be null. The name string is currently unused and should be empty. This constructor exists for use by the java.security.Policy object to instantiate new permission objects.

Parameters:

`name` - the name of the permission.

`actions` - the actions string.

## Methods

### equals(Object)

```
public boolean equals(java.lang.Object obj)
```

Checks for equality against this AppsControlPermission object.

Overrides:

`equals` in class `BasicPermission`

Parameters:

`obj` - the object to test for equality with this AppsControlPermission object.

Returns:

true if and only if obj is an AppsControlPermission.

### getActions()

```
public java.lang.String getActions()
```

Returns the list of actions that had been passed to the constructor - it shall return null.

Overrides:

`getActions` in class `BasicPermission`

Returns:

a null String.

### hashCode()

```
public int hashCode()
```

Returns the hash code value for this object.

Overrides:

`hashCode` in class `BasicPermission`

Returns:

the hash code value for this object.

### implies(Permission)

```
public boolean implies(java.security.Permission permission)
```

Checks if this AppsControlPermission object "implies" the specified permission.

Overrides:

`implies` in class `BasicPermission`

Parameters:

`permission` - the specified permission to check.

Returns:

true if and only if the specified permission is an instanceof AppsControlPermission.

# org.dvb.application

# AppsDatabase

## Declaration

```
public class AppsDatabase

java.lang.Object
  |
  +--org.dvb.application.AppsDatabase
```

## Description

The `AppsDatabase` is an abstract view of the currently available applications. The entries will be provided by the application manager, and gleaned from the AIT signaling. When the service context in which an application is running undergoes service selection, instances of `AppsDatabase` used by that application shall be updated from the new service before an `AppsDatabaseEvent` is sent to the `newDatabase` method of any registered `AppsDatabaseEventListeners`. For applications fully signalled in the current service (i.e. excluding externally authorised ones), the attributes entries shall be the ones from the signalling of the current service even if the application was originally launched from another service and then survived service selection. For running externally authorised applications, the entries will be those from the last service in which they ran fully signalled.

Externally authorized applications shall not appear unless an instance of that application is actually running.

A generic launcher may be written which uses the database to display information in AppAttributes and uses an App-Proxy to launch it

Methods on classes in this package do not block, they return the information the system currently has. Therefore applications should be aware that data may be stale, to within one refresh period of the AIT.

e.g.:

```
AppsDatabase theDatabase = AppsDatabase.getAppsDatabase();
if (theDatabase != null ) {
    CurrentServiceFilter filter = new CurrentServiceFilter();
    Enumeration attributes = theDatabase.getAppAttributes(filter)
    if(attributes != null) {
       while(attributes.hasMoreElements()) {
           AppAttributes info ;
           AppProxy proxy ;

           info = (AppAttributes)attributes.nextElement();
           proxy = (AppProxy)theDatabase.getAppProxy(info.getIdentifier());
           AppIcon icon = info.getAppIcon();
           // blah blah..
           // lets start it.
           proxy.start();
       }
    }
}
```

Where methods on this class as specified as working on "available" applications or "currently available" applications the following definition shall apply. An application is "currently available" if and only if one of the following applies in the service context within which the application calling the method is executing and the visibility of the application is not '00'.

- It is signalled as being present or autostart in the currently selected service of that service context and could be started.
- It is signalled as being present or autostart in the currently selected service of that service context, it could be started if it was stored or cached on the terminal, it is signalled such that it could be stored or cached, and the GEM terminal supports application storage (i.e. the `mhp.stored.services` property is `"SUPPORTED"`). (Note this is **not** affected by whether or not the GEM terminal has sufficient storage space to store the application in question).

- It is currently running in that service context.

In addition to the methods listed below, all calls made using an AppsDatabaseFilter shall only use that filter to test "currently available" applications as defined here.

Applications whose information (e.g. signaling) is invalid (e.g. one or more mandatory descriptors are missing or incorrect) may not be listed in the AppsDatabase. Where applications are signalled in a broadcast AIT and the GEM terminal tunes away from the service on which the AIT is carried, but without selecting a new service, the `AppsDatabase` shall retain the entries as signalled in that AIT until a new service is selected.

### Constructors

#### AppsDatabase()

```
protected AppsDatabase()
```

This constructor is provided for the use of implementations and specifications which extend the present document. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

### Methods

#### addListener(AppsDatabaseEventListener)

```
public void addListener(org.dvb.application.AppsDatabaseEventListener listener)
```

Add a listener to the database so that an application can be informed if the database changes.

Parameters:

`listener` - the listener to be added.

Since:

MHP1.0.

#### getAppAttributes(AppID)

```
public org.dvb.application.AppAttributes getAppAttributes(org.dvb.application.AppID key)
```

Returns the properties associated with the given ID. Returns null if no such application is available.

Only one AppAttributes object shall be returned in the case where there are several applications having the same (organisationId, applicationId) pair. In such a case, the same algorithm as would be used to autostart such applications shall be used to decide between the available choices by the implementation.

This method shall return instances which reflect the contents of the database at the time the method is called. After an AppsDatabaseEvent has been generated, new instances may be returned. After a service selection has taken place, applications which survived the service selection may call this method in order to discover the attributes of the applications signalled on the new service.

Parameters:

`key` - an application ID.

Returns:

the value to which the key is mapped in this dictionary if AppId corresponds to an application which is either a currently available application or remote application or both. Null otherwise.

Since:

MHP1.0.

#### getAppAttributes(AppsDatabaseFilter)

```
public java.util.Enumeration getAppAttributes(org.dvb.application.AppsDatabaseFilter filter)
```

Returns an enumeration of AppAttributes of the applications available. The Enumeration will contain the set of App-Attributes that satisfy the filtering criteria.

This method shall return instances which reflect the contents of the database at the time the method is called. After an AppsDatabaseEvent has been generated, new instances may be returned. After a service selection has taken place, applications which survived the service selection may call this method in order to discover the attributes of the applications signalled on the new service.

This method will return an empty Enumeration if there are no attributes.

Parameters:

`filter` - the filter to apply.

Returns:

an enumeration of the applications attributes.

Since:

MHP1.0.

### getAppIDs(AppsDatabaseFilter)

```
public java.util.Enumeration getAppIDs(org.dvb.application.AppsDatabaseFilter filter)
```

Returns an enumeration of the application IDs available. The Enumeration will contain the set of AppID that match the filtering criteria.

This method shall return instances which reflect the contents of the database at the time the method is called. After an AppsDatabaseEvent has been generated, new instances may be returned. After a service selection has taken place, applications which survived the service selection may call this method in order to discover the identities of the applications signalled on the new service.

This method will return an empty Enumeration if there are no matching applications.

Parameters:

`filter` - the filter to apply.

Returns:

the applications available matching the filtering criteria.

Since:

MHP1.0.

### getAppProxy(AppID)

```
public org.dvb.application.AppProxy getAppProxy(org.dvb.application.AppID key)
```

Returns the ApplicationProxy associated with the given ID. Returns null if no such application available.

Only one AppProxy object shall be returned in the case where there are several applications having the same (organisationId, applicationId) pair. In such a case, the same algorithm as would be used to autostart such applications shall be used to decide between the available choices by the implementation.

If an application has an application instance in the destroyed state then a proxy for that appplication instance shall not be retrieved. Instead, what shall be retrieved is a proxy for another application instance which shall be in the not loaded state unless that application instance has already been started.

Parameters:

`key` - an application ID.

Returns:

the AppProxy associated with the key parameter or null if the key is not an application ID, or not mapped to any application available.

Throws:

`java.lang.SecurityException` - shall not be thrown for AppIDs which are returned by getAppIDs(CurrentServiceFilter) or getAppIDs(RunningApplicationsFilter).

Since:

MHP1.0.

### getAppsDatabase()

```
public static org.dvb.application.AppsDatabase getAppsDatabase()
```

Returns the singleton AppsDatabase object. The AppsDatabase is either a singleton for each GEM application or a singleton for the GEM terminal.

Returns:

the singleton AppsDatabase object.

Since:

MHP1.0.

### removeListener(AppsDatabaseEventListener)

```
public void removeListener(org.dvb.application.AppsDatabaseEventListener listener)
```

Remove a listener on the database.

Parameters:

`listener` - the listener to be removed.

Since:

MHP1.0.

### size()

```
public int size()
```

Returns the number of applications currently available.

Returns:

the number of applications currently available.

Since:

.HP1.0

# org.dvb.application

# AppsDatabaseEvent

### Declaration

```
public class AppsDatabaseEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.application.AppsDatabaseEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

The `AppsDatabaseEvent` class indicates either an entry in the application database has changed, or so many changes have occurred. that the database should be considered totally new. An event with event_id NEW_DATABASE shall always be sent after switching to a new service. After such an event, the contents of the database (both the set of applications and their attributes) shall reflect the new database contents. All former contents of the database shall be discarded except for running externally authorised applications. It is platform dependant if and when a new database event is thrown while tuned to the same service except that a NEW_DATABASE event shall not be sent when only one application has changed within a service.

The APP_ADDED, APP_CHANGED and APP_DELETED events shall not be generated in response to the same database change as caused a NEW_DATABASE event to be generated.

### Since:

MHP1.0.

### Fields

#### APP_ADDED

```
public static final int APP_ADDED
```

The addition event id. The APP_ADDED event is generated whenever an entry is added to the AppsDatabase. It is NOT generated when the entry already in the AppsDatabase changes.

#### APP_CHANGED

```
public static final int APP_CHANGED
```

The changed event id. The APP_CHANGED event is generated whenever any of the information about an application changes. It is NOT generated when the entry is added to or removed from the AppsDatabase. In such cases, the APP_ADDED or APP_DELETED events will be generated instead.

#### APP_DELETED

```
public static final int APP_DELETED
```

The deletion event id. The APP_DELETED event is generated whenever an entry is removed from the AppsDatabase.

#### NEW_DATABASE

```
public static final int NEW_DATABASE
```

The new database event id.

### Constructors

#### AppsDatabaseEvent(int, AppID, Object)

```
public AppsDatabaseEvent(int id, org.dvb.application.AppID appid, java.lang.Object source)
```

Create a new AppsDatabaseEvent object for the entry in the database that changed, or for a new database.

Parameters:

`id` - the cause of the event.

`appid` - the AppId of the entry that changed.

`source` - the AppaDatabase object.

Since:

MHP1.0.

### Methods

#### getAppID()

```
public org.dvb.application.AppID getAppID()
```

gets the application ID object for the entry in the database that changed.

When the event type is NEW_DATABASE, AppID will be null.

Returns:

application ID representing the application.

Since:

MHP1.0.

#### getEventId()

```
public int getEventId()
```

gets the type of the event.

Returns:

an integer that matches one of the static fields describing events.

Since:

MHP1.0.

# org.dvb.application

# AppsDatabaseEventListener

### Declaration

```
public interface AppsDatabaseEventListener extends java.util.EventListener
```

All Superinterfaces:

```
java.util.EventListener
```

Description

The `AppsDatabaseListener` class allows an application to monitor the application database so that it can keep an up to date interface without polling the state. The application shall receive these events in a timely fashion after the AIT changes, however it is system dependant how often the AIT table is checked.

Since:

MHP1.0.

Methods

entryAdded(AppsDatabaseEvent)

```
public void entryAdded(org.dvb.application.AppsDatabaseEvent evt)
```

The AppsDataBase has had an application entry added.

Parameters:

`evt` - the AppsDatabaseEvent.

Since:

MHP1.0.

entryChanged(AppsDatabaseEvent)

```
public void entryChanged(org.dvb.application.AppsDatabaseEvent evt)
```

The AppsDataBase has had an application entry changed.

Parameters:

`evt` - the AppsDatabaseEvent.

Since:

MHP1.0.

entryRemoved(AppsDatabaseEvent)

```
public void entryRemoved(org.dvb.application.AppsDatabaseEvent evt)
```

The AppsDataBase has had an application entry removed.

Parameters:

`evt` - the AppsDatabaseEvent.

Since:

MHP1.0.

newDatabase(AppsDatabaseEvent)

```
public void newDatabase(org.dvb.application.AppsDatabaseEvent evt)
```

The AppsDataBase has radically changed.

Parameters:

`evt` - the AppsDatabaseEvent.

Since:

MHP1.0.

# org.dvb.application
# AppsDatabaseFilter

### Declaration

```
public abstract class AppsDatabaseFilter

java.lang.Object
  |
  +--org.dvb.application.AppsDatabaseFilter
```

### Direct Known Subclasses:

`CurrentServiceFilter, RunningApplicationsFilter`

### Description

Abstract class for the filters. Instances of concrete classes that extend `AppsDatabaseFilter` are passed to the AppsDatabase.getAppAttributes and AppsDatabase.getAppIDs methods to allow an applications to set a filter on the list of applications (respectively AppAttributes and AppIDs) that it wants to retrieve from the AppDatabase.

### Since:

MHP 1.0.

### Constructors

### AppsDatabaseFilter()

```
public AppsDatabaseFilter()
```

Construct an AppsDatabaseFilter object.

### Methods

### accept(AppID)

```
public abstract boolean accept(org.dvb.application.AppID appid)
```

Test if a specified appid should be included in the Enumeration.

### Parameters:

`appid` - the specified appid to test.

### Returns:

true if the application with identifier appid should be listed, false otherwise.

### Since:

MHP1.0.

# org.dvb.application
# AppStateChangeEvent

### Declaration

```
public class AppStateChangeEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.application.AppStateChangeEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

The `AppStateChangeEvent` class indicates a state transition of the application. These events are only generated for running applications or for non-running applications where an attempt to control the application fails. If the state transition was requested by an application through this API, the method `hasFailed` indicates whether the state change failed or not. Where a state change succeeds, `fromState` and `toState` shall indicate the original and destination state of the transition. If it failed, `fromState` shall return the state the application was in before the state transition was requested and the `toState` method shall return the state the application would have been in if the state transition had succeeded.

Attempting to start an application which is already in the active state shall fail and generate an `AppStateChange-Event` with `hasFailed` returning true and both fromstate and tostate being `STARTED`.

### Since:

MHP1.0.

### Constructors

### AppStateChangeEvent(AppID, int, int, Object, boolean)

```
public AppStateChangeEvent(org.dvb.application.AppID appid, int fromstate, int tostate,
java.lang.Object source, boolean hasFailed)
```

Create an AppStateChangeEvent object.

### Parameters:

`appid` - a registry entry representing the tracked application.

`fromstate` - the state the application was in before the state transition was requested, where the value of fromState is one of the state values defined in the AppProxy interface or in the interfaces inheriting from it.

`tostate` - state the application would be in if the state transition succeeds, where the value of toState is one of the state values defined in the AppProxy interface or in the interfaces inheriting from it.

`hasFailed` - an indication of whether the transition failed (true) or succeeded (false).

`source` - the `AppProxy` where the state transition happened.

### Methods

### getAppID()

```
public org.dvb.application.AppID getAppID()
```

The application the listener was tracking has made a state transition from `fromState` to `toState`.

Returns:

a registry entry representing the tracked application.

Since:

MHP1.0.

### getFromState()

```
public int getFromState()
```

The application the listener is tracking was in `fromState`, where the value of fromState is one of the state values defined in the AppProxy interface or in the interfaces inheriting from it.

Returns:

the old state.

Since:

MHP1.0.

### getToState()

```
public int getToState()
```

If the `hasFailed` method returns false, then the application the listener is tracking is now in `toState`. If the `hasFailed` method returns true, then the `toState` is the state where the state transition was attempted to but the transition failed. The value of `toState` is one of the state values defined in the `AppProxy` interface or in the interfaces inheriting from it.

Returns:

the intended or actual new state.

Since:

MHP1.0.

### hasFailed()

```
public boolean hasFailed()
```

This method determines whether an attempt to change the state of an application has failed.

Returns:

true if the attempt to change the state of the application failed, false otherwise.

Since:

MHP1.0.

# org.dvb.application

# AppStateChangeEventListener

Declaration

```
public interface AppStateChangeEventListener extends java.util.EventListener
```

All Superinterfaces:

```
java.util.EventListener
```

### Description

The `AppStateChangeEventListener` class allows a launcher application to keep track of applications it launches or other applications running as part of the same service.

### Since:

MHP1.0.

### Methods

#### stateChange(AppStateChangeEvent)

```
public void stateChange(org.dvb.application.AppStateChangeEvent evt)
```

The application the listener was tracking has made a state transition from `fromState` to `toState` and this method will be given the state event.

### Parameters:

`evt` - the AppStateChangeEvent.

### Since:

MHP1.0.

# org.dvb.application

# CurrentServiceFilter

### Declaration

```
public class CurrentServiceFilter extends AppsDatabaseFilter
```

```
java.lang.Object
  |
  +--org.dvb.application.AppsDatabaseFilter
        |
        +--org.dvb.application.CurrentServiceFilter
```

### Description

Instances of `CurrentServiceFilter` are used to set a filter on the list of applications that are retrieved from the Apps-Database (See methods getAppsAttributes and getAppsIDs).

A `CurrentServiceFilter` is used to indicate that only applications that signalled as part of the current service shall be returned by the getAppsAttributes and getAppIDs methods of AppsDatabase. Externally authorized applications in the AIT are not considered to be signalled as part of the current service for this filter. Subclasses of CurrentServiceFilter can override the accept method so as to implement their own filter criteria on the AppIDs values.

### Since:

MHP 1.0.

### Constructors

#### CurrentServiceFilter()

```
public CurrentServiceFilter()
```

public Constructor of the CurrentServiceFilter.

### Methods

#### accept(AppID)

```
public boolean accept(org.dvb.application.AppID appid)
```

Test if a specified appid should be included in the Enumeration.

Overrides:

`accept` in `class AppsDatabaseFilter`

Parameters:

`appid` - the specified appid to test.

Returns:

true if the application with identifier appid should be listed, false otherwise.

Since:

MHP1.0.

# org.dvb.application

# DVBJProxy

### Declaration

```
public interface DVBJProxy extends AppProxy
```

All Superinterfaces:

`AppProxy`

### Description

A `DVBJProxy` Object is a proxy to a DVBJ application.

### Fields

#### LOADED

```
public static final int LOADED
```

The application is in the loaded state.

### Methods

#### init()

```
public void init()
```

Requests the application manager calls the `initXlet` method on the application.

This method is asynchronous and its completion will be notified by an AppStateChangeEvent. In case of failure, the hasFailed method of the `AppStateChangeEvent` will return true. Calls to this method shall only succeed if the application is in the NOT_LOADED or LOADED states. If the application is in the NOT_LOADED state, the application will move through the LOADED state into the PAUSED state before calls to this method complete.

In all cases, an AppStateChangeEvent will be sent, whether the call was successful or not.

Throws:

`java.lang.SecurityException` - if the application is not entitled to load this application. Being able to init an application requires to be entitled to start it.

Since:

MHP1.0.

### init(String[])

```
public void init(java.lang.String[] args)
```

Requests the application manager calls the `initXlet` method on the application passing to that application the specified parameters.

This method is asynchronous and its completion will be notified by an AppStateChangeEvent. In case of failure, the hasFailed method of the `AppStateChangeEvent` will return true. Calls to this method shall only succeed if the application is in the NOT_LOADED or LOADED states. If the application is in the NOT_LOADED state, the application will move through the LOADED state into the PAUSED state before calls to this method complete.

In all cases, an AppStateChangeEvent will be sent, whether the call was successful or not.

Parameters:

`args` - the parameters to be passed into the application being started.

Throws:

`java.lang.SecurityException` - if the application is not entitled to load this application. Being able to init an application requires to be entitled to start it.

Since:

MHP1.1.2.

### load()

```
public void load()
```

Provides a hint to preload at least the initial class of the application into local storage, resources permitting. This does not require loading of classes into the virtual machine or creation of a new logical virtual machine which are implications of the `init` method.

This method is asynchronous and its completion will be notified by an `AppStateChangeEvent`. In case of failure, the `hasFailed` method of the `AppStateChangeEvent` will return true. Calls to this method shall only succeed if the application is in the NOT_LOADED state. In all cases, an AppStateChangeEvent will be sent, whether the call was successful or not.

Throws:

`java.lang.SecurityException` - if the application is not entitled to load this application. Being able to load an application requires to be entitled to start it.

Since:

MHP1.0.

# org.dvb.application IllegalProfileParameterException

### Declaration

```
public class IllegalProfileParameterException extends java.lang.Exception

java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.application.IllegalProfileParameterException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

The `IllegalProfileParameter` exception is thrown if the application attempts to ask for a version number for a profile not specified for the application.

### Since:

MHP1.0.

### Constructors

#### IllegalProfileParameterException()

```
public IllegalProfileParameterException()
```

Construct a IllegalProfileParameterException with no detail message

#### IllegalProfileParameterException(String)

```
public IllegalProfileParameterException(java.lang.String s)
```

Construct a IllegalProfileParameterException with a detail message

### Parameters:

`s` - detail message.

# org.dvb.application LanguageNotAvailableException

### Declaration

```
public class LanguageNotAvailableException extends java.lang.Exception

java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.application.LanguageNotAvailableException
```

All Implemented Interfaces:

`java.io.Serializable`

Description

The `LanguageNotAvailableException` exception is thrown if the application asks for the name of an application in a language not signalled in the AIT.

Since:

MHP1.0.

Constructors

LanguageNotAvailableException()

```
public LanguageNotAvailableException()
```

Construct a LanguageNotAvailableException with no detail message

LanguageNotAvailableException(String)

```
public LanguageNotAvailableException(java.lang.String s)
```

Construct a LanguageNotAvailableException with a detail message

Parameters:

`s` - detail message.

# org.dvb.application RunningApplicationsFilter

Declaration

```
public class RunningApplicationsFilter extends AppsDatabaseFilter
```

```
java.lang.Object
  |
  +--org.dvb.application.AppsDatabaseFilter
        |
        +--org.dvb.application.RunningApplicationsFilter
```

Description

Instances of `RunningApplicationsFilter` are used to set a filter on the list of applications that are retrieved from the AppsDatabase (See methods getAppsAttributes and getAppsIDs).

A `RunningApplicationsFilter` is used to indicate that only applications that are running as part of the current service shall be returned by the getAppsAttributes and getAppIDs methods of AppsDatabase. Externally authorized applications in the AIT shall be returned if they are currently running in the same service context as the caller. Running applications whose visibility is '00' shall not be returned. Subclasses of RunningApplicationsFilter can override the accept method so as to implement their own filter criteria on the AppIDs values.

Since:

MHP 1.0.

Constructors

RunningApplicationsFilter()

```
public RunningApplicationsFilter()
```

public Constructor of the RunningApplicationsFilter

### Methods

#### accept(AppID)

```
public boolean accept(org.dvb.application.AppID appid)
```

Test if a specified appid should be included in the Enumeration.

**Overrides:**

`accept` in class `AppsDatabaseFilter`

**Parameters:**

`appid` - the specified appid to test.

**Returns:**

true if the application with identifier appid should be listed, false otherwise.

**Since:**

MHP1.0.

# Annex T (normative): Permissions

NOTE 1: See clause 11.10, "Java permissions".

NOTE 2: As a consequence of clause 11.8.3, "Additional permissions classes," the packages `org.dvb.net.ca` and the class `org.dvb.net.tuning.DvbNetworkInterfaceSIUtil` are not required under certain circumstances.

# Package

# org.dvb.net.ca

### Description

Provides extensions to the conditional access API from DAVIC.

| Class Summary | |
|---|---|
| **Classes** | |
| `CAPermission` | This class is for CA permissions. |

# org.dvb.net.ca

# CAPermission

### Declaration

```
public class CAPermission extends java.security.BasicPermission

java.lang.Object
  |
  +--java.security.Permission
        |
        +--java.security.BasicPermission
              |
              +--org.dvb.net.ca.CAPermission
```

### All Implemented Interfaces:

```
java.security.Guard, java.io.Serializable
```

### Description

This class is for CA permissions. A CAPermission contains a name, but no actions list.

A CAPermission contains a range of CA system ids and a specific permission for that range of CA system ids. Instead of a range of CA system ids, the CAPermission can also refer to a single CA system id.

The name has the following syntax:

CASystemIdRange ":" Permission

where CASystemIdRange = CASystemId [ "-" CASystemId ] | "*"
and Permission = "MMI" | "buy" | "entitlementInfo" | "messagePassing" | "*"

EXAMPLES:

- 0x1200-0x120A:buy (The permission to buy entitlement for all the CA systems with ids between 0x1200 and 0x120A inclusive).
- 0x1201:entitlementInfo (The permission to get entitlement information for the CA system with id 0x1201).
- 0x120d:* (This wildcard expresses all the permissions for the CA system with id 0x120d).

NOTE: The CASystemId is expressed as a hexadecimal value.

The permission "MMI" corresponds with the SecurityException on CAModuleManager.addMMIListener(). The permission "buy" corresponds with the SecurityException on CAModule.buyEntitlement(). The permission "entitlementInfo" corresponds with the SecurityException on CAModule.queryEntitlement() and CAModule.list-Entitlements(). The permission "messagePassing" corresponds with CAModule.openMessageSession(Message-Listener).

### Constructors

#### CAPermission(String)

```
public CAPermission(java.lang.String name)
```

Creates a new CAPermission with the specified name. The name is the symbolic name of the CAPermission.

#### Parameters:

`name` - the name of the CAPermission.

#### CAPermission(String, String)

```
public CAPermission(java.lang.String name, java.lang.String actions)
```

Creates a new CAPermission object with the specified name. The name is the symbolic name of the CAPermission, and the actions String is unused and should be null. This constructor exists for use by the Policy object to instantiate new Permission objects.

#### Parameters:

`name` - the name of the CAPermission.

`actions` - should be null.

### Methods

#### implies(Permission)

```
public boolean implies(java.security.Permission p)
```

Checks if the specified permission is "implied" by this object.

#### Overrides:

`implies` in class `BasicPermission`

#### Parameters:

`p` - the permission to check against.

#### Returns:

true if the passed permission is equal to or implied by this permission, false otherwise.

# Package org.dvb.net.tuning

### Description

Provides extensions to the tuning API from DAVIC.

| Class Summary | |
|---|---|
| **Classes** | |
| DvbNetworkInterfaceSIUtil | Each SI database is associated with a network interface and vice versa. |
| TunerPermission | This class is for tuner permissions. |

# org.dvb.net.tuning DvbNetworkInterfaceSIUtil

### Declaration

```
public class DvbNetworkInterfaceSIUtil

java.lang.Object
  |
  +--org.dvb.net.tuning.DvbNetworkInterfaceSIUtil
```

### Description

Each SI database is associated with a network interface and vice versa. This class allows the application to query this association.

Since:

MHP 1.0.1.

### Methods

#### getNetworkInterface(SIDatabase)

```
public static org.davic.net.tuning.NetworkInterface getNetworkInterface(org.dvb.si.SIDatabase sd)
```

Gets the network interface for a particular SI database.

Parameters:

`sd` - the SI database for which the associated network interface will be returned.

Returns:

the associated network interface.

#### getSIDatabase(NetworkInterface)

```
public static org.dvb.si.SIDatabase getSIDatabase(org.davic.net.tuning.NetworkInterface ni)
```

Gets the SI database for a particular network interface.

Parameters:

`ni` - the network interface for which the associated SI database will be returned.

Returns:

the associated SI database.

# org.dvb.net.tuning

# TunerPermission

## Declaration

```
public class TunerPermission extends java.security.BasicPermission

java.lang.Object
  |
  +--java.security.Permission
        |
        +--java.security.BasicPermission
              |
              +--org.dvb.net.tuning.TunerPermission
```

## All Implemented Interfaces:

```
java.security.Guard, java.io.Serializable
```

## Description

This class is for tuner permissions. The name and actions list of a TunerPermission contains no name and no actions list. The return value of the inherited getName() method is implementation dependent. If an application has the tuner permission, then it shall not receive a `SecurityException` from those methods in that API defined to throw one. Without such a permission, it shall receive such an exception.

## Constructors

### TunerPermission(String)

```
public TunerPermission(java.lang.String name)
```

Creates a new TunerPermission. The name string is currently unused and should be empty.

Parameters:

`name` - the name of the TunerPermission.

### TunerPermission(String, String)

```
public TunerPermission(java.lang.String name, java.lang.String actions)
```

Creates a new TunerPermission. The name string is currently unused and should be empty. The actions string is currently unused and should be null. This constructor exists for use by the Policy object to instantiate new Permission objects.

Parameters:

`name` - the name of the TunerPermission.

`actions` - the actions list.

## Methods

### implies(Permission)

```
public boolean implies(java.security.Permission p)
```

Checks if the specified permission is "implied" by this object.

Since name and actions are not used, the only check needed is whether p is also a TunerPermission.

Overrides:

`implies` in class `BasicPermission`

Parameters:

`p` - the permission to check against.

Returns:

true if the passed permission is equal to or implied by this permission, false otherwise.

# Annex U (normative): Extended graphics APIs

# Package

# org.dvb.ui

### Description

Provides extended graphics functionality.

| Class Summary | |
|---|---|
| **Interfaces** | |
| TestOpacity | Interface implemented by Components or Containers in order to allow the platform to query whether their paint method is fully opaque. |
| TextOverflowListener | The TextOverflowListener is an interface that an application may implement and register in the DVBTextLayoutManager. |
| **Classes** | |
| BufferedAnimation | A BufferedAnimation is an AWT component that maintains a queue of one or more image buffers. |
| DVBAlphaComposite | This DVBAlphaComposite class implements the basic alpha compositing rules for combining source and destination pixels to achieve blending and transparency effects with graphics, images and video. |
| DVBBufferedImage | The DVBBufferedImage subclass describes an java.awt.Image with an accessible buffer of image data. |
| DVBColor | A Color class which adds the notion of alpha. |
| DVBGraphics | The DVBGraphics class is a adapter class to support alpha compositing in a GEM device. |
| DVBTextLayoutManager | The DVBTextLayoutManager provides a text rendering layout mechanism for the org.havi.ui.HStaticText org.havi.ui.HText and org.havi.ui.HTextButton classes. |
| FontFactory | Provides a mechanism for applications to instantiate fonts that are not built into the system. |
| **Exceptions** | |
| DVBRasterFormatException | This exception is thrown for some invalid operations on instances of DVBBufferedImage. |
| FontFormatException | Thrown when attempt is made to read a file describing a font when the contents of that file are not valid. |
| FontNotAvailableException | Thrown when attempt is made to instantiate a font that cannot be located. |
| UnsupportedDrawingOperationException | The UnsupportedDrawingOperationException class represents an exception that is thrown if an drawing operation is not supported on this platform. |

# org.dvb.ui

# BufferedAnimation

### Declaration

```
public class BufferedAnimation extends java.awt.Component

java.lang.Object
  |
  +--java.awt.Component
        |
```

```
    +--org.dvb.ui.BufferedAnimation
```

### All Implemented Interfaces:

`java.awt.image.ImageObserver, java.io.Serializable`

### Description

A `BufferedAnimation` is an AWT component that maintains a queue of one or more image buffers. This permits efficient flicker-free animation by allowing a caller to draw to an off-screen buffer, which the system then copies to the framebuffer in coordination with the video output subsystem. This class also allows an application to request a series of buffers, so that it can get a small number of frames ahead in an animation. This allows an application to be robust in the presence of short delays, e.g. from garbage collection. A relatively small number of buffers is recommended, perhaps three or four. A BufferedAnimation with one buffer provides little or no protection from pauses, but does provide double-buffered animation.

This class can be used for frame-synchronous animation. When animation is in progress, it maintains a count of the frame number in the underlying video output device. This frame number increases monotonically by one for each video frame output. It is not influenced by trick play of any video that might be playing on the same screen. However, the framerate of the `BufferedAnimation` may be determined by the framerate of such video; see `setFramerate(float)` and `getFramerate()` for details.

The implementation shall prevent tearing artifacts whenever possible. The maximum size and animation rate that can be achieved without tearing artifacts may be specified by the system model of a specification that includes this class. If it is necessary to avoid a tearing artifact, an implementation shall delay the copying of an internal buffer to the frame buffer by up to one frame.

The size of this component is set using the normal AWT mechanisms. When the method `setBuffers(Dimension, int, boolean, boolean)` is called, initialization is performed. At this time, the size of the graphics buffers is set.

When one of this component's buffers is copied to the frame buffer, it is done without regard to any AWT components which may overlap with this component. This is like the behavior when an application draws directly to the screen using a graphics object obtained with `java.awt.Component.getGraphics()`.

When the system copies a buffer to the frame buffer for a given frame *f*, it shall select the valid buffer associated with the highest-numbered frame *fb* such that *fb* <= *f*. A buffer is valid if `startFrame(int)` has been called for that buffer, `finishFrame(int)` has been called, and the given buffer has not been re-used for a subsequent frame as a result of another call to `startFrame(int)`.

The animation task proceeds at a high relative CPU priority, and can be considered to execute at a priority greater than Java's `Thread.NORM_PRIORITY`. However, when no new image buffer is ready, the system task must always block until one is. Drawing into the buffer is done within a Java thread, which is subject to the normal scheduling guarantees. In this way, a CPU-bound caller can avoid starving more important activities, such as responding to remote control input. CPU-bound applications may wish to invoke `Thread.yield()` after each frame, however, particularly if the application's animation thread is at the same priority level as other application threads.

A component that is not visible and is in the started state will run, but it will not display any buffers to the screen. It will block in the call to `startFrame()` until the component becomes visible, or until it is too late to draw the requested frame, whichever comes first. Once it is too late, it will, of course, return -1, thus ensuring that the caller doesn't waste time drawing to an internal graphics buffer that wouldnt' be displayed.

For the behavior when this component is destroyed, see `removeNotify()`.

Sample usage:

```
BufferedAnimation anim = new BufferedAnimation();
 ... put anim in a component hierarchy
Dimension d = new Dimension(...);
int numBuffers = 4;
for (;;) {
    try {
        anim.setBuffers(d, 4, false, true);
        break;
    } catch (OutOfMemoryError err) {
        ... try smaller buffers, or fewer of them
    }
 }
```

```
 ... set framerate, if needed
 ... Make anim visible
Graphics2D[] bufs = anim.getBuffersGraphics();
anim.startAnimation();
//  Animate 1000 frames...
try {
    for (int f = 0; f < 1000; f++) {
        int n;
        try {
             n = anim.startFrame(f); // blocks until a buffer is free
        } catch (InterruptedException ex) {
             // someone else called stopAnimation. or removeNotify() was
             // called.  In any case, we're being asked to stop the
             // animation immediately.
             break;
        }
        if (n > -1) {
             try {
                 myDrawFrame(f, bufs[n]);
             } finally {
                 anim.finishFrame(f);
             }
         }
    }
 } finally {
     anim.stopAnimation(false);
 }
```

This class does not specify a return value for `Component.isDoubleBuffered()`. That method reports on a different kind of buffering, related to the `repaint()` call. `BufferedAnimation` objects might or might not be double-buffered, in the repaint-related sense meant by `Component.isDoubleBuffered()`.

NOTE: A future version of this API could potentially allow simultaneous drawing to two frames, by relaxing the synchronization condition on startFrame(int). However, it is unclear if this would yield benefits e.g. on a multi-core system, given memory bandwith limitations and the already existing ability to have parallel threads drawing into one frame or doing other computations.

Since:

MHP 1.1.3.

Fields

FRAME_23_98

```
public static float FRAME_23_98
```

Constant representing a common video framerate, approximately 23.98 frames per second, and equal to `24000f/1001f`.

See Also:

`getFramerate(), setFramerate(float)`

FRAME_24

```
public static float FRAME_24
```

Constant representing a common video framerate, equal to `24f`.

See Also:

`getFramerate(), setFramerate(float)`

FRAME_25

```
public static float FRAME_25
```

Constant representing a common video framerate, equal to `25f`.

See Also:

`getFramerate(), setFramerate(float)`

### FRAME_29_97

```
public static float FRAME_29_97
```

Constant representing a common video framerate, approximately 29.97 frames per second, and equal to `30000f/1001f`.

See Also:

`getFramerate(), setFramerate(float)`

### FRAME_50

```
public static float FRAME_50
```

Constant representing a common video framerate, equal to `50f`.

See Also:

`getFramerate(), setFramerate(float)`

### FRAME_59_94

```
public static float FRAME_59_94
```

Constant representing a common video framerate, approximately 59.94 frames per second, and equal to `60000f/1001f`.

See Also:

`getFramerate(), setFramerate(float)`

## Constructors

### BufferedAnimation()

```
public BufferedAnimation()
```

Create a new `BufferedAnimation` component. The BufferedAnimation functionality may be optional. Applications written to device specifications that do not make this functionality mandatory should be prepared to catch UnsupportedOperationException when invoking this constructor.

Throws:

`java.lang.UnsupportedOperationException` - If this feature is not supported on the device.

## Methods

### addNotify()

```
public void addNotify()
```

Makes this `Component` displayable by connecting it to a native screen resource. This method is called internally by the toolkit and should not be called directly by programs.

Overrides:

`addNotify` in class `Component`

### finishFrame(int)

```
public void finishFrame(int frameNumber)
```

Notify the system that the frame currently being drawn is finished. Drawing the current frame is initiated with `startFrame(int)`. Once `finishFrame(int)` is called, the system can copy that frame to the framebuffer, and the caller can move on to preparing the next frame.

Parameters:

`frameNumber` - The frame number that is finished. This must match the value passed into startFrame(int).

Throws:

`java.lang.IllegalStateException` - if a startFrame call has not returned successfully for the given frame number (with a return value other than -1), if `finishFrame(int)` has already been called for this frame number since the startFrame call, or if `stopAnimation(boolean)` has been called since the corresponding startFrame call.

See Also:

`startFrame(int)`

### getBuffersGraphics()

```
public java.awt.Graphics2D[] getBuffersGraphics()
```

Get the graphics objects for drawing into this component's internal image buffers. The size and number of buffers is determined by the `setBuffers` method.

After the first call, subsequent invocations of this method shall return the identical value (i.e. multiple calls will return values that are == to each other).

Other than the `setBuffers` mechanism, the size of the internal buffers will never change. If the component is resized, the system might scale the resulting animation, but this behavior is not guaranteed by the specification of this class. In all cases, the upper-left hand corner of the image buffer will be displayed in the upper-left hand corner of the component.

The initial contents of the graphics buffers is undefined. Callers may wish to initialize the buffers to a known state, such as fully transparent, before starting an animation. Once an animation is started, drawing into a buffer outside of a `startFrame/finishFrame` pair may produce unpredictable results on the screen.

Returns:

An array of graphics objects that can be used to draw into the internal image buffers.

Throws:

`java.lang.IllegalStateException` - if the `setBuffers()` hasn't been successfully called.

See Also:

`setBuffers(Dimension, int, boolean, boolean)`

### getBuffersSize()

```
public java.awt.Dimension getBuffersSize()
```

Get the size of the internal image buffers. If setBuffers has not yet been successfully called, then null is returned.

See Also:

`getBuffersGraphics(), setBuffers(Dimension, int, boolean, boolean)`

### getFramerate()

```
public float getFramerate()
```

Get the actual framerate of the screen associated with this component. This class defines a number of common framerates as constants whose name begin with "`FRAME_`".

### getMediaTime(Clock, int)

```
public long getMediaTime(javax.media.Clock c, int frameNumber)
```

Get the predicted media time of the given frameNumber for this animation. The predicted media time is calculated from the media time of the frame being presented on the screen, and extrapolating assuming a clock rate of 1.0.

The return value can be converted into a javax.media.Time value by calling the `javax.media.Time(long)` constructor.

This method is useful for keeping an animation aligned with a video source, even if "trick play" operations cause the media position to change. Note that because the computer generated animation might be a small number of frames "ahead" of the video due to the buffering this class provides, there might be a perceptable "lag" during trick play itself, but once the video returns to normal play mode, the animation would once again be frame-synchronized.

Parameters:

`c` - The clock to calculate media time relative to.

`frameNumber` - the desired frame number of this animation.

Returns:

The media time, in nanoseconds.

Throws:

`java.lang.IllegalStateException` - if this animation is not in the started state.

`javax.media.IncompatibleTimeBaseException` - If this Clock is incompatible with this animation. This will never be thrown if the Clock is associated with video being displayed on the same screen as this `BufferedAnimation` component.

`java.lang.IllegalStateException` - If this clock is not in the started state.

### isStarted()

```
public boolean isStarted()
```

Return true if this animation is started. A `BufferedAnimation` can either be in the started or stopped state. A stopped `BufferedAnimation` will only draw to the framebuffer if it is in the process of flushing animation buffers due to a call to `stopAnimation(false)`

Returns:

true if this `BufferedAnimation` is started, false otherwise.

See Also:

`stopAnimation(boolean)`

### paint(Graphics)

```
public void paint(java.awt.Graphics g)
```

If this component has an active animation, then this method paints either the last valid image buffer or the next valid image buffer to the given graphics object. If there is no available valid image buffer and a `startFrame/finishFrame` sequence is in progress, this method will block until `finishFrame` is called, thus generating a valid image buffer. If no animation is in progress or no valid frames have yet been generated, then this method does nothing.

Note that in normal operation, this method should be called by the platform very infrequently, if at all. It might be called, for example, due to an "expose event," or due to a call to `Component.print(Graphics)`. Application authors should not request a call to `paint` via the repaint mechanism to animate this component, because this class uses a different model for animation.

Overrides:

`paint` in class `Component`

### removeNotify()

```
public void removeNotify()
```

Make this component undisplayable by destroying any native resources, and freeing its image buffers. `stopAnimation(true)` shall be called by the implementation of this method.

Overrides:

removeNotify in class Component

### setBuffers(Dimension, int, boolean, boolean)

```
public void setBuffers(java.awt.Dimension bufSize, int numBuffers, boolean forceSize,
boolean forceAcceleration)
```

Set the size and number of the internal image buffers. If the system is capable of scaling a BufferedAnimation's buffers to the component's size in real-time, then the internal buffers will be of the requested size, and scaling will occur. If the system is not, then the results will depend on the value of forceSize.

On a system that cannot perform a requested scaling, if forceSize is true, then the buffers' sizes will be set to the requested size regardless. The displayed result will be clipped or will have areas that are unpainted, as needed. In all cases, the upper-left hand corner of the buffers will be painted at to the upper-left hand corner of the component.

On a system that cannot perform scaling, if forceSize is false, the buffers' size will be set to the current size of the component. That is, the requested buffer dimension will be ignored.

The system model of specifications that include this class may specify a set of supported scalings.

Note that the framebuffer itself might be scaled for display on the output device. For example, a specification including this class might include half-resolution mode, e.g. for half-resolution computer graphics over 1080i video.

Graphics Acceleration

Some systems have special faster video memory that gives accelerated graphics performance. The system model of a specification adopting this API may define a minimum amount of such memory. Other hardware architectures, such as "unified memory architecture" platforms, don't have special accelerated memory. On these platforms, all video memory is considered "accelerated", that is, the forceAccelerated parameter has no effect, and does not cause automatic failure.

On platforms with special accelerated video memory, the caller may indicate that all of the graphics buffers must be allocated from this accelerated memory. It does this by setting the forceAcceleration parameter true. This may make an OutOfMemoryError more likely. If forceAcceleration is not true, then the implementation will make a "best-effort" attempt to put the buffers in accelerated memory, but will fall back to normal heap memory, if required.

This method may be called more than once. If it exits with an exception, the state of this object will not be changed. If it returns normally, the new values will override anything set previously.

Parameters:

bufSize - The requested buffer size.

numBuffers - The number of image buffers to allocate.

forceSize - Force sizing the buffers to the requested size, even if this means clipping or having unpainted areas.

forceAcceleration - Force allocation of all buffers in accelerated memory.

Throws:

java.lang.IllegalArgumentException - if d.width or d.height is less than one, or numBuffers is less than one.

java.lang.IllegalStateException - if getBuffersGraphics has been called for this component.

java.lang.IllegalStateException - If this component isn't displayable.

OutOfMemoryException - If there isn't enough memory to allocate the needed buffers.

See Also:

java.awt.Component.isDisplayable()

### setFramerate(float)

```
public void setFramerate(float rate)
```

Attempt to set the framerate of the screen associated with this component. Other factors, such as video being output to the same device or device limitations, might determine the framerate, thus causing this method to have no effect. The frameright might subsequently be changed, e.g. by video being presented on the screen or by other APIs. Unless other behavior is mandated by the system model of a specification incorporating this class, it is an allowable implementation option for this method to never change the framerate.

The system model of specifications including this class might determine under what conditions the framerate can be set, and what framerates are guaranteed to be supported. This class defines a number of common framerates as constants whose name begin with "`FRAME_`".

Note that an application that wishes to animate at a lower framerate than that of the hardware may do so, by simply skipping frames. This is discussed in the `startFrame(int)` method.

Throws:

`java.lang.IllegalStateException` - If this component is not displayable.

See Also:

`java.awt.Component.isDisplayable()`, `startFrame(int)`

### startAnimation()

```
public void startAnimation()
```

Start this animation immediately, and reset the frame number to zero. Frame zero will be the first frame that can be displayed; typically it will be one or two frames after the frame visible on the screen at the time this method is called.

Applications should only call this method on a stopped animation. However, if this method is called when an animation is already started, it is re-started; the effect is equivalent to calling `stopAnimation(true)` followed by `start-Animation()`.

See Also:

`isStarted()`

### startAnimationAt(Clock, Time)

```
public void startAnimationAt(javax.media.Clock c, javax.media.Time t)
throws IncompatibleTimeBaseException
```

Start this animation keyed to the clock at the given media time. If the clock's media time is already greater than the given time, this is equivalent to `startAnimation()`. Otherwise, once the clock's media time is greater than or equal to the given the given value, the animation will be started. Callers should not assume that frame zero of the animation will coincide with the desired time in all cases; for example, the clock's media time might advance in a discontinuous manner. Callers should always consult `getMediaTime(...)`.

This method can be used to initiate frame-accurate animation that is synchronized to video that is being presented on the same screen. The animation enters the started state, and drawing to graphics buffers can begin. The system will start copying these buffers to the framebuffer automatically, when the clock reaches the given time.

Subsequent calls to this method override any previous calls. If this method is called when an animation is already started, it is re-started; the effect is equivlanet to calling `stopAnimation(true)` followed by `startAnimationAt(...)`.

Parameters:

`c` - A JMF Clock that is associated with some media.

`t` - A media time of that clock when the animation should start.

Throws:

`javax.media.IncompatibleTimeBaseException` - If this Clock is incompatible with this animation. This will never be thrown if the Clock is associated with video being displayed on the same screen as this animation.

`java.lang.IllegalStateException` - If this clock is not in the started state.

See Also:

`isStarted()`, `getMediaTime(Clock, int)`

### startFrame(int)

```
public int startFrame(int frameNumber)
throws InterruptedException
```

Start drawing the given frame. The return value gives the index into the array obtained from `getBuffersGraphics()` for drawing of this frame, or -1 if the animation has fallen behind, and a later frame should now be drawn. After calling this method, if a value other than -1 is returned, the caller may draw to the indicated graphics buffer. When it is finished, it shall call `finishFrame()`.

If a buffer is not available for the given frame, this method will block until one is ready.

The caller can always skip frames. For example, a caller wishing to animate at half of the component's framerate could request frames 0, 2, 4, 6, 8, 10, etc. In this example, if there are four buffers and animation does not fall behind, the caller would be instructed to draw into buffer 0, 1, 2, 3, 0, 1, etc. A caller that wishes to start animating at a frame greater than 0 may do so by simply starting with a number greater than zero; when the lower-numbered frames are being presented, the component will simply do no drawing.

The content of the framebuffers is not modified by the system. Thus, a caller that is drawing into buffer number *n* could function correctly if it only drew to pixels that have changed since it last drew into buffer number *n*.

Parameters:

`frameNumber` - The frame number to draw. The first frame is frame 0.

Returns:

An index into the array of graphics objects for drawing the given frame, or -1 if animation has fallen behind, and a later frame should now be drawn.

Throws:

`java.lang.IllegalArgumentException` - if frameNumber is less than or equal to a number previously supplied to this animation, or is less than zero.

`java.lang.IllegalStateException` - if `startFrame()` has already been successfully called without a corresponding `finishFrame()`.

`java.lang.InterruptedException` - If this animation is in the stopped state, either when this method is called or due to a state transition while it is blocked waiting for a graphics buffer.

See Also:

`getBuffersGraphics()`, `isStarted()`

### stopAnimation(boolean)

```
public void stopAnimation(boolean immediate)
```

Stops this animation. If it is already in the stopped state, this method has no effect. If it is in the started state, it is set to the stopped state. Once this component is in the stopped state, it will not draw into any pixels to the screen, or as a result of a call to the `paint()` method.

The caller may request that queued fames of animation be output to the screen. This will be done if the animation is in the started state, and immediate is set false.

If a successful call to `startFrame(int)` has not yet been matched with a call to `finishFrame(int)`, this object is set to the stopped state, which will cause `finishFrame(int)` ti fail. See that method for details.

After this method returns, `isStarted()` will return false. If this animation is not in the started state, calling this method will have no effect.

Parameters:

`immediate` - If the component should immediately stop copying buffers to the screen, instead of letting any queued animation frames be output.

See Also:

`isStarted()`, `finishFrame(int)`

# org.dvb.ui

# DVBAlphaComposite

## Declaration

```
public final class DVBAlphaComposite

java.lang.Object
  |
  +--org.dvb.ui.DVBAlphaComposite
```

## Description

This `DVBAlphaComposite` class implements the basic alpha compositing rules for combining source and destination pixels to achieve blending and transparency effects with graphics, images and video. The rules implemented by this class are a subset of the Porter-Duff rules described in T. Porter and T. Duff, "Compositing Digital Images", SIGGRAPH 84, 253-259.

If any input does not have an alpha channel, an alpha value of 1,0, which is completely opaque, is assumed for all pixels. A constant alpha value can also be specified to be multiplied with the alpha value of the source pixels.

The following abbreviations are used in the description of the rules:

- Cs = one of the color components of the source pixel without alpha.
- cs = color component of a source pixel pre-multiplied with alpha (cs = As*Ar*Cs).
- Cd = one of the color components of the destination pixel without alpha.
- cd = color component of a destination pixel pre-multiplied with alpha
- Cn = the new constructed color without alpha.
- cn = the new constructed color pre-multiplied with alpha.
- As = alpha component of the source pixel.
- Ad = alpha component of the destination pixel.
- An = the new alpha after compositing.
- Ar = alpha, specified by getInstance(int Rule, float Ar). Unless otherwise specified Ar = 1,0f.
- Fs = fraction of the source pixel that contributes to the output.
- Fd = fraction of the input destination pixel that contributes to the output.

The color and alpha components produced by the compositing operation are calculated as follows:

```
cn = (As*Ar)*Cs*Fs + Ad*Cd*Fd
An = (As*Ar)*Fs + Ad*Fd
 Cn = cn/An
```

where Fs and Fd are specified by each rule.

The alpha resulting from the compositing operation is stored in the destination if the destination has an alpha channel. Otherwise, the resulting color is divided by the resulting alpha before being stored in the destination and the alpha is discarded. If the alpha value is 0,0, the color values are set to 0,0.

See Also:

`java.awt.AlphaComposite`

Fields

Clear

```
public static final org.dvb.ui.DVBAlphaComposite Clear
```

`DVBAlphaComposite` object that implements the opaque CLEAR rule with an alpha (Ar) of 1,0f.

See Also:

`CLEAR`

CLEAR

```
public static final int CLEAR
```

Porter-Duff Clear rule. Both the color and the alpha of the destination are cleared. Neither the source nor the destination is used as input.

Fs = 0 and Fd = 0, thus:

```
cn = 0
An = 0
 Cn = 0
```


Note that this operation is a fast drawing operation This operation is the same as using a source with alpha= 0 and the SRC rule.

DST_IN

```
public static final int DST_IN
```

Porter-Duff Destination In Source rule. The part of the destination lying inside of the source replaces the destination.

Fs = 0 and Fd = (As*Ar), thus:

```
cn = Ad*Cd*(As*Ar)
An = Ad*(As*Ar)
 Cn = Cd
```


Note that this operation is faster than e.g. SRC_OVER but slower than SRC

### DST_OUT

```
public static final int DST_OUT
```

Porter-Duff Destination Held Out By Source rule. The part of the destination lying outside of the source replaces the destination.

Fs = 0 and Fd = (1-(As*Ar)), thus:

```
 cn = Ad*Cd*(1-(As*Ar))
 An = Ad*(1-(As*Ar))
  Cn = Cd
```


Note that this operation is faster than e.g. SRC_OVER but slower than SRC

### DST_OVER

```
public static final int DST_OVER
```

Porter-Duff Destination Over Source rule. The destination is composited over the source and the result replaces the destination.

Fs = (1-Ad) and Fd = 1, thus:

```
 cn = (As*Ar)*Cs*(1-Ad) + Ad*Cd
 An = (As*Ar)*(1-Ad) + Ad
```


Note that this can be a very slow drawing operation

### DstIn

```
public static final org.dvb.ui.DVBAlphaComposite DstIn
```

`DVBAlphaComposite` object that implements the opaque DST_IN rule with an alpha (Ar) of 1,0f.

See Also:

`DST_IN`

### DstOut

```
public static final org.dvb.ui.DVBAlphaComposite DstOut
```

`DVBAlphaComposite` object that implements the opaque DST_OUT rule with an alpha (Ar) of 1,0f.

See Also:

`DST_OUT`

### DstOver

```
public static final org.dvb.ui.DVBAlphaComposite DstOver
```

`DVBAlphaComposite` object that implements the opaque DST_OVER rule with an alpha (Ar) of 1,0f.

See Also:

`DST_OVER`

### Src

```
public static final org.dvb.ui.DVBAlphaComposite Src
```

`DVBAlphaComposite` object that implements the opaque SRC rule with an alpha (Ar) of 1,0f.

See Also:

`SRC`

### SRC

```
public static final int SRC
```

Porter-Duff Source rule. The source is copied to the destination. The destination is not used as input.

Fs = 1 and Fd = 0, thus:

```
cn = (As*Ar)*Cs
An = As*Ar
 Cn = Cs
```


Note that this is a fast drawing routine

### SRC_IN

```
public static final int SRC_IN
```

Porter-Duff Source In Destination rule. The part of the source lying inside of the destination replaces the destination.

Fs = Ad and Fd = 0, thus:

```
cn = (As*Ar)*Cs*Ad
An = (As*Ar)*Ad
 Cn = Cs
```


Note that this operation is faster than e.g. SRC_OVER but slower than SRC

### SRC_OUT

```
public static final int SRC_OUT
```

Porter-Duff Source Held Out By Destination rule. The part of the source lying outside of the destination replaces the destination.

Fs = (1-Ad) and Fd = 0, thus:

```
cn = (As*Ar)*Cs*(1-Ad)
An = (As*Ar)*(1-Ad)
 Cn = Cs
```


Note that this operation is faster than e.g. SRC_OVER but slower than SRC

### SRC_OVER

```
public static final int SRC_OVER
```

Porter-Duff Source Over Destination rule. The source is composited over the destination.

Fs = 1 and Fd = (1-(As*Ar)), thus:

```
cn = (As*Ar)*Cs + Ad*Cd*(1-(As*Ar))
An = (As*Ar) + Ad*(1-(As*Ar))
```


Note that this can be a very slow drawing operation

### SrcIn

```
public static final org.dvb.ui.DVBAlphaComposite SrcIn
```

DVBAlphaComposite object that implements the opaque SRC_IN rule with an alpha (Ar) of 1,0f.

See Also:

SRC_IN

### SrcOut

```
public static final org.dvb.ui.DVBAlphaComposite SrcOut
```

DVBAlphaComposite object that implements the opaque SRC_OUT rule with an alpha (Ar) of 1,0f.

See Also:

SRC_OUT

### SrcOver

```
public static final org.dvb.ui.DVBAlphaComposite SrcOver
```

DVBAlphaComposite object that implements the opaque SRC_OVER rule with an alpha (Ar) of 1,0f.

See Also:

SRC_OVER

## Methods

### equals(Object)

```
public boolean equals(java.lang.Object obj)
```

Tests if the specified java.lang.Object is equal to this DVBAlphaComposite object.

Overrides:

equals in class Object

Parameters:

obj - the Object to test for equality.

Returns:

true if obj is a DVBAlphaComposite and has the same values for rule and alpha as this object. Otherwise false shall be returned.

### getAlpha()

```
public float getAlpha()
```

Returns the alpha value of this DVBAlphaComposite. If this DVBAlphaComposite does not have an alpha value, 1,0 is returned.

Returns:

the alpha value of this DVBAlphaComposite.

### getInstance(int)

```
public static org.dvb.ui.DVBAlphaComposite getInstance(int rule)
```

Creates an DVBAlphaComposite object with the specified rule. The value for alpha shall be 1,0f.

Parameters:

rule - the compositing rule.

Returns:

an DVBAlphaComposite object with the specified rule.

### getInstance(int, float)

```
public static org.dvb.ui.DVBAlphaComposite getInstance(int rule, float alpha)
```

Creates an DVBAlphaComposite object with the specified rule and the constant alpha (Ar) to multiply with the alpha of the source (As). The source is multiplied with the specified alpha before being composited with the destination.

Parameters:

rule - the compositing rule.

alpha - the constant alpha (Ar) to be multiplied with the alpha of the source (As). alpha must be a floating point number in the inclusive range [0,0, 1,0].

Returns:

an DVBAlphaComposite object with the specified rule and the constant alpha to multiply with the alpha of the source.

### getRule()

```
public int getRule()
```

Returns the compositing rule of this DVBAlphaComposite.

Returns:

the compositing rule of this DVBAlphaComposite.

# org.dvb.ui

# DVBBufferedImage

### Declaration

```
public class DVBBufferedImage extends java.awt.Image
```

```
java.lang.Object
  |
  +--java.awt.Image
        |
        +--org.dvb.ui.DVBBufferedImage
```

### Description

The DVBBufferedImage subclass describes an java.awt.Image with an accessible buffer of image data. The DVBBufferedImage is an adapter class for java.awt.image.BufferedImage. It supports two different platform dependent sample models TYPE_BASE and TYPE_ADVANCED. Buffered images with the TYPE_BASE have the same sample model as the on screen graphics buffer, thus TYPE_BASE could be CLUT based. TYPE_ADVANCED has a direct color model but it is not specified how many bits are used to store the different color components. By default, a new DVBBufferedImage is transparent. All alpha values are set to 0; Instances of DVBBufferedImage shall be considered to be off-screen images for the purpose of the inherited method Image.getGraphics.

Since:

MHP 1.0.

### Fields

#### TYPE_ADVANCED

```
public static final int TYPE_ADVANCED
```

Represents an image stored in a best possible SampleModel (platform dependent) The image has a DirectColorModel with alpha. The color data in this image is considered not to be pre-multiplied with alpha. The data returned by get-RGB() will be in the TYPE_INT_ARGB color model that is alpha component in bits 24 to 31, the red component in bits 16 to 23, the green component in bits 8 to 15, and the blue component in bits 0 to 7. The data for setRGB() shall be in the TYPE_INT_ARGB color model as well.

Since:

MHP 1.0.

#### TYPE_BASE

```
public static final int TYPE_BASE
```

Represents an image stored in a platform dependent Sample Model. This color model is not visible to applications. The data returned by getRGB() will be in the TYPE_INT_ARGB color model that is alpha component in bits 24 to 31, the red component in bits 16 to 23, the green component in bits 8 to 15, and the blue component in bits 0 to 7. The data for setRGB() shall be in the TYPE_INT_ARGB color model as well.

Since:

MHP 1.0.

### Constructors

#### DVBBufferedImage(int, int)

```
public DVBBufferedImage(int width, int height)
```

Constructs a DVBBufferedImage with the specified width and height. The Sample Model used the image is the native Sample Model (TYPE_BASE) of the implementation. Note that a request can lead to an java.lang.OutOfMemoryError. Applications should be aware of this.

Parameters:

`width` - the width of the created image.

`height` - the height of the created image.

Since:

MHP 1.0.

#### DVBBufferedImage(int, int, int)

```
public DVBBufferedImage(int width, int height, int type)
```

Constructs a new DVBBufferedImage with the specified width and height in the Sample Model specified by type. Note that a request can lead to an java.lang.OutOfMemoryError. Applications should be aware of this.

Parameters:

`width` - the width of the DVBBufferedImage.

`height` - the height of the DVBBufferedImage.

`type` - the ColorSpace of the DVBBufferedImage.

Since:

MHP 1.0.

### Methods

#### createGraphics()

```
public org.dvb.ui.DVBGraphics createGraphics()
```

Creates a `DVBGraphics`, which can be used to draw into this `DVBBufferedImage`. Calls to this method after calls to the `dispose` method on the same instance shall return null.

Returns:

a `DVBGraphics`, used for drawing into this image.

Since:

MHP 1.0.

#### dispose()

```
public void dispose()
```

Disposes of this buffered image. This method releases the resources (e.g. pixel memory) underlying this buffered image. After calling this method:

- the image concerned may not be used again;
- the image shall be considered to have a width and height of -1, -1 as specified for instances of java.awt.Image where the width and height are not yet known;
- the `getGraphics` method may return null.

Since:

MHP 1.0.1.

#### flush()

```
public void flush()
```

Flushes all resources being used to cache optimization information. The underlying pixel data is unaffected. Calls to this method after calls to the `dispose` method on the same instance shall fail silently.

Overrides:

`flush` in class `Image`

#### getGraphics()

```
public java.awt.Graphics getGraphics()
```

This method returns a `java.awt.Graphics` , it is here for backwards compatibility. `createGraphics` is more convenient, since it is declared to return a `DVBGraphics`. Calls to this method after calls to the `dispose` method on the same instance shall return null.

Overrides:

getGraphics in class Image

Returns:

a `Graphics`, which can be used to draw into this image.

#### getHeight()

```
public int getHeight()
```

Returns the height of the `DVBBufferedImage`.

Returns:

the height of this `DVBBufferedImage`.

Since:

MHP 1.0.

### getHeight(ImageObserver)

```
public int getHeight(java.awt.image.ImageObserver observer)
```

Returns the height of the image.

Overrides:

`getHeight` in class `Image`

Parameters:

`observer` - the `ImageObserver` that receives information about the image.

Returns:

the height of the image or `-1` if the height is not yet known.

See Also:

`java.awt.Image.getWidth(ImageObserver), java.awt.image.ImageObserver`

### getImage()

```
public java.awt.Image getImage()
```

Returns a java.awt.Image representing this buffered image. In implementations which implement java.awt.image.BufferedImage this returns a java.awt.image.BufferedImage cast to a java.awt.Image. Otherwise it is implementation dependent whether it returns this image or whether it returns an instance of an underlying platform specific sub-class of java.awt.Image. Calls to this method after calls to the `dispose` method on the same instance shall return null.

Returns:

a java.awt.image representing this buffered image.

Since:

MHP 1.0.

### getProperty(String, ImageObserver)

```
public java.lang.Object getProperty(java.lang.String name, java.awt.image.ImageObserver observer)
```

Returns a property of the image by name. Individual property names are defined by the various image formats. If a property is not defined for a particular image, this method returns the `UndefinedProperty` field. If the properties for this image are not yet known, then this method returns `null` and the `ImageObserver` object is notified later. The property name "comment" should be used to store an optional comment that can be presented to the user as a description of the image, its source, or its author. Calls to this method after calls to the `dispose` method on the same instance shall return null.

Overrides:

`getProperty` in class `Image`

Parameters:

`name` - the property name.

`observer` - the `ImageObserver` that receives notification regarding image information.

Returns:

an `java.lang.Object` that is the property referred to by the specified `name` or `null` if the properties of this image are not yet known.

See Also:

`java.awt.image.ImageObserver`, `java.awt.Image.UndefinedProperty`

### getRGB(int, int)

```
public int getRGB(int x, int y)
```

Returns the specified integer pixel in the default RGB color model (TYPE_INT_ARGB) and default sRGB colorspace. Color conversion takes place if the used Sample Model is not 8-bit for each color component There are only 8-bits of precision for each color component in the returned data when using this method. Note that when a lower precision is used in this buffered image getRGB may return different values than those used in setRGB()

Parameters:

`x` - the x-coordinate of the pixel.

`y` - the y-coordinate of the pixel.

Returns:

an integer pixel in the default RGB color model (TYPE_INT_ARGB) and default sRGB colorspace.

Throws:

`java.lang.ArrayIndexOutOfBoundsException` - if x or y is out of bounds or if the dispose method has been called on this instance.

Since:

MHP 1.0.

### getRGB(int, int, int, int, int[], int, int)

```
public int[] getRGB(int startX, int startY, int w, int h, int[] rgbArray, int offset, int scansize)
```

Returns an array of integer pixels in the default RGB color model (TYPE_INT_ARGB) and default sRGB color space, from a rectangular region of the image data. There are only 8-bits of precision for each color component in the returned data when using this method. With a specified coordinate (x, y) in the image, the ARGB pixel can be accessed in this way:

```
pixel   = rgbArray[offset + (y-startY)*scansize + (x-startX)];
```

Parameters:

`startX` - the x-coordinate of the upper-left corner of the specified rectangular region.

`startY` - the y-coordinate of the upper-left corner of the specified rectangular region.

`w` - the width of the specified rectangular region.

`h` - the height of the specified rectangular region.

`rgbArray` - if not `null`, the rgb pixels are written here.

`offset` - offset into the `rgbArray`.

`scansize` - scanline stride for the `rgbArray`.

Returns:

array of ARGB pixels.

Throws:

java.lang.ArrayIndexOutOfBoundsException - if the specified portion of the image data is out of bounds or if the dispose method has been called on this instance.

Since:

MHP 1.0.

### getScaledInstance(int, int, int)

```
public java.awt.Image getScaledInstance(int width, int height, int hints)
```

Creates a scaled version of this image. A new Image object is returned which will render the image at the specified width and height by default. The new Image object may be loaded asynchronously even if the original source image has already been loaded completely. If either the width or height is a negative number then a value is substituted to maintain the aspect ratio of the original image dimensions. If the dispose method has been called on this instance then null shall be returned.

Overrides:

getScaledInstance in class Image

Parameters:

width - the width to which to scale the image.

height - the height to which to scale the image.

hints - flags to indicate the type of algorithm to use for image resampling.

Returns:

a scaled version of the image.

### getSource()

```
public java.awt.image.ImageProducer getSource()
```

Returns the object that produces the pixels for the image.

Overrides:

getSource in class Image

Returns:

the java.awt.image.ImageProducer that is used to produce the pixels for this image.

If the dispose method has been called on this instance then null shall be returned. The source returned by this method is platform generated to provide access to the current contents of the DVBBufferedImage buffer.

See Also:

java.awt.image.ImageProducer

### getSubimage(int, int, int, int)

```
public org.dvb.ui.DVBBufferedImage getSubimage(int x, int y, int w, int h)
throws DVBRasterFormatException
```

Returns a subimage defined by a specified rectangular region. The returned DVBBufferedImage shares the same data array as the original image. If the dispose method has been called on this instance then null shall be returned.

Parameters:

x - the x-coordinate of the upper-left corner of the specified rectangular region.

y - the y-coordinate of the upper-left corner of the specified rectangular region.

w - the width of the specified rectangular region.

h - the height of the specified rectangular region.

Returns:

a DVBBufferedImage that is the subimage of this DVBBufferedImage.

Throws:

DVBRasterFormatException - if the specified area is not contained within this DVBBufferedImage.

Since:

MHP 1.0.

### getWidth()

```
public int getWidth()
```

Returns the width of the DVBBufferedImage.

Returns:

the width of this DVBBufferedImage.

Since:

MHP 1.0.

### getWidth(ImageObserver)

```
public int getWidth(java.awt.image.ImageObserver observer)
```

Returns the width of the image. If the width is not known yet then the java.awt.image.ImageObserver is notified later and -1 is returned.

Overrides:

getWidth in class Image

Parameters:

observer - the ImageObserver that receives information about the image.

Returns:

the width of the image or -1 if the width is not yet known.

See Also:

java.awt.Image.getHeight(ImageObserver), java.awt.image.ImageObserver

### setRGB(int, int, int)

```
public void setRGB(int x, int y, int rgb)
```

Sets a pixel in this DVBBufferedImage to the specified ARGB value. The pixel is assumed to be in the default RGB color model, TYPE_INT_ARGB, and default sRGB color space. Calls to this method after calls to the dispose method on the same instance shall throw an ArrayIndexOutOfBoundsException.

Parameters:

x - the x-coordinate of the pixel to set.

y - the y-coordinate of the pixel to set.

rgb - the ARGB value.

Since:

MHP 1.0.

### setRGB(int, int, int, int, int[], int, int)

```
public void setRGB(int startX, int startY, int w, int h, int[] rgbArray, int offset, int scansize)
```

Sets an array of integer pixels in the default RGB color model (TYPE_INT_ARGB) and default sRGB color space, into a rectangular portion of the image data. There are only 8-bits of precision for each color component in the returned data when using this method. With a specified coordinate (x, y) in the this image, the ARGB pixel can be accessed in this way:

```
pixel   = rgbArray[offset + (y-startY)*scansize + (x-startX)];
```

WARNING: No dithering takes place.

Calls to this method after calls to the `dispose` method on the same instance shall throw an `ArrayIndexOutOfBounds-Exception`.

Parameters:

`startX` - the x-coordinate of the upper-left corner of the specified rectangular region.

`startY` - the y-coordinate of the upper-left corner of the specified rectangular region.

`w` - the width of the specified rectangular region.

`h` - the height of the specified rectangular region.

`rgbArray` - the ARGB pixels.

`offset` - offset into the `rgbArray`.

`scansize` - scanline stride for the `rgbArray`.

Since:

MHP 1.0.

### toString()

```
public java.lang.String toString()
```

Returns a `String` representation of this `DVBBufferedImage` object and its values.

Overrides:

`toString` in class `Object`

Returns:

a `String` representing this `DVBBufferedImage`.

# org.dvb.ui

# DVBColor

### Declaration

```
public class DVBColor extends javax.tv.graphics.AlphaColor
```

```
java.lang.Object
  |
  +--java.awt.Color
        |
        +--javax.tv.graphics.AlphaColor
              |
              +--org.dvb.ui.DVBColor
```

### All Implemented Interfaces:

```
java.io.Serializable, java.awt.Transparency
```

### Description

A Color class which adds the notion of alpha. Because DVBColor extends Color the signatures in the existing classes do not change. Classes like Component should work with DVBColor internally. Instances of this class are a container for the values which are passed in to the constructor. Any approximations made by the platform are made when the colors are used. Note: org.dvb.ui.DVBColor adds support for alpha (compared to JDK1.1.8) and is intended to be compatible with the JDK1.2 java.awt.Color class - since org.dvb.ui.DVBColor extends javax.tv.graphics.AlphaColor which in turn extends java.awt.Color. In implementations where java.awt.Color supports alpha, such as JDK1.2, etc., the alpha-related methods in org.dvb.ui.DVBColor could just call super.

### Since:

MHP 1.0.

### Constructors

### DVBColor(Color)

```
public DVBColor(java.awt.Color c)
```

Constructs a new DVBColor using the specified color. If c supports alpha, e.g. if it is an instance of javax.tv.graphics.AlphaColor or JDK 1.2's java.awt.Color, then the alpha value of c shall be used. If this color has no alpha value, alpha will be set to 255 (opaque).

### Parameters:

`c` - the java.awt.Color used to create a new DVBColor.

### DVBColor(float, float, float, float)

```
public DVBColor(float r, float g, float b, float a)
```

Creates an sRGB color with the specified red, green, blue, and alpha values in the range (0,0 to 1,0). The actual color used in rendering will depend on finding the best match given the color space available for a given output device.

### Parameters:

`r` - the red component.

`g` - the green component.

`b` - the blue component.

`a` - the alpha component.

See Also:

java.awt.Color.getRed(), java.awt.Color.getGreen(), java.awt.Color.getBlue(), getAlpha(), getRGB()

### DVBColor(int, boolean)

```
public DVBColor(int rgba, boolean hasalpha)
```

Creates an sRGB color with the specified combined RGBA value consisting of the alpha component in bits 24 to 31, the red component in bits 16 to 23, the green component in bits 8 to 15, and the blue component in bits 0 to 7. If the hasalpha argument is False, alpha is defaulted to 255.

Parameters:

`rgba` - the combined RGBA components.

`hasalpha` - true if the alpha bits are valid, false otherwise.

See Also:

java.awt.Color.getRed(), java.awt.Color.getGreen(), java.awt.Color.getBlue(), getAlpha(), getRGB()

### DVBColor(int, int, int, int)

```
public DVBColor(int r, int g, int b, int a)
```

Creates an sRGB color with the specified red, green, blue, and alpha values in the range (0 to 255).

Parameters:

`r` - the red component.

`g` - the green component.

`b` - the blue component.

`a` - the alpha component.

See Also:

java.awt.Color.getRed(), java.awt.Color.getGreen(), java.awt.Color.getBlue(), getAlpha(), getRGB()

## Methods

### brighter()

```
public java.awt.Color brighter()
```

Creates a brighter version of this color. This method applies an arbitrary scale factor to each of the three RGB components of the color to create a brighter version of the same color. Although brighter and darker are inverse operations, the results of a series of invocations of these two methods may be inconsistent because of rounding errors. The alpha value shall be preserved.

Overrides:

`brighter` in class `AlphaColor`

Returns:

a new DVBColor object (cast to a java.awt.Color object) representing a brighter version of this color. Applications can recast it to a org.dvb.ui.DVBColor object.

See Also:

java.awt.Color.brighter()

### darker()

```
public java.awt.Color darker()
```

Creates a darker version of this color. This method applies an arbitrary scale factor to each of the three RGB components of the color to create a darker version of the same color. Although brighter and darker are inverse operations, the results of a series of invocations of these two methods may be inconsistent because of rounding errors. The alpha value shall be preserved.

Overrides:

`darker` in class `AlphaColor`

Returns:

a new DVBColor object (cast to a java.awt.Color object), representing a darker version of this color. Applications can recast it to a org.dvb.ui.DVBColor object.

See Also:

`java.awt.Color.darker()`

### equals(Object)

```
public boolean equals(java.lang.Object obj)
```

Determines whether another object is equal to this color. The result is true if and only if the argument is not null and is a DVBColor object that has the same red, green, blue and alpha values as this object.

Overrides:

`equals` in class `AlphaColor`

Parameters:

`obj` - - the object to compare with.

Returns:

true if the objects are the same; false otherwise.

Since:

MHP 1.0.

### getAlpha()

```
public int getAlpha()
```

Returns the alpha component. In the range 0 to 255.

Overrides:

`getAlpha` in class `AlphaColor`

Returns:

the alpha component.

See Also:

`getRGB()`

### getRGB()

```
public int getRGB()
```

Returns the RGB value representing the color in the default sRGB ColorModel. (Bits 24 to 31 are alpha, 16 to 23 are red, 8 to 15 are green, 0 to 7 are blue).

Overrides:

`getRGB` in class `AlphaColor`

Returns:

the RGB value representing the color in the default sRGB ColorModel.

Since:

MHP 1.0.

See Also:

`java.awt.Color.getRed()`, `java.awt.Color.getGreen()`, `java.awt.Color.getBlue()`, `getAlpha()`

### toString()

```
public java.lang.String toString()
```

Creates a string that represents this color and indicates the values of its ARGB components.

Overrides:

`toString` in class `AlphaColor`

Returns:

a representation of this color as a String object.

Since:

MHP 1.0

# org.dvb.ui

# DVBGraphics

### Declaration

```
public abstract class DVBGraphics extends java.awt.Graphics2D
```

```
java.lang.Object
  |
  +--java.awt.Graphics
        |
        +--java.awt.Graphics2D
              |
              +--org.dvb.ui.DVBGraphics
```

### Description

The `DVBGraphics` class is a adapter class to support alpha compositing in a GEM device. At all times, the DVBAlphaComposite returned by getDVBComposite and the Composite returned by the getComposite method inherited from Graphics2D shall be consistent. This means that the results of calling getRule and getAlpha on those shall be identical. In GEM devices all Graphics Objects are DVBGraphics objects. Thus one can get a DVBGraphics by casting a given Graphics object. The normal compositing rule used is DVBAlphaComposite.SRC_OVER. Note that the default rule of SRC_OVER may not give the highest performance. Under many circumstances, applications will find that the SRC rule will give higher performance. The intersection between setDVBComposite in this class and the `setPaintMode` and `setXORMode` methods inherited from java.awt.Graphics shall be as follows:

- Calling `setPaintMode` on an instance of this class shall be equivalent to calling `setDVBComposite(DVBAlphaComposite.SrcOver)`.
- The present document does not tighten, refine or detail the definition of the `setXORMode` beyond what is specified for the parent class.

NOTE: Implementations of XOR mode may change colours with alpha to without and vice versa (reversibly).

Since:

MHP1.0.

See Also:

`java.awt.Graphics`

### Constructors

### DVBGraphics()

```
protected DVBGraphics()
```

Constructs a new `DVBGraphics` object. This constructor is the default constructor for a graphics context.

Since `DVBGraphics` is an abstract class, applications cannot call this constructor directly. DVBGraphics contexts are obtained from other DVBGraphics contexts or are created by casting java.awt.Graphics to DVBGraphics.

Since:

MHP 1.0.

See Also:

`java.awt.Graphics.create()`, `java.awt.Component.getGraphics()`

### Methods

### getAvailableCompositeRules()

```
public abstract int[] getAvailableCompositeRules()
```

Returns all available Porter-Duff Rules for this specific Graphics context. E.g. a devices could support the SRC_OVER rule when using a destination which does not have Alpha or where the alpha is null, while this rule is not available when drawing on a graphic context where the destination has alpha. Which rules are supported for the different graphics objects is defined in the Minimum Platform Capabilities of the GEM spec.

Returns:

all available Porter-Duff Rules for this specific Graphics context.

Since:

MHP 1.0.

### getBestColorMatch(Color)

```
public org.dvb.ui.DVBColor getBestColorMatch(java.awt.Color c)
```

Returns the best match for the specified Color as a DVBColor, in a device-dependent manner, as constrained by the GEM graphics reference model.

Parameters:

`c` - the specified Color.

Returns:

the best DVBColor match for the specified Color.

Since:

MHP 1.0.

### getColor()

```
public abstract java.awt.Color getColor()
```

Gets this graphics context's current color. This will return a DVBColor cast to java.awt.Color.

Overrides:

`getColor` in class `Graphics`

Returns:

this graphics context's current color.

Since:

MHP 1.0.

See Also:

`DVBColor`, `java.awt.Color`, `setColor(Color)`

### getDVBComposite()

```
public abstract org.dvb.ui.DVBAlphaComposite getDVBComposite()
```

Returns the current `DVBAlphaComposite` in the `DVBGraphics` context. This method could delegate to a java.awt.Graphics2D object where available

Returns:

the current `DVBGraphics` `DVBAlphaComposite`, which defines a compositing style.

Since:

MHP 1.0.

See Also:

`setDVBComposite(DVBAlphaComposite)`

### getType()

```
public int getType()
```

Returns the Sample Model (DVBBufferedImage.TYPE_BASE, DVBBufferedImage.TYPE_ADVANCED) which is used in the on/off screen buffer this graphics object draws into.

Returns:

the type of the Sample Model.

Since:

MHP 1.0.

See Also:

`DVBBufferedImage`

### setColor(Color)

```
public abstract void setColor(java.awt.Color c)
```

Sets this graphics context's current color to the specified color. All subsequent graphics operations using this graphics context use this specified color. Note that color c can be a DVBColor.

Overrides:

`setColor` in class `Graphics`

Parameters:

c - the new rendering color.

Since:

MHP 1.0.

See Also:

java.awt.Color, DVBColor, getColor()

### setDVBComposite(DVBAlphaComposite)

```
public abstract void setDVBComposite(org.dvb.ui.DVBAlphaComposite comp)
throws UnsupportedDrawingOperationException
```

Sets the DVBAlphaComposite for the DVBGraphics context. The DVBAlphaComposite is used in all drawing methods such as drawImage, drawString, draw, and fill. It specifies how new pixels are to be combined with the existing pixels on the graphics device during the rendering process.

This method could delegate to a Graphics2D object or to an native implementation

Parameters:

comp - the DVBAlphaComposite object to be used for rendering.

Throws:

UnsupportedDrawingOperationException - when the requested Porter-Duff rule is not supported by this graphics context.

Since:

MHP 1.0.

See Also:

java.awt.Graphics.setXORMode(Color), java.awt.Graphics.setPaintMode(), DVBAlphaComposite

### toString()

```
public java.lang.String toString()
```

Returns a String object representing this DVBGraphics object's value.

Overrides:

toString in class Graphics

Returns:

a string representation of this graphics context.

Since:

MHP 1.0.

# org.dvb.ui
# DVBRasterFormatException

### Declaration

```
public class DVBRasterFormatException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.ui.DVBRasterFormatException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This exception is thrown for some invalid operations on instances of `DVBBufferedImage`. The precise conditions are defined in the places where this exception is thrown.

### Since:

MHP 1.0.1.

### See Also:

```
DVBBufferedImage
```

### Constructors

#### DVBRasterFormatException(String)

```
public DVBRasterFormatException(java.lang.String s)
```

Constructs an instance of `DVBRasterFormatException` with the specified detail message.

### Parameters:

`s` - the detail message.

### Since:

MHP1.0.

# org.dvb.ui
# DVBTextLayoutManager

### Declaration

```
public class DVBTextLayoutManager implements org.havi.ui.HTextLayoutManager
```

```
java.lang.Object
  |
  +--org.dvb.ui.DVBTextLayoutManager
```

### All Implemented Interfaces:

```
org.havi.ui.HTextLayoutManager
```

### Description

The DVBTextLayoutManager provides a text rendering layout mechanism for the org.havi.ui.HStaticText org.havi.ui.HText and org.havi.ui.HTextButton classes.

The semantics of the rendering behaviour and the settings are specified in the "Text presentation" annex of the present document. The DVBTextLayoutManager renders the text according to the semantics described in that annex.

### Fields

#### HORIZONTAL_CENTER

```
public static final int HORIZONTAL_CENTER
```

The text should be centered horizontally.

#### HORIZONTAL_END_ALIGN

```
public static final int HORIZONTAL_END_ALIGN
```

The text should be horizontally to the horizontal end side (e.g. when start corner is upper left and line orientation horizontal, meaning text that is read left to right from top to bottom, this implies alignment to right).

#### HORIZONTAL_START_ALIGN

```
public static final int HORIZONTAL_START_ALIGN
```

The text should be aligned horizontally to the horizontal start side (e.g. when start corner is upper left and line orientation horizontal, meaning text that is read left to right from top to bottom, this implies alignment to left).

#### LINE_ORIENTATION_HORIZONTAL

```
public static final int LINE_ORIENTATION_HORIZONTAL
```

Horizontal line orientation.

#### LINE_ORIENTATION_VERTICAL

```
public static final int LINE_ORIENTATION_VERTICAL
```

Vertical line orientation.

#### START_CORNER_LOWER_LEFT

```
public static final int START_CORNER_LOWER_LEFT
```

Lower left text start corner.

#### START_CORNER_LOWER_RIGHT

```
public static final int START_CORNER_LOWER_RIGHT
```

Lower right text start corner.

#### START_CORNER_UPPER_LEFT

```
public static final int START_CORNER_UPPER_LEFT
```

Upper left text start corner.

#### START_CORNER_UPPER_RIGHT

```
public static final int START_CORNER_UPPER_RIGHT
```

Upper right text start corner.

#### VERTICAL_CENTER

```
public static final int VERTICAL_CENTER
```

The text should be centered vertically.

### VERTICAL_END_ALIGN

```
public static final int VERTICAL_END_ALIGN
```

The text should be aligned vertically to the vertical end side (e.g. when start corner is upper left and line orientation horizontal, meaning text that is read left to right from top to bottom, this implies alignment to bottom).

This is defined by the clause "Vertical limits" in the "Text presentation" annex of the present document.

### VERTICAL_START_ALIGN

```
public static final int VERTICAL_START_ALIGN
```

The text should be aligned vertically to the vertical start side (e.g. when start corner is upper left and line orientation horizontal, meaning text that is read left to right from top to bottom, this implies alignment to top).

This is defined by the clause "Vertical limits" in the "Text presentation" annex of the present document.

## Constructors

### DVBTextLayoutManager()

```
public DVBTextLayoutManager()
```

Constructs a DVBTextLayoutManager object with default parameters (HORIZONTAL_START_ALIGN, VERTICAL_START_ALIGN, LINE_ORIENTATION_HORIZONTAL, START_CORNER_UPPER_LEFT, wrap = true, linespace = (point size of the default font for HVisible) + 7, letterspace = 0, horizontalTabSpace = 56)

### DVBTextLayoutManager(int, int, int, int, boolean, int, int, int)

```
public DVBTextLayoutManager(int horizontalAlign, int verticalAlign, int lineOrientation,
int startCorner, boolean wrap, int linespace, int letterspace, int horizontalTabSpace)
```

Constructs a DVBTextLayoutManager object.

### Parameters:

`horizontalAlign` - Horizontal alignment setting.

`verticalAlign` - Vertical alignment setting.

`lineOrientation` - Line orientation setting.

`startCorner` - Starting corner setting.

`wrap` - Text wrapping setting.

`linespace` - Line spacing setting expressed in points.

`letterspace` - Letterspacing adjustment relative to the default letterspacing. Expressed in units of 1/256th point as the required increase in the spacing between consecutive characters. May be either positive or negative.

`horizontalTabSpace` - Horizontal tabulation setting in points.

## Methods

### addTextOverflowListener(TextOverflowListener)

```
public void addTextOverflowListener(org.dvb.ui.TextOverflowListener l)
```

Register a TextOverflowListener that will be notified if the text string does not fit in the component when rendering.

### Parameters:

`l` - a listener object.

### getHorizontalAlign()

```
public int getHorizontalAlign()
```

Get the horizontal alignment.

Returns:

Horizontal alignment setting.

### getHorizontalTabSpacing()

```
public int getHorizontalTabSpacing()
```

Get the horizontal tabulation spacing.

Returns:

the horizontal tabulation spacing.

### getInsets()

```
public java.awt.Insets getInsets()
```

Returns the insets set by the setInsets method. These Insets are added to the ones passed to the `render` method for rendering the text. When not previously set, zero Insets are returned.

Returns:

Insets set by the setInsets method.

### getLetterSpace()

```
public int getLetterSpace()
```

Get the letter space setting. This is a 16 bit signed integer specifying in units of 1/256th point the required increase in the spacing between consecutive characters. It corresponds to the "track" parameter in the GEM text rendering rules.

Returns:

letter space setting.

### getLineOrientation()

```
public int getLineOrientation()
```

Get the line orientation.

Returns:

Line orientation setting.

### getLineSpace()

```
public int getLineSpace()
```

Get the line space setting.

Returns:

line space setting or -1, if the default line spacing is determined from the size of the default font used.

### getStartCorner()

```
public int getStartCorner()
```

Get the starting corner.

Returns:

Starting corner setting.

getTextWrapping()

```
public boolean getTextWrapping()
```

Get the text wrapping setting.

Returns:

text wrapping setting.

getVerticalAlign()

```
public int getVerticalAlign()
```

Get the vertical alignment.

Returns:

Vertical alignment setting.

removeTextOverflowListener(TextOverflowListener)

```
public void removeTextOverflowListener(org.dvb.ui.TextOverflowListener l)
```

Removes a TextOverflowListener that has been registered previously.

Parameters:

`l` - a listener object.

render(String, Graphics, HVisible, Insets)

```
public void render(java.lang.String markedUpString, java.awt.Graphics g, org.havi.ui.HVisible v,
java.awt.Insets insets)
```

Render the string. The `HTextLayoutManager` should use the passed `HVisible` object to determine any additional information required to render the string, e.g. `Font`, `Color`, etc.

The text should be laid out in the layout area, which is defined by the bounds of the specified `HVisible` , after subtracting the insets. If the insets are `null` the full bounding rectangle is used as the area to render text into.

The `HTextLayoutManager` shall not modify the clipping rectangle of the `Graphics` object.

Specified By:

`render` in interface `HTextLayoutManager`

Parameters:

`markedUpString` - the string to render.

`g` - the graphics context, including a clipping rectangle which encapsulates the area within which rendering is permitted. If a valid insets value is passed to this method then text must only be rendered into the bounds of the widget after the insets are subtracted. If the insets value is `null` then text is rendered into the entire bounding area of the `HVisible` . It is implementation specific whether or not the renderer takes into account the intersection of the clipping rectangle in each case for optimization purposes.

`v` - the `HVisible` into which to render.

`insets` - the insets to determine the area in which to layout the text, or `null`.

### setHorizontalAlign(int)

```
public void setHorizontalAlign(int horizontalAlign)
```

Set the horizontal alignment. The setting shall be one of `HORIZONTAL_CENTER`, `HORIZONTAL_END_ALIGN` or `HORIZONTAL_START_ALIGN`. The failure mode if other values are used is implementation dependent.

Parameters:

`horizontalAlign` - Horizontal alignment setting.

### setHorizontalTabSpacing(int)

```
public void setHorizontalTabSpacing(int horizontalTabSpace)
```

Set the horizontal tabulation spacing.

Parameters:

`horizontalTabSpace` - tab spacing in points.

### setInsets(Insets)

```
public void setInsets(java.awt.Insets insets)
```

Sets the insets which shall be used by this DVBTextLayoutManager to provide a "virtual margin". These shall be added to the insets passed to the `Render` method (which are to be considered as "bounds"). If this method is not called, the default insets are 0 at each edge.

Parameters:

`insets` - Insets that should be used.

### setLetterSpace(int)

```
public void setLetterSpace(int letterSpace)
```

Set the letter space setting. This is a 16 bit signed integer specifying in units of 1/256th point the required increase in the spacing between consecutive characters. It corresponds to the "track" parameter in the GEM text rendering rules.

Parameters:

`letterSpace` - letter space setting.

### setLineOrientation(int)

```
public void setLineOrientation(int lineOrientation)
```

Set the line orientation. The setting shall be one of `LINE_ORIENTATION_VERTICAL`, `LINE_ORIENTATION_HORIZONTAL`. The failure mode if other values are used is implementation dependent.

Parameters:

`lineOrientation` - Line orientation setting.

### setLineSpace(int)

```
public void setLineSpace(int lineSpace)
```

Set the line space setting. Using -1 as the line space setting shall cause the line spacing to be set to the default value as defined for the no-argument constructor for this class.

Parameters:

`lineSpace` - line space setting.

### setStartCorner(int)

```
public void setStartCorner(int startCorner)
```

Set the starting corner. The setting shall be one of `START_CORNER_UPPER_LEFT`, `START_CORNER_UPPER_RIGHT`, `START_CORNER_LOWER_LEFT` or `START_CORNER_LOWER_RIGHT`. The failure mode if other values are used is implementation dependent.

#### Parameters:

`startCorner` - Starting corner setting.

### setTextWrapping(boolean)

```
public void setTextWrapping(boolean wrap)
```

Set the text wrapping setting.

#### Parameters:

`wrap` - Text wrapping setting.

### setVerticalAlign(int)

```
public void setVerticalAlign(int verticalAlign)
```

Set the vertical alignment. The setting shall be one of `VERTICAL_CENTER`, `VERTICAL_END_ALIGN` or `VERTICAL_START_ALIGN`. The failure mode if other values are used is implementation dependent.

#### Parameters:

`verticalAlign` - Vertical alignment setting.

# org.dvb.ui

# FontFactory

#### Declaration

```
public class FontFactory

java.lang.Object
  |
  +--org.dvb.ui.FontFactory
```

#### Description

Provides a mechanism for applications to instantiate fonts that are not built into the system. The two constructors of this class allow fonts to be downloaded either through the font index file of the application or directly from a font file in the format(s) specified in the main body of the present document.

#### Constructors

### FontFactory()

```
public FontFactory()
throws FontFormatException, IOException
```

Constructs a FontFactory for the font index file bound to this application in the application signalling. The call to the constructor is synchronous and shall block until the font index file has been retrieved or an an exception is thrown.

#### Throws:

`FontFormatException` - if there is an error in the font index file bound with the application.

`java.io.IOException` - if there is no font index file bound with the application, or if there is an error attempting to access the data in that file.

### FontFactory(URL)

```
public FontFactory(java.net.URL u)
throws IOException, FontFormatException
```

Constructs a FontFactory for the font file found at the given location. The call to the constructor is synchronous and shall block until the font file has been retrieved or an exception is thrown.

Parameters:

`u` - The location of the font file.

Throws:

`java.io.IOException` - if there is an error attempting to access the data referenced by the URL.

`java.lang.IllegalArgumentException` - if the URL is not both valid and supported.

`java.lang.SecurityException` - if access to the specified URL is denied by security policy.

`FontFormatException` - if the file at that URL is not a valid font file as specified in the main body of the present document.

### Methods

### createFont(String, int, int)

```
public java.awt.Font createFont(java.lang.String name, int style, int size)
throws FontNotAvailableException, FontFormatException, IOException
```

Creates a font object from the font source associated with this FontFactory. This font will remain valid even if the FontFactory is no longer reachable from application code. The name returned by Font.getName() might not be the same as the name supplied, for example, it might have a string prepended to it that identifies the source FontFactory in a platform-dependant manner. For FontFactory instances bound to the font index file of an application, the call to the method is synchronous and shall block until either an exception is thrown or any required network access has completed.

The value of the style argument must be as defined in java.awt.Font. Valid values are the following:

- java.awt.Font.PLAIN.
- java.awt.Font.BOLD.
- java.awt.Font.ITALIC.
- java.awt.Font.BOLD + java.awt.Font.ITALIC.

Parameters:

`name` - the font name.

`style` - the constant style used, such as java.awt.Font.PLAIN.

`size` - the point size of the font.

Throws:

`FontNotAvailableException` - if a font with given parameters cannot be located or created.

`java.io.IOException` - if there is an error retrieving a font from the network. Thrown only for font factory instances bound to the font index file of an application.

`java.lang.IllegalArgumentException` - if the style parameter is not in the set of valid values, or if the size parameter is zero or negative.

`FontFormatException` - if the font file is not a valid font file as specified in the main body of the present document. Thrown only for font factory instances bound to the font index file of an application.

# org.dvb.ui FontFormatException

### Declaration

```
public class FontFormatException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.ui.FontFormatException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown when attempt is made to read a file describing a font when the contents of that file are not valid.

### Constructors

#### FontFormatException()

```
public FontFormatException()
```

Constructs a `FontFormatException` with `null` as its error detail message.

#### FontFormatException(String)

```
public FontFormatException(java.lang.String s)
```

Constructs a `FontFormatException` with the specified detail message. The error message string `s` can later be retrieved by the `java.lang.Throwable.getMessage()` method of class `java.lang.Throwable`.

### Parameters:

`s` - the detail message.

# org.dvb.ui FontNotAvailableException

### Declaration

```
public class FontNotAvailableException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.ui.FontNotAvailableException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown when attempt is made to instantiate a font that cannot be located.

### Constructors

#### FontNotAvailableException()

```
public FontNotAvailableException()
```

Constructs a `FontNotAvailableException` with `null` as its error detail message.

#### FontNotAvailableException(String)

```
public FontNotAvailableException(java.lang.String s)
```

Constructs a `FontNotAvailableException` with the specified detail message. The error message string `s` can later be retrieved by the `java.lang.Throwable.getMessage()` method of class `java.lang.Throwable`.

#### Parameters:

`s` - the detail message.

# org.dvb.ui

# TestOpacity

### Declaration

```
public interface TestOpacity
```

### All Known Implementing Classes:

```
org.havi.ui.HComponent
```

### Description

Interface implemented by Components or Containers in order to allow the platform to query whether their paint method is fully opaque.

### Methods

#### isOpaque()

```
public boolean isOpaque()
```

Returns true if the entire area of the component as given by the getBounds method, is fully opaque. Hence its paint method (or surrogate methods) guarantees that all pixels are painted in an opaque Color.

Classes implementing this interface shall return true from their implementation of this method if and only if their implementation can guarantee full opacity. The consequences of an invalid overridden value are implementation specific.

#### Returns:

true if all the pixels with the java.awt.Component#getBounds method are fully opaque, otherwise false.

# org.dvb.ui

# TextOverflowListener

### Declaration

```
public interface TextOverflowListener extends java.util.EventListener
```

### All Superinterfaces:

```
java.util.EventListener
```

### Description

The TextOverflowListener is an interface that an application may implement and register in the DVBTextLayout-Manager. This listener will be notified if the text string does not fit within the component as a result of a call to the render method. It is the rendering process which triggers this event to be dispatched. The timing of this is implementation dependent.

### Methods

#### notifyTextOverflow(String, HVisible, boolean, boolean)

```
public void notifyTextOverflow(java.lang.String markedUpString, org.havi.ui.HVisible v,
boolean overflowedHorizontally, boolean overflowedVertically)
```

This method is called by the DVBTextLayoutManager if the text does not fit within the component

#### Parameters:

`markedUpString` - the string that was successfully rendered within the component.

`v` - the HVisible object that was being rendered.

`overflowedHorizontally` - true if the text overflowed the bounds of the component in the horizontal direction; otherwise false.

`overflowedVertically` - true if the text overflowed the bounds of the component in the vertical direction; otherwise false.

# org.dvb.ui

# UnsupportedDrawingOperationException

### Declaration

```
public class UnsupportedDrawingOperationException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.ui.UnsupportedDrawingOperationException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

The `UnsupportedDrawingOperationException` class represents an exception that is thrown if an drawing operation is not supported on this platform. E.g. DVBGraphics.setComposite could throw an Exception when setting the DST_IN rule on some devices while the SRC_OVER rule will always work.

### Since:

MHP 1.0.

### Constructors

#### UnsupportedDrawingOperationException(String)

```
public UnsupportedDrawingOperationException(java.lang.String s)
```

Constructs an instance of `UnsupportedDrawingOperationException` with the specified detail message.

Parameters:

s - the detail message.

Since:

MHP1.0.

# Annex V:
# Void

# Annex W (informative): DVB-J examples

## W.1 DVB-J examples from MHP

MHP [1], annex W is included in the present document.

## W.2 Example of enumeration extension

To illustrate the importance of the requirement in clause 4.1.4.1, consider an application that is written to the GEM specification which wishes to query the type of return channel connection, and react accordingly. Such code might be written in the following manner:

```
import org.dvb.net.rc.RCInterface;
import org.dvb.net.rc.RCInterfaceManager;

public class AppRCTester {

    /**
     * Set up the return channel.
     * @return true if it was successfully set up, false otherwise.
     */
    public boolean setUpRC() {
        RCInterface[] ifs = RCInterfaceManager.getInstance().getInterfaces();
        boolean success = false;
        for (int i = 0; !success && i < ifs.length; i++) {
            RCInterface inter = ifs[i];
            switch(inter.getType()) {
                case TYPE_CATV:
                    success = setupCATV(inter);
                    break;
                case TYPE_DECT:
                    success = setupDECT(inter);
                    break;
                case TYPE_ISDN:
                    success = setupISDN(inter);
                    break;
                case TYPE_LMDS:
                    success = setupLMDS(inter);
                    break;
                case TYPE_MATV:
                    success = setupMATV(inter);
                    break;
                case TYPE_UNKNOWN:
                case TYPE_OTHER:
                    success = setupOTHER(inter);
                    break;
                default:
                    // Do nothing - this always fails
        }
        return success;
    }

    .... definition of methods setupCATV et al.
}
```

If it were permissible for a GEM terminal specification to sub-divide `TYPE_CATV` by introducing new values into the enumeration, then this code would always fail.

If a terminal specification needs to sub-divide the values of an enumeration, it may do so by introducing a new method to report the sub-divisions. For example, to sub-divide `TYPE_CATV`, a terminal specification could introduce an interface and a set of values like the following:

```
package org.specbody.net.rc;

    /**
     * On specbody terminals, all instances of org.dvb.net.rc.RCInterface
     * for which getType() returns TYPE_CATV shall implement this interface.
     */

public interface CATVRCInterface {

    public final static int TYPE_CATV_SUBTYPE_1 = 1;
    public final static int TYPE_CATV_SUBTYPE_2 = 2;
    public final static int TYPE_CATV_SUBTYPE_3 = 3;

    /**
     * @returns one of TYPE_CATV_SUBTYPE_1, TYPE_CATV_SUBTYPE_2
     *                  or TYPE_CATV_SUBTYPE_3
     */
    public int getCATVType();

}
```

Note that this extension mechanism works for `org.dvb.net.rc.RCInterface` because instances of this class are always created by a factory method that is a part of the platform. This particular method for extending the behaviour of GEM would not work if the enumeration value were returned by a method in a class with a constructor that is accessible to applications, because it would be impossible to mandate that all instances conforming to certain criteria implement an additional interface. In this case, other extension mechanisms would need to be employed.

# W.3 Example of testing for optional APIs

```
    /**
     * This is an example of writing application code that uses
     * an optional API that may not be present on all
     * platforms.  In this case, the application tests for
     * the presence of the class org.dvb.net.rc.RCInterfaceManager.
     * If it is present, then the application has available to
     * it that API, and other APIs needed for the return channel.
     * <p>
     * This is the main Xlet class
     **/

    public class MyXlet implements javax.tv.xlet.Xlet {

        ...

        public RCFeatures rcFeatures = null;
        // This object is the gateway to all of the Xlet
        // code that relies on the return channel.  If
        // null, the Xlet is running on a box with no
        // return channel capability.

        public void initXlet(javax.tv.xlet.XletContext ctx) {
        ...

        try {
            Class c = Class.forName(
                              "org.dvb.net.rc.RCInterfaceManager");
            // We have the return channel API
            c = Class.forName("RCFeaturesImpl");
                        // An Xlet class
            rcFeatures = c.newInstance();
                        // Calls default constructor
            rcFeatures.init();
        } catch (ClassNotFoundException ex) {
            // No return channel, so we leave rcFeatures null.
        }
        }
    }

    /**
```

```
 * This interface is used by the Xlet to call into
 * code that uses the return channel in any way.  It
 * is essential that the only code in the Xlet that
 * uses classes not present on the EB profile be reached
 * via the class that implements this interface.  If an
 * Xlet directly references an API from some other place, then
 * the entire Xlet might fail to load on valid MHP
 * implementations.
 * <p>
 * In technical terms, when a class is loaded, a valid Java
 * implementation may load the transitive closure of all
 * statically-referenced classes, and fail to load the first
 * class if any of the other classes are not found.  Thus,
 * if an Xlet directly references an API that is not present,
 * the entire Xlet could fail to load on valid MHP
 * implementations.  As examples, static references can
 * be the types of data members or local variables in code
 * blocks, even if those code blocks are never executed.
 * <p>
 * To put the APIs that might not be present outside of the
 * transitive closure of statically referenced classes, we
 * introduce this interface, and dynamically load the single
 * class that implements it using Class.forName().  The class
 * that implements this interface can contain static references
 * to APIs that might not be present, and it can contain static
 * references to classes that reference such classes.
 *  <p>
 *  This interface can be thought of as a Facade.
 *  See the Facade pattern on page 185 of GoF
 *  ("Design Patterns" by Gamma et al, ISBN
 *  0-201-63361-2).
 **/
public interface RCFeatures {

   /**
    * Set up the return channel, and get ready for the Xlet
    * to run.
    **/
   public void init();

   ... Here, there are declarations of all of the features
       of the Xlet that use the return channel.  They're
   probably quite high-level, such as "submit survey
   results."  There might be other methods, to manage
   the return channel during Xlet state transitions,
   such as to the paused or destroyed state  ...

}


/**
 * This class contains implementations of all features of the
 * xlet that depend on the presence of a return channel.  It
 * can safely reference classes that are not present in the
 * EB profile, and it can safely reference classes that
 * reference such classes.
 **/

public class RCFeaturesImpl implements RCFeatures {

   private org.dvb.net.rc.RCInterfaceManager mgr;

   public void init() {
   mgr = org.dvb.net.rc.RCInterfaceManager.getInstance();
   ...  use mgr to set up the return channel, as
                necessary ...
   }

   ...  Here, there are implementations of the features
        declared above ...
```

# W.4 Example of lightweight trigger API

```
public class example implements StreamEventListener {

  public void example() {
    //  s is a ServiceDomain that carries a lightweight trigger binding.  An
    // interoperable GEM application might get the locator to the service domain
    // from a simple text file used for configuration to a certain platform.

    ServiceDomain s = ....

    DSMCCObject topLevelObject = s.getMountPount();
    DSMCCObject triggerObject = new DSMCCObject( topLevelObject,
        "lightweight_triggers");

    //  This assumes a lightweight trigger binding with names "a" to "z" hard-coded.  If
    //  the name of the trigger might vary based on platform or signalling, this name could
    // be parameterized through a text file, as with the ServiceDomain's locator.

    DSMCCStreamEvent eventSource = new DSMCCStream( triggerObject );
    eventSource.subscribe("a", this);
  }
  public void receiveStreamEvent( StreamEvent e )
  {
      //....
  }
}
```

# W.5 Example of media stream synchronization API

```
import java.io.IOException;
import javax.media.Controller;
import javax.media.Manager;
import javax.media.NoPlayerException;
import javax.media.Player;
import javax.media.Time;
import javax.tv.service.selection.InsufficientResourcesException;

import org.davic.media.MediaLocator;
import org.davic.net.InvalidLocatorException;
import org.davic.net.dvb.DvbLocator;
import org.dvb.media.IncompatibleSynchronizableMediaTypeException;
import org.dvb.media.MasterSlaveSyncLinkageControl;
import org.dvb.media.SyncControl;
import org.dvb.media.SyncStateChangedEvent;
import org.dvb.media.SyncStateChangedEventListener;
import org.havi.ui.HPermissionDeniedException;
import org.havi.ui.HScreen;
import org.havi.ui.HVideoDevice;


public class SyncControlExample implements SyncStateChangedEventListener {

     private Player broadcastPlayer = null;
     private Player CCPlayer = null;
     private org.davic.media.MediaLocator CCMediaLocator;
     private DvbLocator CCLocator;
     private SyncControl bpSC = null;
     private SyncControl cpSC = null;
     private MasterSlaveSyncLinkageControl aMSSLC = null;

     public SyncControlExample() {

      // Create Broadcast Player
      HScreen screen = HScreen.getDefaultHScreen();
      HVideoDevice vd = screen.getHVideoDevices()[0];

      try {
         broadcastPlayer = (Player)vd.getVideoController();
      } catch (SecurityException e) {
         e.printStackTrace();
      } catch (HPermissionDeniedException e) {
         e.printStackTrace();
      }
      // Obtain SyncControl for Broadcast Player to be a slave Player
      bpSC = (SyncControl)broadcastPlayer.getControl("org.dvb.media.SyncControl");

      // Obtain MasterSlaveSyncLinkageControl to be a master Player
      aMSSLC =
```

```
(MasterSlaveSyncLinkageControl)broadcastPlayer.getControl("org.dvb.media.MasterSlaveSyncLinkageContr
ol");

      // Add listener for event related to synchronization
      aMSSLC.addSyncStateChangedEventListener(this);

      // Instantiate closed caption Player
      try {
         CCLocator = new DvbLocator("http://service/CCcomponent");
      } catch (InvalidLocatorException e2) {
         e2.printStackTrace();
      }
      CCMediaLocator = new MediaLocator(CCLocator);
      try {
         CCPlayer = Manager.createPlayer(CCMediaLocator);
      } catch (NoPlayerException e) {
         e.printStackTrace();
      } catch (IOException e) {
         e.printStackTrace();
      }

      // Obtain SyncControl to be a slave Player
      cpSC = (SyncControl)CCPlayer.getControl("org.dvb.media.SyncControl");

      if (CCPlayer != null) CCPlayer.start();

      Time tolerance = new Time(0.03); // Tolerance = 30ms
      Time offset = new Time(3.3); // Offset = 3.3s

      cpSC.setTolerance(tolerance);

      // Add closed caption player as a slave Player
      // Synchronization process starts automatically.
      try {
         aMSSLC.addSlave(CCPlayer, offset);
      } catch (InsufficientResourcesException e) {
         e.printStackTrace();
      } catch (IllegalArgumentException e) {
         // Checking reasons
         // Followings are sample to check whether CCPlayer is already a slave Player of other
master
         if (cpSC.getMaster() != null) {
            aMSSLC.removeSlave(CCPlayer);
         }
         // Followings are sample to check whether CCPlayer is a master
         // Instance of SyncControl can be created at most one.
         if
(((MasterSlaveSyncLinkageControl)CCPlayer.getControl("org.dvb.media.MasterSlaveSyncLinkageControl"))
.getSlaves().length != 0) {
            aMSSLC.removeSlave(CCPlayer);
         }
         e.printStackTrace();
      } catch (IncompatibleSynchronizableMediaTypeException e) {
         e.printStackTrace();
      } catch (IncompatibleTimeBaseException e) {
         e.printStackTrace();
      }

     }

     /*
      * Methods as an SyncStateChangedEvent event listener
      */

     public void inSync(SyncStateChangedEvent e) {

     }

     public void outOfSync(SyncStateChangedEvent e) {
      System.out.println("Now Resyncing...");
      Controller ctrl = e.getSourceController();
      int status = e.getSyncStatus();
      switch(status) {
      case MasterSlaveSyncLinkageControl.OUT_OF_SYNC_MASTER_CLK : {
         do_something;
         break;
      }
      case MasterSlaveSyncLinkageControl.OUT_OF_SYNC_SLAVE_CLK : {
         do_another_thing;
         break;
      }
       ....
      }

     }

     public void unSynchronizable(SyncStateChangedEvent e) {
      MasterSlaveSyncLinkageControl aSC = (MasterSlaveSyncLinkageControl)e.getSource();
      Controller aCtrl = e.getSourceController();
```

```
        aSC.removeSlave(aCtrl);  // Remove a slave Player that 'Give-up' to synchronize
      }
}
```

# Annex X (normative): Test support

NOTE: The introduction and the figure are specific to broadcast systems. It is not applicable to the packaged media target. Many of the descriptive elements in this annex are based on a broadcast system; the details for both broadcast and packaged media implementations may differ.

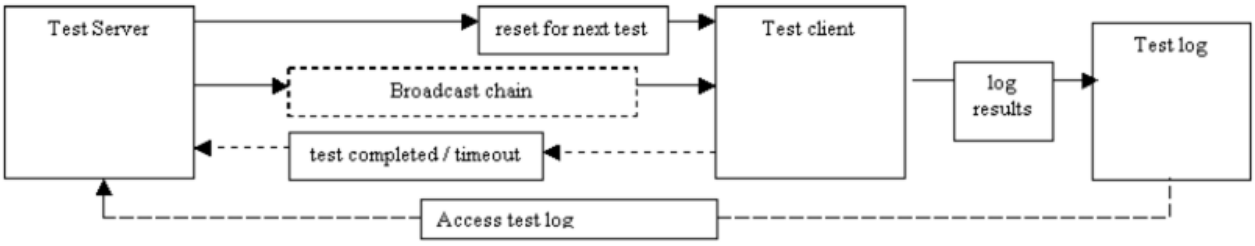
# Package

# Broadcast model

Description

In a broadcast-based conformance system, there are effectively three main entities involved in an automated test process:

1. The test server that is used to hold and initiate all of the tests.
2. The test client which runs the tests and logs the results.
3. The broadcast chain that is used to transfer applications and application data from the server to the client.




The communication order is as follows:

1. The test-server uses the "reset for next test" mechanism to set the test client into a known default state, ready to receive the test-application.
2. The test-server uses the "broadcast chain" mechanism to supply the test-application to the test client and to signal that the test-application should be executed.
3. The test-client runs the test-application.
4. The test-application either:
   - Finishes within a given timelimit, the result of the test is known and shall be considered to be the value reported by the test application for the purposes of compliance.
   - Optionally, the test-client may signal to the test-server that the test-application has finished executing and that the test-client is ready to be reset in order to receive the next test-application.
   - Fails to finish the test-application within a given timeout, the result of the test is unknown and shall be treated as a failure for the purposes of compliance. The test-server may treat the test-client as ready to be reset in order to receive the next application.

[Successive tests are then repeated from stage 1.]

### "Reset for next test"

The "reset for next test" path is used by the test server to reset the test client to receive the next test. The reset for next test API is considered to be a private implementation issue between the test-server and test-client and therefore has no public Java API implications. Note that this "communication" needs to take place prior to any application being executed. Note that the precise manner of the reset mechanism is intentionally not specified - in the worst case, this may involve "power cycling" the test client.

### Test log

Communication from the test client to the test log is considered as write-only access. Hence, results from successive tests cannot overwrite results from previous ones. Multiple (intermediate) results may be sent to the test log for any given test. It is recommended that all communication to the test log is synchronous.

See the DVBTest.log method for details of the proposed API and implementation issues.

### "Test completed"

The "test completed" path is used by the test client to indicate to the test server that it has completed the previous test and is now able to accept a subsequent one. Note that this communication path is an optimization, since direct communication from the client to the server is not actually required, e.g. the server might simply "time-out" the client, and then perform a "reset for next test" action. However, this optimization may be important when large numbers of tests are being performed on a "capable" platform, since e.g. if a 30 second timeout is applied for 1 000 test cases which typically run within say 6 seconds, then the timeout implies a typical running time of 500 minutes, i.e. ~4,5 hours- rather than 100 minutes, i.e. ~1,5 hours.

See the DVBTest.terminate method for details of the proposed API and implementation issues.

### Access test log

The mechanism by which the test-log is accessed is not considered in the present document, this is a private mechanism, which might include reading a file from flash/RAM. Similarly, the mechanism by which results are recovered from the test log is not considered in the present document, e.g. the test-log may actually reside on the test server, e.g. as in the case that results are transmitted over an IP connection.

| Class Summary | |
|---|---|
| **Classes** | |
| DVBTest | The DVBTest class allows test applications to log messages during their execution and to indicate their termination condition in a platform independent manner. |

# org.dvb.test

# DVBTest

### Declaration

```
public class DVBTest

java.lang.Object
  |
  +--org.dvb.test.DVBTest
```

### Description

The DVBTest class allows test applications to log messages during their execution and to indicate their termination condition in a platform independent manner.

A number of constants are defined in the DVBTest class and are reserved as follows:

- Zero and negative values defined within the class are reserved by DVB.
- Positive return values are available for test application specific return values, which must be defined within the procedure for executing the test application as to their precise meaning as regards conformance.

### Fields

#### FAIL

```
public static final int FAIL
```

The application executed and terminated unsuccessfully and has therefore operated in a non-conformant manner.

#### HUMAN_INTERVENTION

```
public static final int HUMAN_INTERVENTION
```

The application is unable to determine whether it has operated conformantly and therefore requires some human intervention to determine whether conformance has been achieved. Until the application has been checked the result of the application should be considered as non-conformant.

It is envisaged that tests returning this value may be those requiring evaluation of presented content, such as graphics, etc. Such presentation may require (subjective) human evaluation.

#### OPTION_UNSUPPORTED

```
public static final int OPTION_UNSUPPORTED
```

The platform does not contain the option under test and therefore the test is inapplicable, the test result should not be considered when determining the status of the platform's conformance.

#### PASS

```
public static final int PASS
```

The application executed and terminated successfully and has therefore operated in a conformant manner.

#### UNRESOLVED

```
public static final int UNRESOLVED
```

A setup stage necessary to execute the application failed, and hence the result of the application is unknown and therefore should be considered to have operated in a non-conformant manner.

#### UNTESTED

```
public static final int UNTESTED
```

The application ran successfully, but the particular test was unable to execute. Hence the result of is unknown, and may require human evaluation to determine conformance.

For example, an out of disk space test may not execute within a fixed number of iterations (within a practical amount of time) for devices with large capacity storage, etc.

### Methods

#### log(String, int)

```
public static void log(java.lang.String id, int no)
throws IOException
```

This method has the same behaviour, implementation options and restrictions as log(String, String) - except that. it allows an integer value to be logged, rather than a String, which may prove a useful option for automating tests.

#### Parameters:

`id` - a string identifying the application (thread) that is logging the test result.

`no` - the integer value that the application wishes to be logged.

Throws:

`java.io.IOException` - under the same conditions as log(String, String).

### log(String, String)

```
public static void log(java.lang.String id, java.lang.String message)
throws IOException
```

This synchronous, blocking, method logs a result (intermediate result) of a test application using write-only access. The method takes both an identifier string, e.g. "Test number 1" and a message to output, e.g. "Now invoking the xletPause method...". The application is not required to open a file or network connection, per se, and the log() method is always available for writing (in principle).

The precise format of the logged message is left deliberately unspecified, implementers may choose to output compressed messages, XML documents, or other formats of their choice (obviously provided that the original information can be recovered). It is an implementation option to include additional information with each logging message, e.g. including:

- version of the specification being implemented;
- compiler version and options;
- build-version;
- timestamp;
- date;
- debug info.

Messages sent using this method should "atomic", i.e. that they are not interleaved with other messages sent using the methods defined in the DVBTest class.

#### Implementation

The precise mechanism(s) by which the this method may be implemented are intentionally unspecified, implementation options might include:

- logging the message to a local file system;
- logging the message to a mounted remote file system;
- logging the message to a RAM disk, etc.;
- logging the message via an RS-232 (or other serial) connection;
- logging the message to a remote host via some IP / UDP based mechanism, e.g. using a socket-based connection.

Note that the implementation of the log method may use the same or a different mechanism to that used by the terminate method.

The log method does not require any explicit initialisation on the part of the application under test. For example if messages are being stored to a file system, then the application is not required to mount / open any storage file. Similarly, if the messages are being logged via a network connection, then the application is not required to open a connection to the storage host, etc. In principle, the mechanism should always be available to accept messages.

If this method is implemented on top of some buffering mechanism, it is strongly recommended that the buffer be flushed for each occurrence of a message being logged.

#### Security and implementation options

There is no Java security mechanism that is used to secure the log method.

Note that even if the log method is based on a particular implementation option, it shall be able to operate in spite of that particular implementation option itself being subject to security checks. For example, a log method implemented using the java.net.Socket class shall always be able to log a message from a test-application, even if the test-application is unable to directly access the java.net.Socket class due to security restrictions, etc.

It is an allowed implementation option to require that the test-client be put into some particular "test-mode" before any test-results are logged. This mechanism is required to reduce any inadvertent interaction due to downloaded applications accessing the test methods.

#### Authoring guidelines

The log method is not intended to be accessed by downloaded applications directly, it is purely intended for the use of conformance test applications. Authors of downloaded applications should not call this method, since there may be interactions between this method and normal in-field operation of the test-client (GEM platform).

It is an allowed implementation option to require that the test-client be put into some particular "test-mode" before any test-results are logged. This mechanism is required to reduce any inadvertent interaction due to downloaded applications accessing the test methods.

It is an allowed implementation option to have a number of "test-modes" that are appropriate to different elements being conformance tested, for example, it is a valid implementation for a test-client to have a test-mode where results are stored via a serial port, and a separate test-mode where results are stored via a RAM disk. It is allowable for a conformance test to be performed with the test-client in some specific test- mode, e.g. a java.net test (using a serial modem) might have its test results logged to a RAM disk, to avoid interaction between test-log messages and the serial protocol.

The mechanism by which a test-client is put into a given test mode is intentionally left unspecified.

#### Relationship to java.io

It is an implementation option to map the implementation of this method onto corresponding write method(s) of appropriate java.io classes. These classes may in turn be obtained, e.g. from java.net Socket classes, etc.

#### Parameters:

`id` - a string identifying the application (thread) that is logging the test result.

`message` - the message that the application wishes to be logged.

#### Throws:

`java.io.IOException` - if there is any problem in providing synchronous logging to an application. This IOException may be due to failure to write to a file system, inability to access a remote socket, etc. the precise causes are deliberately unspecified and are implementation dependent.

### prompt(String, int, String)

```
public static void prompt(java.lang.String id, int controlCode, java.lang.String message)
throws IOException
```

This is a method is used to "approximately" synchronise a test-client and test-server, the method blocks until the test-server positively or negatively acknowledges the particular message. The intended use of this method is to remove critical timing issues from conformance tests, e.g. a conformance test to ensure that an Xlet responds to a change in broadcast signalling must first ensure that the Xlet is in a state where it is able to respond to such signalling - since the time taken for an Xlet to achieve such a state is reliant on aspects outside of the scope of the conformance test itself (delivery bit rate, hardware and CPU capabilities of the test-client, etc.).

Messages sent using this method should "atomic", i.e. that they are not interleaved with other messages sent using the methods defined in the DVBTest class.

### Implementation

The precise mechanism(s) by which the this method may be implemented are intentionally unspecified. Implementation options for sending the prompt might include:

- logging the controlCode via an RS-232 (or other serial) connection;
- logging the controlCode to a remote host via some IP / UDP based mechanism, e.g. using a socket-based connection;
- displaying the message on-screen for a (human) test operator, e.g. for systems not implementing a return channel capability.

Implementation options for receiving the acknowledgement might include:

- acknowledgement via an RS-232 (or other serial) connection;
- acknowledgement from a remote host via some IP / UDP based mechanism, e.g. using a socket-based connection;
- a (human) test operator manually acknowledging the message, e.g. for systems not implementing a return channel capability.

#### Parameters:

`id` - a string identifying the application (thread) that is sending the prompt.

`controlCode` - an integer value (unique within a given Xlet) intended for use by some automated test process (corresponding to the readable message).

`message` - a message (unique within a given Xlet) intended to be readable by a (human) test operator (corresponding to the automated controlCode).

#### Throws:

`java.io.IOException` - If there is any problem in receiving a positive acknowledgement from the test-server, then an this shall be thrown. This may be due to a negative acknowledgement from the test-server, or due to other communication based causes - which are deliberately left unspecified.

### terminate(String, int)

```
public static void terminate(java.lang.String id, int terminationCondition)
throws IOException
```

This synchronous, blocking, method logs the termination condition of a test application using write-only access. The method takes both an identifier string, e.g. "Test number 1" and a integer value to output, e.g. org.dvb.test.DVBTest.PASS. In addition to logging the termination condition of the test, invoking this method also indicates that the test application has terminated its operation. Note that termination of operation does not necessarily correspond to the application being in any particular lifecycle state (as defined in the "Application Model" clause of the GEM specification). The application is not required to open a file or network connection, per se, and the terminate() method is always available for writing (in principle).

The precise format of the termination message is left deliberately unspecified, implementers may choose to output compressed messages, XML documents, or other formats of their choice (obviously provided that the original information can be recovered). It is an implementation option to include additional information with each termination message, e.g. including:

- version of the specification being implemented;
- compiler version and options;
- build-version;
- timestamp;
- date;

- debug info.

On test-clients whose implementation of the terminate() method supports external communication to its test-server, implementations of this method may optionally indicate to the test-server that the test-client can be reset by its test-server so that another test may be initiated. The precise mechanism by which this communication takes place is not specified it may be via a IP / socket, serial port, etc.

In the case of an test-client that does not support communication to its test-server, or in the case of an unsuccessful (hanging) test, or inability of this method to return (without throwing an exception) the test-server must be prepared to "time out" the application running on the test-client and then reset the test-client.

### Implementation

The precise mechanism(s) by which the this method may be implemented are intentionally unspecified, implementation options might include:

- storing the termination condition to a local file system;
- storing the termination condition to a mounted remote file system;
- storing the termination condition to a RAM disk, etc.;
- storing the termination condition via an RS-232 (or other serial) connection;
- storing the termination condition to a remote host via some IP / UDP based mechanism, e.g. using a socket-based connection.

Messages sent using this method should "atomic", i.e. that they are not interleaved with other messages sent using the methods defined in the DVBTest class.

Note that the implementation of the terminate method may use the same or a different mechanism to that used by the log method.

The terminate method does not require any explicit initialisation on the part of the application under test. For example if termination conditions are being stored to a file system, then the application is not required to mount / open any storage file. Similarly, if the results are being logged via a network connection, then the application is not required to open a connection to the storage host, etc. In principle, the mechanism should always be available to accept termination messages.

If this method is implemented on top of some buffering mechanism, it is strongly recommended that the buffer be flushed for each occurrence of a message being logged.

### Security and implementation options

There is no Java security mechanism that is used to secure the terminate method.

Note that even if the terminate methods is based on a particular implementation option, it shall be able to operate in spite of that particular implementation option itself being subject to security checks. For example, a terminate method implemented using the java.net.Socket class shall always be able to log the termination condition of a test-application, even if the test-application is unable to directly access the java.net.Socket class due to security restrictions, etc.

It is an allowed implementation option to require that the test-client be put into some particular "test-mode" before any test-results are logged. This mechanism is required to reduce any inadvertent interaction due to downloaded applications accessing the test methods.

### Authoring guidelines

The terminate method is not intended to be accessed by downloaded applications directly, it is purely intended for the use of conformance test applications. Authors of downloaded applications should not call this method, since there may be interactions between this method and normal in-field operation of the test-client (GEM platform).

It is an allowed implementation option to require that the test-client be put into some particular "test-mode" before any test-results are logged. This mechanism is required to reduce any inadvertent interaction due to downloaded applications accessing the test methods.

It is an allowed implementation option to have a number of "test-modes" that are appropriate to different elements being conformance tested, for example, it is a valid implementation for a test-client to have a test-mode where results are stored via a serial port, and a separate test-mode where results are stored via a RAM disk. It is allowable for a conformance test to be performed with the test-client in some specific test- mode, e.g. a java.net test (using a serial modem) might have its test results logged to a RAM disk, to avoid interaction between test-log messages and the serial protocol.

The mechanism by which a test-client is put into a given test mode is intentionally left unspecified.

### Relationship to java.io

It is an implementation option to map the implementation of this method onto corresponding write method(s) of appropriate java.io classes. These classes may in turn be obtained, e.g. from java.net Socket classes, etc.

### Parameters:

`id` - a string identifying the application (thread) that is terminating the test.

`terminationCondition` - the termination condition of the test application.

### Throws:

`java.io.IOException` - thrown if there is any problem in terminating an application. This may be due to failure to write to a file system, inability to access a remote socket, etc. the precise causes are deliberately unspecified.

# Annex Y (normative): Inter-application and Inter-Xlet communication API

GEM accomplishes inter-application and inter-Xlet communication by means of the org.dvb.io.ixc API with the following modifications.

The class description of `org.dvb.io.IxcRegistry` contains the following clause:

NOTE 1: organisation ID values between 0x80000000 and 0xffffffff are known to create problems in some implementations and their use is strongly discouraged.

In the description of the method `lookup(XletContext, String)` the following note is added:

NOTE 2: organisation ID values between 0x80000000 and 0xffffffff are known to create problems in some implementations and their use is strongly discouraged.

# Package

# org.dvb.io.ixc

### Description

Provides support for inter-application communication.

| Class Summary | |
|---|---|
| **Classes** | |
| IxcRegistry | This is the bootstrap mechanism for obtaining references to remote objects residing in other Xlets executing on the same GEM terminal, using a URL-like syntax. |

# org.dvb.io.ixc

# IxcRegistry

### Declaration

```
public class IxcRegistry

java.lang.Object
  |
  +--org.dvb.io.ixc.IxcRegistry
```

### Description

This is the bootstrap mechanism for obtaining references to remote objects residing in other Xlets executing on the same GEM terminal, using a URL-like syntax. The identification of a remote object is given using a syntax indicating the organisation ID and application ID:

/organisation_id/application_id/name
organisation_id = the organisation ID of the Xlet, as signalled in the application_identifier record, defined in the GEM specification.
application_id = the application ID of the Xlet, as signalled in the application_identifier record, defined in the GEM specification.
name = the name under which the remote object was exported.

The organisation ID and the application ID shall each be encoded as a hexadecimal string, as would be accepted by java.lang.Integer.parseInt(String s, 16).

When RMI is used to communicate over a network, stubs generated by a tool like rmic are often required. This is not necessary for inter-xlet communication initiated with IxcRegistry. If such stubs are present, they shall be ignored.

Similarly, network RMI objects often extend the class server.RemoteObject, in order to get appropriate implementations for Object.hashCode(), Object.equals(), and Object.toString(). Overriding Object's implementation of these methods in this way is not necessary for inter-xlet communication initiated with IxcRegistry, although it is not harmful. Note that the class server.RemoteObject is not required in all GEM profiles.

### Fields

#### GLOBAL

```
public static final int GLOBAL
```

Definition of the scope for bind or rebind - exported object is visible to any Xlet running within the same GEM terminal subject to requirements of the security model.

Since:

MHP 1.1.2.

#### PAGE

```
public static final int PAGE
```

Definition of the scope for bind or rebind - exported object is only visible to Xlets within the same DVB-HTML application. Note that an embedded Xlet with a different app ID than its enclosing HTML page is still considered to be the same application as that which contains the enclosing page.

Since:

MHP 1.1.2.

#### SERVICE

```
public static final int SERVICE
```

Definition of the scope for bind or rebind - exported object is only visible to Xlets running within the same service context

Since:

MHP 1.1.2.

### Methods

#### bind(XletContext, String, Remote)

```
public static void bind(javax.tv.xlet.XletContext xc, java.lang.String name, java.rmi.Remote obj)
throws AlreadyBoundException
```

Exports an object under a given name in the namespace of an Xlet. The name can be any valid non-null String. No hierarchical namespace exists, e.g. the names "foo" and "bar/../foo" are distinct. If the exporting xlet has been destroyed, this method may fail silently.

The object shall be made visible to other applications running in the same service context. A call to bind(xc, name, obj) is thus equivalent to a call to bind(xc, name, obj, SERVICE).

Parameters:

`xc` - The context of the Xlet exporting the object.

`name` - The name identifying the object.

`obj` - The object being exported.

Throws:

`java.rmi.AlreadyBoundException` - if this Xlet has previously exported an object under the given name.

`java.lang.NullPointerException` - if xc, name or obj is null.

### bind(XletContext, String, Remote, int)

```
public static void bind(javax.tv.xlet.XletContext xc, java.lang.String name, java.rmi.Remote obj,
int scope)
throws AlreadyBoundException
```

Exports an object under a given name in the namespace of an Xlet. The name can be any valid non-null String. No hierarchical namespace exists, e.g. the names "foo" and "bar/../foo" are distinct. If the exporting xlet has been destroyed, this method may fail silently.

Parameters:

`xc` - The context of the Xlet exporting the object.

`name` - The name identifying the object.

`obj` - The object being exported.

`scope` - The scope to which the object is to be exported.

Throws:

`java.rmi.AlreadyBoundException` - if this Xlet has previously exported an object under the given name.

`java.lang.NullPointerException` - if xc, name or obj is null.

Since:

MHP 1.1.

### list(XletContext)

```
public static java.lang.String[] list(javax.tv.xlet.XletContext xc)
```

Returns an array of string path objects available in the registry. The array contains a snapshot of the names present in the registry that the current Xlet would be allowed to import using IxcRegistry.lookup.

Parameters:

`xc` - The context of the current Xlet.

Returns:

A non-null array of strings containing a snapshot of the path names of all objects available to the caller in this registry.

See Also:

`lookup(XletContext, String)`

### lookup(XletContext, String)

```
public static java.rmi.Remote lookup(javax.tv.xlet.XletContext xc, java.lang.String path)
throws NotBoundException, RemoteException
```

Returns a remote object previously exported by an Xlet that has not been destroyed. The identification of a remote object is given using a syntax indicating the organisation ID and application ID:

/organisation_id/application_id/name
organisation_id = the organisation ID of the Xlet, as signalled in the application_identifier record.
application_id = the application ID of the Xlet, as signalled in the application_identifier record.
name = the name under which the remote object was exported.

The organisation ID and the application ID shall each be encoded as a hexadecimal string, as would be accepted by java.lang.Integer.parseInt(String s, 16). If the caller is not authorized to import a given object due to the security policy, then this API will behave as though the object had not been exported, that is, a NotBoundException shall be thrown.

Parameters:

`xc` - The context of the current Xlet (that is, the Xlet importing the object).

`path` - A file pathname-like string identifying the Xlet and the name of the object to be imported.

Returns:

A remote object.

Throws:

`java.rmi.NotBoundException` - If the path is not currently bound.

`java.rmi.RemoteException` - If a remote stub class cannot be generated for the object being imported.

`java.lang.IllegalArgumentException` - If the path is not formatted in the syntax given above.

`java.lang.NullPointerException` - if path is null.

### rebind(XletContext, String, Remote)

```
public static void rebind(javax.tv.xlet.XletContext xc, java.lang.String name, java.rmi.Remote obj)
```

Rebind the name to a new object in the context of an Xlet; replaces any existing binding. The name can be any valid non-null String. No hierarchical namespace exists, e.g. the names "foo" and "bar/../foo" are distinct. If the exporting xlet has been destroyed, this method may fail silently.

The object shall be made visible to other applications running in the same service context. A call to rebind(xc, name, obj) is thus equivalent to a call to rebind(xc, name, obj, SERVICE).

Parameters:

`xc` - The context of the Xlet that exported the object.

`name` - The name identifying the object.

`obj` - The object being exported.

Throws:

`java.lang.NullPointerException` - if xc, name or obj is null.

### rebind(XletContext, String, Remote, int)

```
public static void rebind(javax.tv.xlet.XletContext xc, java.lang.String name, java.rmi.Remote obj,
int scope)
```

Rebind the name to a new object in the context of an Xlet; replaces any existing binding. The name can be any valid non-null String. No hierarchical namespace exists, e.g. the names "foo" and "bar/../foo" are distinct. If the exporting xlet has been destroyed, this method may fail silently.

Narrowing the scope of the binding (e.g. from GLOBAL to SERVICE) shall have the same effect as a call to unbind for any applications which had references to that object and which were in scope but which are now out of scope.

Parameters:

`xc` - The context of the Xlet that exported the object.

`name` - The name identifying the object.

`obj` - The object being exported.

`scope` - The scope to which the object is to be exported.

Throws:

`java.lang.NullPointerException` - if xc, name or obj is null.

Since:

MHP 1.1.

### unbind(XletContext, String)

```
public static void unbind(javax.tv.xlet.XletContext xc, java.lang.String name)
throws NotBoundException
```

Unbind the name.

Parameters:

`xc` - The context of the Xlet that exported the object to be unbound.

`name` - The name identifying the object.

Throws:

`java.rmi.NotBoundException` - if this is not currently any object exported by this Xlet under the given name.

`java.lang.NullPointerException` - if xc or name is null.

# Annex Z (informative): Services, service contexts and applications in a GEM environment

This informative clause includes references to some signalling details not required by GEM. Where this is the case, it is to be read as an example of one possible way of fulfilling the abstract requirements placed on terminal signalling by GEM.

## Z.1 Introduction

GEM describes the concepts that link the various parts of a GEM execution environment so that it can display a complete GEM service, including media and applications. This is really an overview to the GEM application lifecycle model, but does include some additional information.

We assume some familiarity with GEM and the Java TV specification.

## Z.2 Basic concepts

The unit for the presentation and execution of content in the GEM specification is the service. A service in GEM represents a group of pieces of content which are intended to be presented together to the end-user. In this version of the specification, the service is the contents of a service, including audio/video streams, data streams and all the service information, applications and application signalling that is being broadcast. The current service will largely be responsible for determining what media and applications are presented to the user.

Every service that gets presented by a GEM platform is presented within a service context. These form one of the foundations for the runtime environment and the execution model. A service context is an "environment" in which a service gets presented - it defines the boundaries of the service (letting the platform and applications identify which of the pieces of content that are being presented make up a given service). It also enables that service to be addressed and controlled as a single entity. A DVB-J application can call the select method on a service context (represented by javax.tv.service.selection.ServiceContext) and the platform will stop presenting all of the content that makes up the current service being presented by that service context and start presenting the content that makes up the new service. In this case, "content" may include one or more applications.

A service context has some major differences from a DVB-J Xlet context. It is not necessary to have one service context for every possible service that can or will get presented - one service context is needed for every service that can be presented simultaneously, but that is all. Also, a service context is not destroyed when the service within it is stopped, unlike the Xlet context for a DVB-J application. The service context exists until an application or the platform explicitly destroys it. In normal operational mode, the built-in navigator or EPG for a GEM system will create one single service context when it starts and never destroy that. GEM applications will run in that service context.

## Z.3 Presenting a service in GEM

From the GEM point of view, the content of a service can be one of two types - media or applications. Media is the simplest case and is described first.

### Z.3.1 Presenting the media components of a service

If there are several different streams of media that may get presented (such as several different video streams) then the platform uses a variety of methods to tell which streams should be used. These include user preferences and platform defaults, but will also include using service information do determine which streams get presented to the user.

For a DVB-J application, all real-time media components sharing the same clock are presented by the same JMF player. These players are directly linked to the service context, and the service context can be queried to find out which JMF Players are linked with it. It is also possible for applications to create JMF Player objects directly without linkage to the applications service context.

## Z.3.2 Presenting the application components of a service

Applications are handled in a slightly different way. The lifecycle of all applications in a GEM environment is controlled by an application manager, a software entity that forms part of the GEM runtime environment. It takes its instructions on which applications to start and stop from the user, but also from information that is included in the GEM broadcast (called an "Application Description", can logically be considered an extension to MPEG's PMT) and the free resources in the platform.

The information carried in the Application Description not only says which applications are available, but also provides some instructions to the application manager about whether an application should be started automatically or whether an application should be killed automatically. The application manager monitors this information for changes, and creates, starts or kills applications as appropriate.

Every DVB-J application executes within an Xlet context. This is a similar concept to the applet context that a Java applet executes in, and it provides the Xlet with a link to its environment, both for accessing system properties and for telling the environment that the Xlet has changed its own state.

Since every Xlet executes as part of a service, there is also a link between the Xlet and its associated service. This link is the class javax.tv.service.selection.ServiceContextFactory. Using methods on this class, an Xlet can lookup its service context from its xlet context. From the service context, an Xlet can discover which service it is current running as part of. It can also register for events to be told when the service being presented in its service context changes.

The application manager maintains a list of all the applications in a system, so that it knows which ones are currently executing. It also knows (directly or indirectly) which applications are associated with which service context, so that if the service being presented in that service context changes due to a new service being selected, it can kill the applications that are not signalled in the new service. When a service is selected in a service context, the following steps happen in approximately this order:

a) The platform examines the MPEG PMT for the service that has been selected (tuning first if necessary) and works out which media streams are to be presented.

b) Any media streams currently playing are stopped and any new media streams that need to be presented are started. Any JMF players presenting the old content are stopped and destroyed, and players for the new content are created and started if necessary.

c) The platform monitors the application information table for the new service to find out which applications should be running. Any applications that are currently running but which are not signalled in the application information table of the new service (or which are signalled as killed) are killed and the Xlet contexts of DVB-J applications associated with the old service are destroyed.

d) For any applications which are signalled as autostart and are not currently running, the following steps are taken:

   - The platform attaches to the object carousel signalled in the application information table.
   - The platform attempts to load the main application file as signalled in the application information table.
   - For a DVB-J application, the platform creates an instance of the main class using the default constructor, creates an XletContext object for the application and calls the Xlet.initXlet() method on the newly loaded class, passing the XletContext object as a parameter. Once this call is complete, the platform calls the Xlet.startXlet() method.

If several applications are signalled as autostart, the platform will load and start every one in the same way. Each DVB-J application will have a different Xlet context and will execute independently, although they are all associated with the same service context.

# Z.4 Service Context Object Design

## Z.4.1 Overview

This clause explains the underlying design behind service contexts, resources such as video and audio decoders, and JMF players.

## Z.4.2 Single-Application Model

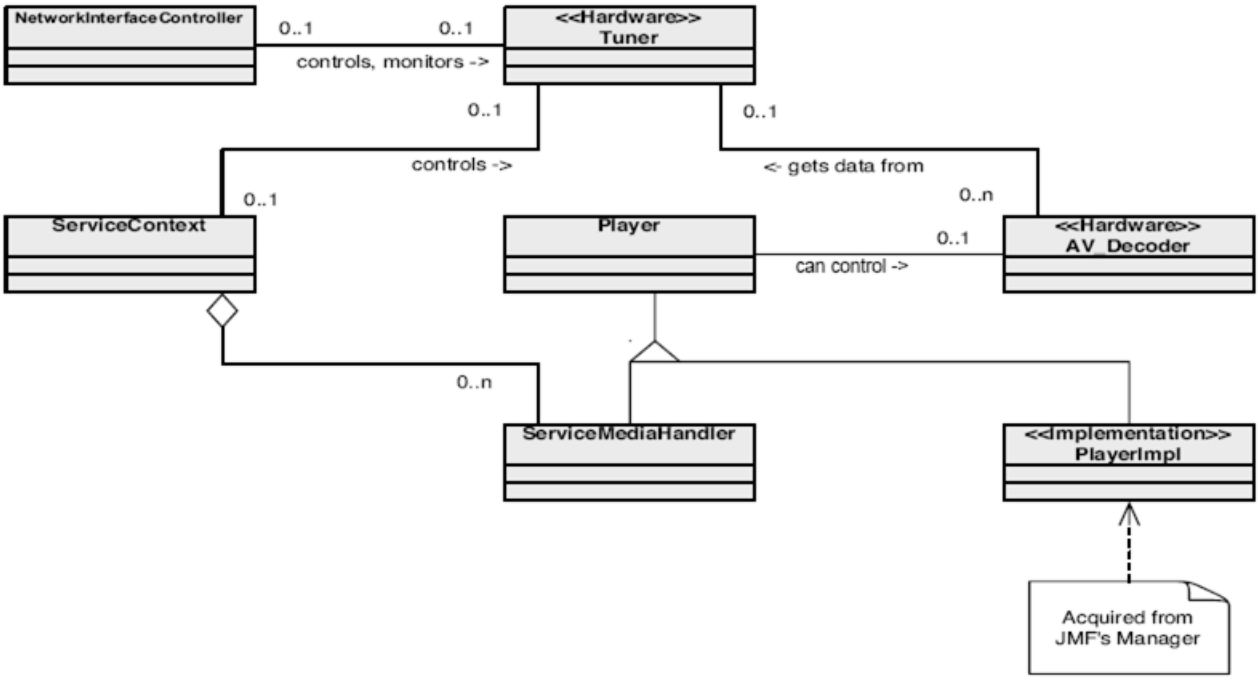
For simplicity, we will first consider the operation of this API when a single application is executing.

A service context represents the context in which services execute. There are different kinds of services, including application-only services, stand-alone stored services, services including audio/video content, and recorded services. All services may have applications, but some kinds of services cannot include audio/video content. Some, but not all services are played from a tuner. Thus, a service context:

- Has the ability to present applications.
- May control resources needed to present broadcast or recorded services, such as a tuner, a video decoder, and audio decoder.
- Is the entity an application accesses to select a new service.

Applications may select a different audio/video stream without selecting a new service. For example, an EPG that runs within an EPG service might wish to display the picture and play the audio associated with a different service, without actually selecting that service. This is not done by manipulating a service context; rather, it is done by using a JMF Player (which is not a `ServiceMediaHandler`), and if needed the tuning API. This is explained in greater detail below.

A service context is exposed to applications using the `javax.tv.service.ServiceContext` interface. The model underlying the `ServiceContext` interface is pictured in the following UML class diagram. The types shown without stereotypes indicating otherwise (e.g. "<<Hardware>>") are part of the public API. They are taken from different packages, but are pictured without the full package name for readability.




All `ServiceContext` instances act as a context to present services. Services may include applications, audio content, or video content. When a service containing A/V content is selected in a `ServiceContext`, the appropriate `ServiceMediaHandler` will attempt to acquire the underlying A/V decoder in hardware. If an application wishes to present a different A/V stream without selecting a new service (and thus, without potentially selecting a new set of applications), it cannot do so using the `ServiceMediaHandler` associated with the `ServiceContext`, because `ServiceMediaHandler` is restricted to only presenting streams within the service. Instead, the application may acquire a `Player` from the JMF `Manager`. This JMF player can be set to present a different A/V stream, but it is not allowed to tune. If that is required, the application acquires a DAVIC `NetworkController` to manually tune; on a GEM terminal with a single tuner, this would necessarily cause the underlying hardware for the tuner to be taken away from the `ServiceContext`.

## Z.4.3 Multiple-Application Model

When multiple applications are running as part of the same service, it is important to understand the relationship between the `javax.tv.service.ServiceContext` instances in the two applications. Both applications may obtain `ServiceContext` instances using `ServiceContext.getServiceContext(XletContext)`, in order to manipulate the service context in which the two applications are running. In this case the two applications will, of course, have different instances of `ServiceContext`, but they represent the same underlying service context. That is, when one application uses its `ServiceContext` instance, for example to select a different service, it causes this action to happen in the underlying service context. If, then, the other application selects yet another service, this also occurs in the same underlying service context, so no additional acquisition of resources (e.g. as reported through the DAVIC resource management API) need take place.

## Z.4.4 Considerations for standalone stored services

A standalone stored service contains only applications, and no A/V content. Further, because it is stored, no tuner is required. Thus, when a service context is presenting a standalone stored service, it is not required to be bound to a tuner or an audio or video decoder. However, the same service context may in the future be used to select a service that does include audio/video context, or that is a broadcast service received over a tuner. If such a service is selected, it is currently unspecified whether the tuner and A/V decoders remain bound to the service context during the presentation of the standalone stored service, or whether they are released and re-acquired. If an application has registered for `ResourceStatusEvent` notifications, it may or may not receive them, depending on the underlying implementation choice.

# Z.5 Multiple service contexts in a GEM platform

The GEM specification allows the platform to have any number of service contexts, although the platform may choose to limit the number it can produce, possibly even to one. Each service context can present a different service, completely independently of the other service contexts. Operations carried out on one service context will not affect another (unless they cause tuning which prevents the contents of a service in another service context from being presented). It is even possible to display the same service in several service contexts simultaneously - this may not be very useful, but it is allowed. This may result in several instances of the same application running in at the same time in different service contexts. In practice, this is most likely to happen in future GEM terminals with multiple independent video output channels.

# Z.6 How does the platform know which services are available?

In order to select a new service, an application (or the platform) has to know three things about the service it wants to start: the original network ID, the transport stream ID and the service ID for the service in question. In the case of an application, those values may be hard-wired into the application by the developer, but this not required. The application can find out about a service in the same way that the platform can - using service information.

The platform can not know the details of every available service when it started for the first time, and so it needs another method to find out about what services it can receive. A set-top box may do the following to find this out, for instance:

a) Scan the input (satellite, cable or some other input) to find out which transport streams are available and the physical parameters it needs to tune to them successfully.

b) For every transport stream:

   - Tune to it.
   - Use the service information API to access the service description table (SDT) and network information table (NIT) for that transport stream.
   - Use these tables to find what services are in the transport stream and the values needed to select those services.
   - In the case of the platform performing an initial scan, write this information to non-volatile memory.

c) When the application has found the service it wants (or in the case of the platform performing a scan, once every transport stream has been scanned in this way), a service can be selected.

At this point, the platform has all the information it needs to start presenting services to the user. It is important to realize that this is not the only way of finding this information - other ways may be possible, depending on the GEM implementation.

# Annex AA:
# Void

# Annex AB:
# Void

# Annex AC:
# Void

# Annex AD:
# Void

# Annex AE (normative): Inner Applications

# Package org.dvb.application.inner

### Description

Provides support for embedding other interactive applications within the user interface of a DVB-J application.

| Class Summary | |
|---|---|
| **Interfaces** | |
| DVBScene | Represents various features which are common to the top level Container of applications whether running stand-alone or embedded in a DVB-HTML application. |
| InnerApplicationListener | Listener for events related to inner applications |
| **Classes** | |
| InnerApplication | Encapsulates the information needed to define an inner application. |
| InnerApplicationContainer | Represents embedding of an inner application within the user interface of a DVB-J application. |
| InnerApplicationEvent | InnerApplicationEvent is used to notify applications which have registered interest in events coming from inner applications that such an event has happened. |

# org.dvb.application.inner DVBScene

### Declaration

```
public interface DVBScene
```

### Description

Represents various features which are common to the top level Container of applications whether running stand-alone or embedded in a DVB-HTML application. All integer constants used in this class are those defined in the class org.havi.ui.HScene.

### Methods

#### addShortcut(int, HActionable)

```
public boolean addShortcut(int keyCode, org.havi.ui.HActionable comp)
```

Adds a shortcut key to action the specified `HActionable` . Generating the defined java.awt.KeyEvent or `HRcEvent` keycode causes the specified component to become actioned ——- potentially any keyCode may have a shortcut associated with it. The shortcut will only be added if the `HActionable` component is a child component in the container hierarchy of which the `HScene` is the root container. If this is not the case this method shall return false.

Note that a maximum of one `HActionable` may be associated with a given keyCode. An `HActionable` can have at most one associated shortcut keycode. Calling `addShortcut` repeatedly with the same `HActionable` will result in the previous short cut being removed. A short cut can be set on an invisible `HActionable` and therefore it is possible to provide short-cuts that have no user representation.

If the relevant keyCode is received by the HScene , then the HActionable will be actioned by the HScene sending it an ACTION_PERFORMED . Note that it is the responsibility of the application to ensure that the keyCode used is included in the set registered with the setKeyEvents method and is one which the platform can generate, i.e. where the method HRcCapabilities.isSupported() returns true.

Parameters:

keyCode - the keycode that represents the short cut. If keycode is java.awt.event.KeyEvent#VK_UNDEFINED, then the shortcut will not be added, any existing shortcut for this component will not be changed and this method shall return false.

comp - The actionable component that will be actioned.

Returns:

true if the shortcut was added, false otherwise.

### addWindowListener(WindowListener)

```
public void addWindowListener(java.awt.event.WindowListener wl)
```

Add a listener to receive any java.awt.event.WindowEvents sent from this DVBScene . If the listener has already been added further calls will add further references to the listener, which will then receive multiple copies of a single event.

Parameters:

wl - The java.awt.event.WindowListener to be notified of any java.awt.event.WindowEvents.

### enableShortcuts(boolean)

```
public void enableShortcuts(boolean enable)
```

Enables or disables all short cuts that are currently set on the Scene. To enable or disable a single shortcut use addShortcut or removeShortcut .

Note enableShortcuts(false) does not remove existing added DVBScene shortcuts - they are merely disabled and may be subsequently re-enabled with enableShortcuts(true) .

Parameters:

enable - a value of true indicates all shortcuts are to be enabled, and a value of false indicates all shortcuts are to be disabled.

### getAllShortcutKeycodes()

```
public int[] getAllShortcutKeycodes()
```

Returns all keycodes added in the DVBScene as shortcuts.

Returns:

all keycodes added in the DVBScene as shortcuts, there are no ordering guarantees.

### getBackgroundImage()

```
public java.awt.Image getBackgroundImage()
```

Retrieve any image used as a background for this DVBScene .

Returns:

an image used as a background, or null if no image is set. Note that depending on the current render mode any image set may not actually be rendered.

See Also:

setRenderMode(int)

### getBackgroundMode()

```
public int getBackgroundMode()
```

Get the background mode of this DVBScene . The return value specifies whether the paint method should draw the background (i.e. a rectangle filling the bounds of the DVBScene).

Returns:

one of NO_BACKGROUND_FILL or BACKGROUND_FILL.

### getFocusOwner()

```
public java.awt.Component getFocusOwner()
```

Returns the child component of this DVBScene which has focus if and only if this DVBScene is active.

Returns:

the component with focus, or null if no children have focus assigned to them.

### getRenderMode()

```
public int getRenderMode()
```

Get the rendering mode of any background image associated with this DVBScene.

Returns:

the rendering mode, one of IMAGE_NONE , IMAGE_STRETCH , IMAGE_CENTER or IMAGE_TILE .

### getShortcutComponent(int)

```
public org.havi.ui.HActionable getShortcutComponent(int keyCode)
```

Retrieve the HActionable associated with the specified shortcut key.

Parameters:

keyCode - the shortcut key code to be queried for an associated HActionable.

Returns:

the HActionable associated with the specified key if keyCode is a valid shortcut key for this DVBScene , null otherwise.

### getShortcutKeycode(HActionable)

```
public int getShortcutKeycode(org.havi.ui.HActionable comp)
```

Returns the keycode associated with the specified HActionable component.

Parameters:

comp - the HActionable to return the keycode that it is associated with.

Returns:

the keycode associated with the specified HActionable component, if it is currently a valid shortcut "target", otherwise return java.awt.event.KeyEvent#VK_UNDEFINED.

### isDoubleBuffered()

```
public boolean isDoubleBuffered()
```

Returns true if all the drawing done during the update and paint methods for this specific DVBScene object is automatically double buffered.

Returns:

`true` if all the drawing done during the update and paint methods for this specific `DVBScene` object is automatically double buffered, or `false` if drawing is not double buffered. The default value for the double buffering setting is platform-specific.

### isEnableShortcuts()

```
public boolean isEnableShortcuts()
```

Returns the status of all short cuts that are currently set on the DVBScene.

Returns:

true if shortcuts are enabled, false otherwise.

See Also:

`enableShortcuts(boolean)`

### isOpaque()

```
public boolean isOpaque()
```

Returns true if the entire `DVBScene` area, as given by the `java.awt.Component#getBounds` method, is fully opaque, i.e. its paint method (or surrogate methods) guarantee that all pixels are painted in an opaque `Color`.

By default, the return value depends on the value of the current background mode, as set by the `setBackgroundMode` method. The return value should be overridden by subclasses that can guarantee full opacity. The consequences of an invalid overridden value are implementation specific.

Returns:

`true` if all the pixels within the area given by the `java.awt.Component#getBounds` method are fully opaque, i.e. its paint method (or surrogate methods) guarantee that all pixels are painted in an opaque Color, otherwise `false`.

### isVisible()

```
public boolean isVisible()
```

Determines if the `DVBScene` (or more properly its added child components) is Visible. Initially an `DVBScene` is invisible.

Returns:

true if the `DVBScene` is visible; false otherwise.

### removeShortcut(int)

```
public void removeShortcut(int keyCode)
```

Removes the specified short-cut key. if the specified short-cut key is not registered, the method has no effect

Parameters:

`keyCode` - The keycode that represents the short cut.

### removeWindowListener(WindowListener)

```
public void removeWindowListener(java.awt.event.WindowListener wl)
```

Remove a listener so that it no longer receives any java.awt.event.WindowEvents. If the specified listener is not registered, the method has no effect. If multiple references to a single listener have been registered it should be noted that this method will only remove one reference per call.

Parameters:

`wl` - The java.awt.event.WindowListener to be removed from notification of any java.awt.event.WindowEvents.

### setBackgroundImage(Image)

```
public void setBackgroundImage(java.awt.Image image)
```

Set an image which shall be painted in the background of the DVBScene , after the background has been drawn according to the current mode set with setBackgroundMode, but before any children are drawn. The image is rendered according to the current render mode set with setRenderMode.

Note that the use of a background image in this way may affect the return value of the isOpaque method, depending on the image and the current rendering mode.

Parameters:

image - the image to be used as a background. If this parameter is null any current image is removed. Note that depending on the current render mode any image set may not actually be rendered.

See Also:

setRenderMode(int)

### setBackgroundMode(int)

```
public void setBackgroundMode(int mode)
```

Set the background mode of this DVBScene . The value specifies whether the paint method should draw the background (i.e. a rectangle filling the bounds of the DVBScene ).

Note that the background mode will affect the return value of the isOpaque method, depending on the value of the mode parameter. A fill mode of BACKGROUND_FILL implies that isOpaque must return true.

Parameters:

mode - one of NO_BACKGROUND_FILL or BACKGROUND_FILL . If mode is not a valid value, an IllegalArgument-Exception will be thrown.

### setRenderMode(int)

```
public boolean setRenderMode(int mode)
```

Set the rendering mode of any background image associated with this DVBScene .

Note that the minimum requirement is to support only the IMAGE_NONE mode. Support of the other modes is platform and implementation specific.

Parameters:

mode - the rendering mode, one of IMAGE_NONE , IMAGE_STRETCH , IMAGE_CENTER or IMAGE_TILE .

Returns:

true if the mode was set successfully, false if the mode is not supported by the platform.

# org.dvb.application.inner

# InnerApplication

### Declaration

```
public abstract class InnerApplication

java.lang.Object
  |
  +--org.dvb.application.inner.InnerApplication
```

### Direct Known Subclasses:

org.dvb.dom.inner.HTMLApplication

### Description

Encapsulates the information needed to define an inner application.

### Constructors

### InnerApplication()

```
protected InnerApplication()
```

A protected constructor. This shall not be used by inter-operable applications.

# org.dvb.application.inner

# InnerApplicationContainer

### Declaration

```
public abstract class InnerApplicationContainer extends org.havi.ui.HComponent implements
org.havi.ui.HNavigable

java.lang.Object
  |
  +--java.awt.Component
        |
        +--org.havi.ui.HComponent
              |
              +--org.dvb.application.inner.InnerApplicationContainer
```

### All Implemented Interfaces:

```
org.havi.ui.HMatteLayer, org.havi.ui.HNavigable, org.havi.ui.HNavigationInputPreferred,
java.awt.image.ImageObserver, java.io.Serializable, org.dvb.ui.TestOpacity
```

### Direct Known Subclasses:

```
org.dvb.dom.inner.HTMLInnerApplicationContainer
```

### Description

Represents embedding of an inner application within the user interface of a DVB-J application. The inner application shall be started when the object of this class is constructed. The behaviour of instances of this class when an inner application exits is either implementation or inner application format dependent.

When an instance of this class gains input focus,shall gain the input focus. The meaning of this is specific to the format of the inner application concerned. Inner applications are allowed to consume the events VK_UP, VK_DOWN, VK_LEFT and VK_RIGHT. While this is happening, the HNavigable focus management shall be disabled and the actions defined by `setFocusTraversal` shall not happen. The mechanisms by which inner applications start and stop consuming these events are specific to the content format of the inner application.

### Constructors

### InnerApplicationContainer(InnerApplication)

```
protected InnerApplicationContainer(org.dvb.application.inner.InnerApplication a)
throws IOException
```

Construct an instance of this class with a particular inner application as its content. This shall not be used by inter-operable applications.

### Parameters:

`a` - the inner application.

### Throws:

`java.io.IOException` - if an error occurred while reading the code or data for the inner application.

### Methods

#### addHFocusListener(HFocusListener)

```
public void addHFocusListener(org.havi.ui.event.HFocusListener l)
```

Adds the specified `HFocusListener` to receive `HFocusEvent` events sent from this `HNavigable`. If the listener has already been added further calls will add further references to the listener, which will then receive multiple copies of a single event.

Specified By:

`addHFocusListener` in interface `HNavigable`

Parameters:

`l` - the HFocusListener to add.

#### addInnerApplicationListener(InnerApplicationListener)

```
public void addInnerApplicationListener(org.dvb.application.inner.InnerApplicationListener l)
```

Add a listener for events when the inner application exits.

Parameters:

`l` - the listener to be notified when the inner application exits.

#### getGainFocusSound()

```
public org.havi.ui.HSound getGainFocusSound()
```

Get the sound associated with the gain focus event.

Specified By:

`getGainFocusSound` in interface `HNavigable`

Returns:

The sound played when the component gains focus. If no sound is associated with gaining focus, then null shall be returned.

#### getLoseFocusSound()

```
public org.havi.ui.HSound getLoseFocusSound()
```

Get the sound associated with the lost focus event.

Specified By:

`getLoseFocusSound` in interface `HNavigable`

Returns:

The sound played when the component loses focus. If no sound is associated with losing focus, then null shall be returned.

#### getMove(int)

```
public org.havi.ui.HNavigable getMove(int keyCode)
```

Provides the `HNavigable` object that is navigated to when a particular key is pressed.

Specified By:

`getMove` in interface `HNavigable`

Parameters:

keyCode - The key code of the pressed key.

Returns:

Returns the HNavigable object, or if no HNavigable is associated with the keyCode then returns null.

### getNavigationKeys()

```
public int[] getNavigationKeys()
```

Retrieve the set of key codes which this component maps to navigation targets.

Specified By:

getNavigationKeys in interface HNavigationInputPreferred

Returns:

an array of key codes, or null if no navigation targets are set on this component.

### isSelected()

```
public boolean isSelected()
```

Indicates if this component has focus.

Specified By:

isSelected in interface HNavigable

Returns:

true if the component has focus, otherwise returns false.

### processHFocusEvent(HFocusEvent)

```
public void processHFocusEvent(org.havi.ui.event.HFocusEvent evt)
```

Process an HFocusEvent sent to this HNavigationInputPreferred .

Specified By:

processHFocusEvent in interface HNavigationInputPreferred

Parameters:

evt - the HFocusEvent to process.

### removeHFocusListener(HFocusListener)

```
public void removeHFocusListener(org.havi.ui.event.HFocusListener l)
```

Removes the specified HFocusListener so that it no longer receives HFocusEvent events from this HNavigable. If the specified listener is not registered, the method has no effect. If multiple references to a single listener have been registered it should be noted that this method will only remove one reference per call.

Specified By:

removeHFocusListener in interface HNavigable

Parameters:

l - the HFocusListener to remove.

### removeInnerApplicationListener(InnerApplicationListener)

```
public void removeInnerApplicationListener(org.dvb.application.inner.InnerApplicationListener l)
```

Remove a listener for events when the inner application exits.

Parameters:

`l` - the listener to be removed.

### setFocusTraversal(HNavigable, HNavigable, HNavigable, HNavigable)

```
public void setFocusTraversal(org.havi.ui.HNavigable up, org.havi.ui.HNavigable down,
org.havi.ui.HNavigable left, org.havi.ui.HNavigable right)
```

Set the focus control for an `HNavigable` component. Note `setFocusTraversal` is a convenience function for application programmers where a standard up, down, left and right focus traversal between components is required.

Note `setFocusTraversal` is equivalent to multiple calls to `setMove` , where the key codes VK_UP, VK_DOWN, VK_LEFT, VK_RIGHT are used.

Note that this API does not prevent the creation of "isolated" `HNavigable` components ——- authors should endeavor to avoid confusing the user.

Specified By:

`setFocusTraversal` in interface `HNavigable`

Parameters:

`up` - The `HNavigable` component to move to, when the user generates a VK_UP KeyEvent. If there is no `HNavigable` component to move "up" to, then null should be specified.

`down` - The `HNavigable` component to move to, when the user generates a VK_DOWN KeyEvent. If there is no `HNavigable` component to move "down" to, then null should be specified.

`left` - The `HNavigable` component to move to, when the user generates a VK_LEFT KeyEvent. If there is no `HNavigable` component to move "left" to, then null should be specified.

`right` - The `HNavigable` component to move to, when the user generates a VK_RIGHT KeyEvent. If there is no `HNavigable` component to move "right" to, then null should be specified.

### setGainFocusSound(HSound)

```
public void setGainFocusSound(org.havi.ui.HSound sound)
```

Associate a sound with gaining focus, i.e. when the `HNavigable` receives a `java.awt.event.FocusEvent` event of type `FOCUS_GAINED`. This sound will start to be played when an object implementing this interface gains focus. It is not guaranteed to be played to completion. If the object implementing this interface loses focus before the audio completes playing, the audio will be truncated. Applications wishing to ensure the audio is always played to completion must implement special logic to slow down the focus transitions.

By default, an `HNavigable` object does not have any gain focus sound associated with it.

Note that the ordering of playing sounds is dependent on the order of the focus lost, gained events.

Specified By:

`setGainFocusSound` in interface `HNavigable`

Parameters:

`sound` - the sound to be played, when the component gains focus. If sound content is already set, the original content is replaced. To remove the sound specify a null `HSound` .

### setLoseFocusSound(HSound)

```
public void setLoseFocusSound(org.havi.ui.HSound sound)
```

Associate a sound with losing focus, i.e. when the HNavigable receives a java.awt.event.FocusEvent event of type FOCUS_LOST. This sound will start to be played when an object implementing this interface loses focus. It is not guaranteed to be played to completion. It is implementation dependent whether and when this sound will be truncated by any gain focus sound played by the next object to gain focus.

By default, an HNavigable object does not have any lose focus sound associated with it.

Note that the ordering of playing sounds is dependent on the order of the focus lost, gained events.

#### Specified By:

setLoseFocusSound in interface HNavigable

#### Parameters:

sound - the sound to be played, when the component loses focus. If sound content is already set, the original content is replaced. To remove the sound specify a null HSound .

### setMove(int, HNavigable)

```
public void setMove(int keyCode, org.havi.ui.HNavigable target)
```

Defines the navigation path from the current HNavigable to another HNavigable when a particular key is pressed.

Note that setFocusTraversal is equivalent to multiple calls to setMove , where the key codes VK_UP, VK_DOWN, VK_LEFT, VK_RIGHT are used.

#### Specified By:

setMove in interface HNavigable

#### Parameters:

keyCode - The key code of the pressed key. Any numerical keycode is allowed, but the platform may not be able to generate all keycodes. Application authors should only use keys for which HRcCapabilities.isSupported() or HKeyCapabilities.isSupported() returns true.

target - The target HNavigable object that should be navigated to. If a target is to be removed from a particular navigation path, then null should be specified.

# org.dvb.application.inner

# InnerApplicationEvent

### Declaration

```
public class InnerApplicationEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.application.inner.InnerApplicationEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

InnerApplicationEvent is used to notify applications which have registered interest in events coming from inner applications that such an event has happened.

### Constructors

#### InnerApplicationEvent(InnerApplicationContainer)

```
public InnerApplicationEvent(org.dvb.application.inner.InnerApplicationContainer source)
```

Constructor for an event.

Parameters:

`source` - the `InnerApplicationContainer` which generated the event.

# org.dvb.application.inner

# InnerApplicationListener

### Declaration

```
public interface InnerApplicationListener
```

### Description

Listener for events related to inner applications.

### Methods

#### InnerApplicationExited(InnerApplicationEvent)

```
public void InnerApplicationExited(org.dvb.application.inner.InnerApplicationEvent e)
```

Called when an inner application exits.

Parameters:

`e` - an event object encapsulating what happened.

# Annex AF (normative): Plug-in APIs

NOTE: Support for the classes `XletContainer` and `XletSystemCall` is optional in the present document.

# Package org.dvb.application.plugins

## Description

Provides support for inter-operable plug-ins in the GEM specification. The following classes and interfaces are intended for generic plug-ins and are not specific to any particular content format.

- InvalidApplicationException.
- Plugin.
- ApplicationSecurityContext.

The following classes are intended specifically for DVB-HTML plug-ins.

- XletContainer.
- XletSystemCall.

| Class Summary | |
|---|---|
| **Interfaces** | |
| Plugin | This interface allows an application or service manager to control the lifecycle of applications being executed by an inter-operable plug-in. |
| **Classes** | |
| ApplicationSecurityContext | This class represents the security context for an application whose permissions are defined by the GEM security model, in particular a permission request file. |
| XletContainer | This class provides a container that can be embedded in a widget hierarchy, and can provide a container that can safely be used as the top-level container of an embedded Xlet. |
| XletSystemCall | This class permits user code to intercept certain system calls initiated by an embedded Xlet that need to be serviced by a support application. |
| **Exceptions** | |
| InvalidApplicationException | Thrown if an application is not valid for execution by a plug-in. |

# org.dvb.application.plugins ApplicationSecurityContext

## Declaration

```
public class ApplicationSecurityContext

java.lang.Object
  |
  +--org.dvb.application.plugins.ApplicationSecurityContext
```

### Description

This class represents the security context for an application whose permissions are defined by the GEM security model, in particular a permission request file. One example of this is DVB-HTML applications managed by a plug-in application. This would also apply to applications in other content format where the permissions for that content format are signalled with GEM mechanisms and governed by the GEM security model.

### Constructors

#### ApplicationSecurityContext(URL[], URL, AppID)

```
public ApplicationSecurityContext(java.net.URL[] path, java.net.URL entryPoint,
org.dvb.application.AppID appID)
throws IOException
```

Creates a new security context for an application whose code can be found on the path supplied, and whose entry point directory is as given. Under no circumstances will an application managed by a plug-in be given access to resources not available to the plug-in itself. This policy is reflected in the permissions granted to the ApplicationSecurityContext, as well as the permissions granted to any classes loaded by any class loaders managed by a security context.

If there is a permission request file in the directory identified by the entryPoint, it will be processed in the same way as the permission request file of a DVB-J application. i.e. reading it in, parsing it and taking account of the access rights granted by the user as defined under "General principles" in the main body of the present document.

#### Parameters:

`path` - The search path for locating resources within the application.

`entryPoint` - The directory containing the permission request file to use.

`appID` - the application ID which the application is to run under.

#### Throws:

IOException - when there is an IO error reading in the permission request file or attempting to read in the permission request file or attempting to discover the existence of a permission request file.

`java.lang.NullPointerException` - if entryPoint is null, if path is null, or if any element of path is null.

`java.lang.IllegalArgumentException` - if path.length < 1.

`java.io.IOException`.

### Methods

#### checkPermission(Permission)

```
public void checkPermission(java.security.Permission p)
```

Throws a `SecurityException` if the requested access, specified by the given permission, is not permitted to the application or sub-application represented by this application security context object. The set of permissions granted to an entity is be a function of receiver policy, possibly influenced by user settings, application signer, and permission request file.

#### Parameters:

`p` - A permission object representing the resource for which access is being checked.

#### Throws:

`java.lang.NullPointerException` - if p is null.

`java.lang.SecurityException` - if this application has not been granted access to the resource represented by p.

### createEmbeddedContext(URL[], URL)

```
public org.dvb.application.plugins.ApplicationSecurityContext createEmbeddedContext(java.net.URL[]
path, java.net.URL entryPoint)
```

Creates a context for an embedded part of an application, e.g. an Xlet embedded within a DVB-HTML page. The set of permissions granted to the application will be the same as the parent ApplicationSecurityContext.

Parameters:

`path` - The search path for locating resources within the application.

`entryPoint` - The resource of the entry point of this application.

Returns:

a context for the part of the application concerned.

Throws:

`java.lang.NullPointerException` - if entryPoint is null, if path is null, or if any element of path is null.

`java.lang.IllegalArgumentException` - if path.length < 1.

### doPrivileged(PrivilegedAction)

```
public java.lang.Object doPrivileged(java.security.PrivilegedAction action)
```

Performs the specified PrivilegedAction with privileges enabled and restricted by the specified AccessControlContext. The action is performed with the intersection of the permissions possessed by the caller's protection domain, and those possessed by the domains represented by the specified AccessControlContext. If the action's run method throws an (unchecked) exception, it will propagate through this method.

Parameters:

`action` - the action to be performed.

Returns:

the value returned by the action's run method.

### getClassLoader(String[])

```
public java.lang.ClassLoader getClassLoader(java.lang.String[] forbiddenPackages)
```

Get a classloader that is appropriate for loading classes for the application (or sub-application) represented by this application security context object. If this method is called more than once, the same instance will be returned.

It is important that embedded DVB-J code be prevented from accessing classes that implement the plug-in application. To this end, the plug-in may specify a list of forbidden packages. Classes loaded by the returned classloader will be forbidden from loading or accessing classes in the named packages.

Parameters:

`forbiddenPackages` - a list of forbidden package names, e.g. `{ "de.tu-bs.ing.ifn.plugin.impl" }`.

Returns:

A class loader that is appropriate for loading and DVB-J classes that are a part of this application or sub-application.

### getResource(String, boolean)

```
public java.net.URL getResource(java.lang.String name, boolean sameSigner)
```

Get a locator to the named resource, within the search path for this application. Will return null if a resource with the given name that is appropriately signed (if necessary) cannot be found.

NOTE: This method can be used, for example, by an interoperable plug-in that needs to fetch part of an application that is not loaded by a classloader. For example, it could be used to get a locator to an HTML page, if and only if that page is appropriately signed.

Parameters:

`name` - The name of the resource (e.g. com/foo/MyPage.html).

`sameSigner` - True if this is code, or any other resource for which the signer must be the same as the signer of the entry point.

Returns:

URL to the named resource, or null.

# org.dvb.application.plugins

# InvalidApplicationException

Declaration

```
public class InvalidApplicationException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.application.plugins.InvalidApplicationException
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

Thrown if an application is not valid for execution by a plug-in.

Since:

MHP1.1.

Constructors

InvalidApplicationException()

```
public InvalidApplicationException()
```

Construct a InvalidApplicationException with no detail message.

InvalidApplicationException(String)

```
public InvalidApplicationException(java.lang.String s)
```

Construct a InvalidApplicationException with a detail message.

Parameters:

`s` - detail message.

# org.dvb.application.plugins

# Plugin

### Declaration

```
public interface Plugin
```

### Description

This interface allows an application or service manager to control the lifecycle of applications being executed by an inter-operable plug-in. It shall be implemented by inter-operable plug-ins and shall be called by implementation of the application manager in a GEM terminal. It shall not be called by GEM applications. The usage model for this interface is the reuse of the `javax.tv.xlet.Xlet` interface to represent applications in formats other than DVB-J. It is the responsibility of the plug-in to provide implementations of the methods in that interface and to perform the translation between the semantics specified for DVB-J applications and the semantics of the application format which the plug-in supports. The plug-in is initialized by the application manager calling the no-argument constructor of the object implementing this interface. Plug-ins are not expected to perform any time consuming activities or hold any scarce resources in this call.

### Methods

#### initApplication(AppAttributes)

```
public javax.tv.xlet.Xlet initApplication(org.dvb.application.AppAttributes app)
throws InvalidApplicationException
```

Request the plug-in to prepare for executing an instance of an application in a format which it supports. If the application cannot be prepared then null shall be returned. While the plug-in is in the terminated state then this method shall always return null. The application shall only be considered to be started when the application manager calls the methods on the Xlet interface returned by this method.

Parameters:

`app` - an instance of `AppAttributes` for the application which is to be started.

Returns:

an object implementing the Xlet interface or null if the application cannot be started.

Throws:

`InvalidApplicationException` - if the application to be started is not valid for this plug-in.

#### initApplication(InnerApplication)

```
public javax.tv.xlet.Xlet initApplication(org.dvb.application.inner.InnerApplication app)
throws InvalidApplicationException
```

Request the plug-in to prepare for executing an instance of an application in a format which it supports.If the application cannot be prepared then null shall be returned. While the plug-in is in the terminated state then this method shall always return null. The application shall only be considered to be started when the application manager calls the methods on the Xlet interface returned by this method.

Parameters:

`app` - an instance of InnerApplication for the application which is to be started.

Returns:

an object implementing the Xlet interface or null if the application cannot be started.

Throws:

`InvalidApplicationException` - if the application to be started is not valid for this plug-in.

### initPlugin()

```
public boolean initPlugin()
```

Initialize the plug-in. Any time consuming initializations should be performed during this method call. This method shall be called after the no-argument constructor of the object implementing this interface and before any calls to the `startApplication`. It may also be called after a call to `terminatePlugin` if a plug-in was not removed from the virtual machine where it was executing. The behaviour of an application manager should a plug-in fail to initialize is intentionally unspecified.

#### Returns:

true if the plug-in initialized successfully otherwise false.

### isSupported(AppAttributes)

```
public boolean isSupported(org.dvb.application.AppAttributes app)
```

Test whether this plug-in is able to support a particular application. This method shall work regardless of whether the plug-in has been initialized or not.

#### Parameters:

`app` - an instance of `AppAttributes` for the application which is to be started.

#### Returns:

true if the plug-in can support this application otherwise false.

### terminatePlugin()

```
public void terminatePlugin()
```

Terminate the plug-in. This method shall only be called after all applications being executed by this plug-in have been terminated. The implementation of this method shall release all resources held by the plug-in. Once this method has returned, the plug-in shall either be unloaded from the virtual machine where it was executing or the `initPlugin` method shall be called again to put the plug-in again in an initialized condition. A plug-in may not refuse to terminate and throwing a runtime exception or error from this method shall result in the plug-in being removed from the virtual machine where it was executing.

# org.dvb.application.plugins

# XletContainer

### Declaration

```
public class XletContainer extends java.awt.Container
```

```
java.lang.Object
  |
  +--java.awt.Component
        |
        +--java.awt.Container
              |
              +--org.dvb.application.plugins.XletContainer
```

### All Implemented Interfaces:

```
java.awt.image.ImageObserver, java.io.Serializable
```

### Description

This class provides a container that can be embedded in a widget hierarchy, and can provide a container that can safely be used as the top-level container of an embedded Xlet. This solves the problem of an embedded Xlet being able to call getParent() to discover the widget hierarchy in which it is embedded.

Since:

MHP 1.1.

### Constructors

#### XletContainer(Container)

```
public XletContainer(java.awt.Container parent)
```

Construct a new XletContainer as a child of Parent. The XletContainer contains exactly one widget: A child container. The child container is constrained to be at the same location and the same size as the parent XletContainer.

Parameters:

`parent` - The widget that is to have an Xlet embedded within it.

Throws:

`java.lang.NullPointerException` - if parent is null.

See Also:

`getXletContainer()`

### Methods

#### getXletContainer()

```
public java.awt.Container getXletContainer()
```

Get the only child of the XletContainer. This container can safely be used as the top-level container of an Xlet, e.g. an Xlet embedded in a DVB-HTML page. Calling getParent() on this container from client code will return null, unless the caller has been granted special access via a platform-dependent mechanism.

If this method is invoked more than once for the same instance of XletContainer, it will return the same instance.

Returns:

The child of this XletContainer.

# org.dvb.application.plugins

# XletSystemCall

### Declaration

```
public abstract class XletSystemCall
```

```
java.lang.Object
  |
  +--org.dvb.application.plugins.XletSystemCall
```

### Description

This class permits user code to intercept certain system calls initiated by an embedded Xlet that need to be serviced by a support application. For example, a DVB-HTML plug-in application needs to service requests that are made by an embedded Xlet, typically via static method calls.

Since:

MHP1.1.

### Constructors

#### XletSystemCall()

```
protected XletSystemCall()
```

Create a new XletSystemCall.

### Methods

#### getRootContainer(XletContext)

```
public abstract java.awt.Container getRootContainer(javax.tv.xlet.XletContext ctx)
```

Called when the Xlet calls javax.tv.graphics.TVContainer.getRootContainer().

Parameters:

`ctx` - The context of the Xlet making the request; it shall be identical to the XletContext used to create this instance of XletSystemCall.

Returns:

a container object to be returned to the embedded xlet

See Also:

`javax.tv.graphics.TVContainer.getRootContainer(XletContext)`

#### register(Plugin, XletContext)

```
public final void register(org.dvb.application.plugins.Plugin p, javax.tv.xlet.XletContext ctx)
```

Register this instance of XletSystemCall with the system.

Parameters:

`p` - The Plugin that services calls made by the xlet, i.e. the Plugin of which this instance of XletSystemCall is a part.

`ctx` - The XletContext of the Xlet making the calls.

Throws:

`java.lang.NullPointerException` - if p or ctx is null.

See Also:

`unregister(Plugin, XletContext)`

#### unregister(Plugin, XletContext)

```
public final void unregister(org.dvb.application.plugins.Plugin p, javax.tv.xlet.XletContext ctx)
```

Unregister this instance of XletSystemCall with the system. When an interoperable Plugin terminates, of an Xlet managed by a Plugin is asked to terminate, the Plugin must unregister any relevant XletSystemCall instances.

Parameters:

`p` - The Plugin that services calls made by the xlet, i.e. the Plugin of which this instance of XletSystemCall is a part.

`ctx` - The XletContext of the Xlet making the calls.

Throws:

`java.lang.NullPointerException` - if p or ctx is null.

See Also:

`register(Plugin, XletContext)`

# Annex AG (normative): Stored application APIs

# Package org.dvb.application.storage

Description

Provides support for storage of applications.

| Class Summary | |
|---|---|
| **Interfaces** | |
| ApplicationStorageController | Defines the methods for storing, listing and removing applications to and from a service. |
| ApplicationStorageListener | Listener to receive reports on progress of application storage operations. |
| ExtendedAppAttributes | The `ExtendedAppAttributes` interface provides additional attributes that are useful when application can be stored in the GEM terminal. |
| StoredApplicationService | Defines the information about a stored application service. |
| **Classes** | |
| ApplicationCache | A cache to store applications so that they may be started faster when signalled in broadcast. |
| ApplicationStoragePermission | This class represents a permission to manage applications stored in the GEM terminal. |
| StoredApplicationServiceFactory | This factory creates new Service objects representing stand-alone stored application services. |
| StoredApplicationServiceType | The service type used for stored application services. |
| **Exceptions** | |
| ApplicationDownloadException | Thrown if the downloading of the application failed. |
| InvalidApplicationException | When an application is being installed into a service, this is thrown when: The application does not include an `application_storage_descriptor`. |
| InvalidDescriptionFileException | Thrown when an application descriptio file is invalid. |
| NotEnoughResourcesException | Thrown if the GEM terminal does not have enough resources to install the application. |
| ServiceAlreadyExistsException | Thrown if a StoredApplicationService already exists with the identifiers the application tried to create a new one. |
| UserRejectedInstallException | Thrown if the request to install an application was rejected. |

# org.dvb.application.storage ApplicationCache

Declaration

```
public abstract class ApplicationCache

java.lang.Object
  |
  +--org.dvb.application.storage.ApplicationCache
```

### Description

A cache to store applications so that they may be started faster when signalled in broadcast.

Applications stored via this mechanism cannot be started directly, only via an AIT entry in a broadcast service.

Note that applications are globally uniquely identified by a combination of organisation ID, application ID, and version number. Application authors should ensure that their storable applications are consistently broadcast.

Each instance of ApplicationCache can store only a single version of an application - i.e. within an `ApplicationCache`, an organisation ID and application ID pair can uniquely identify an application. Successfully storing a different version of a cached application than the current one in an ApplicationCache shall result in the current one being removed from that ApplicationCache instance. If the call to store fails then the current version shall not be removed.

A single GEM terminal will support many instances of `ApplicationCache`. These shall share a single underlying application store, and that underlying store shall use reference-counting or similar techniques to allow several stored services and `ApplicationCache` instances to contain the same application without wasting storage space by storing the same application multiple times. Space permitting, the underlying application store shall support storing several different versions of a single application (i.e. same organisation ID and application ID but different version numbers) where these are stored via different `ApplicationCache` instances or form part of different stored services. The underlying application store shall manage the removal of versions of stored applications which are not used in any stored services or application caches.

If an application is already cached in one `ApplicationCache`, storing the same version of the same application in a different `ApplicationCache` shall not fail due to insufficient disk space, and if all files in the ADF are labelled as critical, it shall not fail due to inability to download files listed in the ADF and shall complete quickly (without waiting for the files listed in the ADF to be broadcast).

### Since:

MHP1.1.2.

### Constructors

### ApplicationCache()

```
protected ApplicationCache()
```

Interoperable applications shall not call this constructor.

### Since:

MHP1.1.2.

### Methods

### getDefaultCache()

```
public static org.dvb.application.storage.ApplicationCache getDefaultCache()
```

Get the default cache that this application can use. Note that in MHP 1.1, applications cannot access any cache other than the default one.

The object returned depends on the organisation_id of the calling application. Applications with the same organisation_id shall recieve the same `ApplicationCache`. Applications with different organisation_id shall recieve different, independent instances of `ApplicationCache`. (I.e. if one application stores an application or removes a stored application, that operation shall be visible to another application via the methods on this class if and only if the other application has the same organisation_id as the first application).

### Returns:

An `ApplicationCache` instance.

### Throws:

`java.lang.SecurityException` - Thrown if the application calling this method does not have an `ApplicationStoragePermission` with action "manageCache" for it's own organisation_id.

Since:

MHP1.1.2.

### getStoredAppIDs()

```
public abstract org.dvb.application.AppID[] getStoredAppIDs()
```

Lists the AppIDs of the applications that are stored within this cache.

Returns:

an array of AppID objects representing the stored applications.

Since:

MHP1.1.2.

### getVersionNumber(AppID)

```
public abstract int getVersionNumber(org.dvb.application.AppID appId)
```

Return the version number of the stored application whose AppID is given as a parameter.

Parameters:

appId - the AppID of the application whose version is queried.

Returns:

the version number of the stored application, returns -1 if the application given as a parameter is not stored as part of this cache.

Since:

MHP1.1.2.

### remove(AppID)

```
public abstract void remove(org.dvb.application.AppID appId)
```

Requests the GEM terminal to initiate the removal of an application stored in the GEM terminal from this cache.

The platform is not expected to consult the end user of the GEM terminal for the permission to remove the application.

This specification does not prevent a terminal keeping an application in cache even after all references to it have been removed via this API.

If the application identified by the AppID passed in as a parameter is not installed in this cache, the method shall fail silently.

If the application is already stored elsewhere (e.g. in another ApplicationCache, or in a StoredApplicationService) then those caches and stored services shall not be affected by calling this method.

Parameters:

appId - AppID of the application to be removed.

Throws:

java.lang.SecurityException - thrown if the application calling this method does not have an ApplicationStoragePermission with action "removeCache" for the organisation_id of the application to be removed and an ApplicationStoragePermission with action "manageCache".

Since:

MHP1.1.2.

### store(AppProxy, boolean)

```
public abstract void store(org.dvb.application.AppProxy app, boolean canPrompt)
throws InvalidApplicationException, NotEnoughResourcesException, InvalidDescriptionFileException, Ap
plicationDownloadException
```

Store an application in the GEM terminal.

GEM terminals may prompt the user for permission to free up resources if and only if all of the following conditions hold:

- `canPrompt` is true;
- the caller has the necessary permissions (i.e. the `SecurityException` will not be thrown);
- the application is valid (i.e. the `InvalidApplicationException` will not be thrown);
- the application description file is valid (i.e. the `InvalidDescriptionFileException` will not be thrown);
- the GEM terminal determines that there are insufficient resources to store the application, and it cannot or will not silently free up enough resources (i.e. the `NotEnoughResourcesException` would be thrown);
- the GEM terminal determines that it could free up sufficient resources to store the application (i.e. to ensure the `NotEnoughResourcesException` would not be thrown) if it had the user's consent;

Prompting the end-user for permission is not required however applications setting canPrompt to true should be prepared for the possibility of it happening. Note that if the user decides not to allow the terminal to free up resources, the NotEnoughResourcesException will be thrown (as if the terminal had not asked the user at all).

NOTE: This method is synchronous and will block until the storing is either completed or fails.

Storing (the same version of) an application that is already stored elsewhere (e.g. in another ApplicationCache, or in a StoredApplicationService) shall use the already-stored files.

Parameters:

`app` - an AppProxy representing the application to be installed.

`canPrompt` - If `true`, the terminal is allowed to prompt the user for permission to free up space in order to store this application. If `false`, the terminal shall not prompt the user for permission to cache this application.

Throws:

`InvalidApplicationException` - thrown if the specified application is not signalled as capable of being stored, or if the specified application is not signalled as part of the same service as the calling application.

`NotEnoughResourcesException` - thrown if the GEM terminal does not have enough resources, e.g. storage space, available for the application.

`InvalidDescriptionFileException` - thrown if the application description file is missing, invalid or otherwise not conformant to the specification.

`ApplicationDownloadException` - thrown if the downloading of the application files was not successful (e.g. a carousel error, a file in the application description file is missing in the carousel, if the application was removed from the AIT or from its transport mechanism while installation is in progress, etc).

`java.lang.SecurityException` - Thrown if the application calling this method does not have an `ApplicationStoragePermission` with action "storeCache" or "*" for the organisation_id of the application to be stored.

Since:

MHP1.1.2.

# org.dvb.application.storage ApplicationDownloadException

### Declaration

```
public class ApplicationDownloadException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.application.storage.ApplicationDownloadException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown if the downloading of the application failed. E.g. a carousel error, a file in the application description file is missing in the carousel, the application was being authenticated as part of downloading and this failed, the application was specified as being stored from another service and that service is not currently reachable (e.g. requires tuning), etc.

### Since:

MHP1.1.

### Constructors

#### ApplicationDownloadException()

```
public ApplicationDownloadException()
```

Constructs an `ApplicationDownloadException` with no detail message.

#### ApplicationDownloadException(String)

```
public ApplicationDownloadException(java.lang.String s)
```

Constructs an `ApplicationDownloadException` with a detail message

### Parameters:

`s` - detail message.

# org.dvb.application.storage ApplicationStorageController

### Declaration

```
public interface ApplicationStorageController
```

### All Known Subinterfaces:

`StoredApplicationService`, `org.dvb.service.UnboundApplicationService`

### Description

Defines the methods for storing, listing and removing applications to and from a service.

Since:

MHP1.1.3.

Methods

getStoredAppIDs()

```
public org.dvb.application.AppID[] getStoredAppIDs()
```

Lists the AppIDs of the applications that are stored within this service.

Returns:

an array of AppID object representing the stored application.

Throws:

java.lang.SecurityException - Thrown if the application calling this method does not have an ApplicationStoragePermission with action "manageService" for the organisation_id of this service.

Since:

MHP1.1.2.

getVersionNumber(AppID)

```
public int getVersionNumber(org.dvb.application.AppID appId)
```

Return the version number of the stored application whose AppID is given as a parameter.

Parameters:

appId - the AppID of the application whose version is queried.

Returns:

the version number of the stored application, returns -1 if the application given as a parameter is not stored.

Throws:

java.lang.SecurityException - Thrown if the application calling this method does not have an ApplicationStoragePermission with action "manageService" for the organisation_id of this service.

Since:

MHP1.1.2.

remove(AppID)

```
public void remove(org.dvb.application.AppID appId)
throws UserRejectedInstallException
```

Removes a stored application from this service.

Applications should be prepared for the platform consulting the end user of the GEM terminal for the permission to remove the application

If the application identified by the AppID passed in as a parameter is not installed in this service, and the caller has the permissions that would be needed to remove the application if it was installed, the method shall fail silently.

Successfully removing an application from a stored service that is currently being presented shall cause the terminal to send an AppsDatabaseEvent.APP_DELETED event to registered listeners, and if the application is Loaded, Active or Paused then it shall be destroyed as if it had been signalled as KILL in the AIT. (Note that there is no special-case for if an application removes itself from a stored service - this is not an error and shall be handled normally).

Parameters:

appId - AppID of the application to be removed.

Throws:

UserRejectedInstallException - If the user chose not to remove the application.

java.lang.SecurityException - Thrown if the application calling this method does not have an ApplicationStoragePermission with action "removeService" for the organisation_id of the application to be removed, and an ApplicationStoragePermission with action "manageService" for the organisation_id of this service.

Since:

MHP1.1.2.

### remove(AppID[])

```
public void remove(org.dvb.application.AppID[] appIds)
throws UserRejectedInstallException
```

Removes multiple stored applications from this service.

Applications should be prepared for the platform consulting the end user of the GEM terminal for permission to remove the applications

If an application identified by the AppIDs passed in as a parameter is not installed in this service, and the caller has the permissions that would be needed to remove the application if it was installed, then this method shall ignore that AppID.

This method either succeeds or fails completely. It will never remove some installed applications but not other installed applications.

Successfully removing application(s) from a stored service that is currently being presented shall cause the terminal to send an AppsDatabaseEvent.APP_DELETED event for each application to registered listeners, and if any of the removed applications are Loaded, Active or Paused then they shall be destroyed as if they had been signalled as KILL in the AIT. (Note that there is no special-case for if an application removes itself from a stored service - this is not an error and shall be handled normally).

Parameters:

appIds - AppIDs of the applications to be removed.

Throws:

java.lang.IllegalArgumentException - If the array passed to this method has length zero.

java.lang.SecurityException - thrown if the application calling this method does not have ApplicationStoragePermission s with action "removeService" for the organisation_ids of all the applications to be removed, and an ApplicationStoragePermission with action "manageService" for the organisation_id of this service.

UserRejectedInstallException - If the user chose not to remove the application.

Since:

MHP1.1.2.

### store(AppProxy[], boolean[], String[][])

```
public void store(org.dvb.application.AppProxy[] apps, boolean[] autoStart,
java.lang.String[][] args)
throws InvalidApplicationException, UserRejectedInstallException, NotEnoughResourcesException, Inval
idDescriptionFileException, ApplicationDownloadException
```

Installs several applications into this stored service.

If ApplicationStorageHandler is supported by the implementation and such a handler is registered then the handler will be asked for permission to install the application. If no such handler is asked then the end user of the GEM terminal may be asked for permission. Applications should be prepared for the possibility of a user interface being shown under these circumstances.

NOTE: This method is synchronous and will block until the installation is either completed or fails.

This method either succeeds or fails completely. It will never install some applications but not others.

For each specified application, if that application is already installed in the service (whether the same version or a different version, then the rules specified in the `store()` method that stores a single application shall apply regarding updating the application and/or the stored representation of the AIT for that application. Note that the preceeding statement that this method either completely succeeds or completely fails still applies.

If this service is being presented, then on success the rules in the `store()` method that stores a single application regarding sending `AppsDatabaseEvent`s and starting and killing applications shall apply.

Parameters:

`apps` - an array of AppProxys representing the applications to be installed. May not be `null` or zero-length.

`autoStart` - An array that is the same length as `apps` where each element is `true`, if the corresponding application in `apps` becomes an autostart application in the stored application service; or `false`, if the application becomes a normal present (non-autostart) application. May not be `null`.

`args` - parameters to be available to the applications when started. May be `null`, indicating all applications take no parameters. Otherwise it is an array that is the same length as `apps`, where each element is the arguments for the corresponding application in `apps`. If the element is either null or a subarray of size zero, that indicates no parameters are to be available. These parameters shall be available to applications when running as part of the stored application service using the Xlet property "dvb.installer.parameters".

Throws:

`java.lang.IllegalArgumentException` - If the arrays passed to this method have different lengths, or if they have length zero.

`InvalidApplicationException` - thrown if any of the applications do not include an application_storage_descriptor, or if any of the applications are not identified as able to run stand-alone in their application_storage_desciptor.

`InvalidDescriptionFileException` - thrown if one of the application description files is missing, invalid or otherwise not conformant to the specification.

`NotEnoughResourcesException` - thrown if the GEM terminal does not have enough resources, e.g. storage space, available for the applications.

`UserRejectedInstallException` - thrown if either the end user or an ApplicationStorageHandler rejects the installation.

`ApplicationDownloadException` - thrown if the downloading of the application files was notsuccessful (e.g. a carousel error, a file in the application description file is missing in the carousel, if the application was removed from the AIT or from its transport mechanism while installation is in progress, etc) or if the application failed authentication while being downloaded.

`java.lang.SecurityException` - thrown if the application calling this method does not have `ApplicationStoragePermission` s with action "storeService" for the organisation_ids of all the applications to be stored, and an `ApplicationStoragePermission with action "manageService" for the organisation_id of this service.`

Since:

MHP1.1.2.

### store(AppProxy[], boolean[], String[][], ApplicationStorageListener)

```
public void store(org.dvb.application.AppProxy[] apps, boolean[] autoStart,
java.lang.String[][] args, org.dvb.application.storage.ApplicationStorageListener listener)
```

Installs several applications into this stored service.

If ApplicationStorageHandler is supported by the implementation and such a handler is registered then the handler will be asked for permission to install the application. If no such handler is asked then the end user of the GEM terminal may be asked for permission. Applications should be prepared for the possibility of a user interface being shown under these circumstances.

This method is asynchronous with completion reported via the ApplicationStorageListener.

This method either succeeds or fails completely. It will never install some applications but not others.

For each specified application, if that application is already installed in the service (whether the same version or a different version, then the rules specified in the `store()` method that stores a single application shall apply regarding updating the application and/or the stored representation of the AIT for that application. Note that the preceeding statement that this method either completely succeeds or completely fails still applies.

If this service is being presented, then on success the rules in the `store()` method that stores a single application regarding sending `AppsDatabaseEvent`s and starting and killing applications shall apply.

Parameters:

`apps` - an array of AppProxys representing the applications to be installed. May not be `null` or zero-length.

`autoStart` - An array that is the same length as `apps` where each element is `true`, if the corresponding application in `apps` becomes an autostart application in the stored application service; or `false`, if the application becomes a normal present (non-autostart) application. May not be `null`.

`args` - parameters to be available to the applications when started. May be `null`, indicating all applications take no parameters. Otherwise it is an array that is the same length as `apps`, where each element is the arguments for the corresponding application in `apps`. If the element is either null or a subarray of size zero, that indicates no parameters are to be available. These parameters shall be available to applications when running as part of the stored application service using the Xlet property "dvb.installer.parameters".

`listener` - the listener to be notified of the success or failure of this installation.

Throws:

`java.lang.IllegalArgumentException` - If the arrays passed to this method have different lengths, or if they have length zero.

`java.lang.SecurityException` - thrown if the application calling this method does not have `ApplicationStoragePermission` s with action "storeService" for the organisation_ids of all the applications to be stored, and an `ApplicationStoragePermission with action "manageService" for the organisation_id of this service.`

Since:

MHP1.1.3.

### store(AppProxy, boolean, String[])

```
public void store(org.dvb.application.AppProxy app, boolean autoStart, java.lang.String[] args)
throws InvalidApplicationException, UserRejectedInstallException, NotEnoughResourcesException, Inval
idDescriptionFileException, ApplicationDownloadException
```

Installs an application into this stored service.

If ApplicationStorageHandler is supported by the implementation and such a handler is registered then the handler will be asked for permission to install the application. If no such handler is asked then the end user of the GEM terminal may be asked for permission. Applications should be prepared for the possibility of a user interface being shown under these circumstances.

NOTE: This method is synchronous and will block until the installation is either completed or fails.

Successfully adding an application to a stored service that is currently being presented shall cause the terminal to behave as if the application was added to the AIT - i.e. it will send an `org.dvb.application.AppsDatabaseEvent.APP_ADDED` event to registered listeners, and if the `autoStart` parameter is `true` then the application shall be started.

If the same version of the same application is already installed in this stored service, then (if the caller has sufficient permissions to install the application) this method shall update the stored representation of the AIT from the `autoStart` and `args` parameters, and return without throwing an exception. Terminals shall not store the application again. Terminals are not expected (but are allowed) to prompt the user for permission in this case. If the stored service is currently being presented, and the autostart setting is changed, then:

- An `org.dvb.application.AppsDatabaseEvent.APP_CHANGED` event shall be sent as usual to registered listeners.
- If the autostart setting is changed to `true`, then the application shall be started as usual. (Note that the opposite is not true: setting autostart to `false` shall not cause the application to be killed if it is running).

Note that calling this method with a new value for the `args` parameter shall not have any effect on an existing instance of an application.

If a different version of the same application is already installed in this stored service, then this method shall install the new version of the application as normal. (Note that for the purposes of this method description, "new" and "old" versions refer to the version specified by the AppProxy passed to this method and the version currently installed in the service, respectively. They do not refer to numeric comparison of version numbers). On success, the old version shall be removed from the service. On failure, the old version of the application shall remain as it was before this method was called. If the stored service is currently being presented, then on success:

- An `org.dvb.application.AppsDatabaseEvent.APP_CHANGED` event shall be sent as usual to registered listeners.
- If the old version of the application is running (or paused) then it shall be terminated as if it had been signalled as KILL in the AIT.
- If the new version is signalled as autostart, then the new version shall be started as usual, once the old version has terminated.

Parameters:

`app` - an AppProxy representing the application to be installed.

`autoStart` - true, if the application becomes an autostart application in the stored application service; false, if the application becomes a normal present (non-autostart) application in the stored application service.

`args` - parameters to be available to the application when started. Passing in either null or an array of size zero indicates no parameters are to be available. These parameters shall be available to applications when running as part of the stored application service using the Xlet property "dvb.installer.parameters".

Throws:

`InvalidApplicationException` - thrown if the application does not include an application_storage_descriptor, or if the application is not identified as able to run stand-alone in it's application_storage_desciptor.

`InvalidDescriptionFileException` - thrown if the application description file is missing, invalid or otherwise not conformant to the specification.

`NotEnoughResourcesException` - thrown if the GEM terminal does not have enough resources, e.g. storage space, available for the application.

`UserRejectedInstallException` - thrown if either the end user or an ApplicationStorageHandler rejects the installation.

`ApplicationDownloadException` - thrown if the downloading of the application files was notsuccessful (e.g. a carousel error, a file in the application description file is missing in the carousel, if the application was removed from the AIT or from its transport mechanism while installation is in progress, etc) or if the application failed authentication while being downloaded.

`java.lang.SecurityException` - thrown if the application calling this method does not have an `ApplicationStoragePermission` with action "storeService" for the organisation_id of the application to be stored, and an `ApplicationStoragePermission` with action "manageService" for the organisation_id of this service.

Since:

MHP1.1.2.

### store(Locator, AppID[], boolean[], String[][], ApplicationStorageListener)

```
public void store(org.davic.net.Locator locator, org.dvb.application.AppID[] apps,
boolean[] autoStart, java.lang.String[][] args,
org.dvb.application.storage.ApplicationStorageListener listener)
```

Installs several applications into this stored service. The applications are from a service which need not be selected.

If ApplicationStorageHandler is supported by the implementation and such a handler is registered then the handler will be asked for permission to install the application. If no such handler is asked then the end user of the GEM terminal may be asked for permission. Applications should be prepared for the possibility of a user interface being shown under these circumstances.

This method is asynchronous with completion reported via the ApplicationStorageListener.

This method either succeeds or fails completely. It will never install some applications but not others.

For each specified application, if that application is already installed in the service (whether the same version or a different version, then the rules specified in the `store()` method that stores a single application shall apply regarding updating the application and/or the stored representation of the AIT for that application. Note that the preceeding statement that this method either completely succeeds or completely fails still applies.

If this service is being presented, then on success the rules in the `store()` method that stores a single application regarding sending `AppsDatabaseEvent`s and starting and killing applications shall apply.

Parameters:

`locator` - the locator of the service from which the applications are to be stored.

`apps` - an array of application ids identifying the applications to be stored.

`autoStart` - An array that is the same length as `apps` where each element is `true`, if the corresponding application in `apps` becomes an autostart application in the stored application service; or `false`, if the application becomes a normal present (non-autostart) application. May not be `null`.

`args` - parameters to be available to the applications when started. May be `null`, indicating all applications take no parameters. Otherwise it is an array that is the same length as `apps`, where each element is the arguments for the corresponding application in `apps`. If the element is either null or a subarray of size zero, that indicates no parameters are to be available. These parameters shall be available to applications when running as part of the stored application service using the Xlet property "dvb.installer.parameters".

`listener` - the listener to be notified of the success or failure of this installation.

Throws:

`java.lang.IllegalArgumentException` - If the arrays passed to this method have different lengths, or if they have length zero.

`java.lang.SecurityException` - thrown if the application calling this method does not have `ApplicationStoragePermission` s with action "storeService" for the organisation_ids of all the applications to be stored, and an `ApplicationStoragePermission with action "manageService" for the organisation_id of this service.`

Since:

MHP1.1.3.

# org.dvb.application.storage ApplicationStorageListener

### Declaration

```
public interface ApplicationStorageListener
```

### Description

Listener to receive reports on progress of application storage operations. When multiple applications are being stored, calls to this listener shall be generated for each individual application.

### Methods

#### storageCompleted(AppID)

```
public void storageCompleted(org.dvb.application.AppID id)
```

Called when a storage operation is successfully completed.

Parameters:

`id` - the application which is now successfully stored.

#### storageFailed(AppID, Exception)

```
public void storageFailed(org.dvb.application.AppID id, java.lang.Exception failureMode)
```

Called if a storage operation fails.

Parameters:

`id` - the application whose storage failed.

`failureMode` - the reason for the failure as would be reported by one of the checked exceptions thrown by the method StoredApplicationService.store.

#### storageUpdate(AppID, byte, int)

```
public void storageUpdate(org.dvb.application.AppID id, byte completion, int size)
```

Called at 1 second intervals after a storage operation starts until it completes. The precise definition of completeness is implementation dependent however it would typically reflect the extent to which the data of the application has been downloaded. In implementations where an application is first downloaded and then stored, assuming the storage phase takes much less time than the downloading phase, the storage phase can be ignored when reporting completeness. If an application is (already) downloaded and stored before the first call to this method becomes due, this method may never be called.

Parameters:

`id` - the application being stored.

`size` - the size in bytes of the application to be stored, as calculated by adding the size attributes in the application description file of the application.

`completion` - the extent to which the storage operation is complete, 0 being not complete and 255 being almost complete.

# org.dvb.application.storage ApplicationStoragePermission

### Declaration

```
public class ApplicationStoragePermission extends java.security.Permission

java.lang.Object
  |
  +--java.security.Permission
        |
        +--org.dvb.application.storage.ApplicationStoragePermission
```

### All Implemented Interfaces:

```
java.security.Guard, java.io.Serializable
```

### Description

This class represents a permission to manage applications stored in the GEM terminal. An ApplicationStorage-Permission contains a name representing the organisation_id whose applications can be managed and an actions list representing the permitted actions, e.g. store and/or remove applications.

The name of the permission contains the organisation_id represented in hexadecimal form as defined in the section "Text encoding of application identifiers" in the main body of this specification. Valid organization ids must be in the range `"1"` to `"ffffffff"` inclusive. Alternatively, the value `"*"` indicates all organization ids.

The actions string shall be a comma-separated list of one or more of the following:

- `"manageService"`, representing permission to query what applications are stored in a stored application service with the given organisation ID. This permission is also necessary (but not sufficient) to store applications into and remove applications from a stored application service, where the stored application service has the given organisation ID.
- `"storeService"`, representing permission to store an application in a stored application service, where the application has the given organisation ID.
- `"removeService"`, representing permission to remove an application from a stored application service, where the application has the given organisation ID.
- `"createService"`, representing permission to create a stored application service with a given organisation ID.
- `"deleteService"`, representing permission to remove a stored application service with a given organisation ID.
- `"manageCache"`, representing permission to query what applications are stored in an application cache with the given organisation ID. This permission is also necessary (but not sufficient) to store applications into and remove applications from an application cache, where the application cache has the given organisation ID.
- `"storeCache"`, representing permission to store an application in a cache.
- `"removeCache"`, representing permission to remove an application from a cache.

### Since:

MHP1.1.

### Constructors

### ApplicationStoragePermission(String, String)

```
public ApplicationStoragePermission(java.lang.String name, java.lang.String actions)
```

Creates a new `ApplicationStoragePermission` object with the specified name and actions string. The name contains the organisation_id of the applications that can be managed and the actions String shall be a comma-separated list of actions as defined above. Permission objects constructed with incorrectly encoded parameters do not represent any permission and are ignored by the platform.

Parameters:

`name` - the organisation_id whose applications can be managed. This is encoded in hexadecimal representation as if by `java.lang.Integer.toHexString(int)`, and must be in the range `"1"` to `"ffffffff"` inclusive. Alternatively, the value `"*"` indicates all organization ids.

`actions` - Shall conform to the syntax described above.

Methods

equals(Object)

```
public boolean equals(java.lang.Object obj)
```

Checks two permission objects for equality.

Do not use the `equals` method for making access control decisions; use the `implies` method.

If `X` is an `ApplicationStoragePermission`, and `Y` is any `Object`, then `X.equals(Y)` returns true if and only if all of the following hold:

- `Y` is not `null`.
- `Y` is an instance of `ApplicationStoragePermission`.
- `X` and `Y` have the same run-time type (i.e. `X.getClass() == Y.getClass()`).
- `X` and `Y` were both constructed with correctly encoded parameters.
- `X` and `Y` both have the same organization ID, or both have organization ID `"*"`.
- `X` has the same actions as `Y`. The order of the comma-separated actions list does not affect the results of this check.

Overrides:

`equals` in class `Permission`

Parameters:

`obj` - the object we are testing for equality with this object.

Returns:

`true` if both `ApplicationStoragePermission` objects are equivalent.

getActions()

```
public java.lang.String getActions()
```

Returns the actions as a `String`. Must always return actions in canonical form.

Permission objects constructed with an incorrectly encoded action parameter shall return an empty string.

Permission objects constructed with a correctly encoded action parameter shall return a comma-separated list of actions, with the actions sorted in the order given by `java.lang.String.compareTo(String)`.

Overrides:

`getActions` in class `Permission`

Returns:

the actions of this `Permission`.

### hashCode()

```
public int hashCode()
```

Returns the hash code value for this `ApplicationStoragePermission` object.

The required `hashCode` behavior for `ApplicationStoragePermission` objects is the following:

- Whenever it is invoked on the same ApplicationStoragePermission object more than once during an execution of a Java application, the `hashCode` method must consistently return the same integer. This integer need not remain consistent from one execution of an application to another execution of the same application.
- If two `ApplicationStoragePermission` objects are equal according to the `equals(Object)` method, then calling the `hashCode` method on each of the two Permission objects must produce the same integer result.

Overrides:

`hashCode` in class `Permission`

Returns:

a hash code value for this object.

### implies(Permission)

```
public boolean implies(java.security.Permission permission)
```

Checks if the specified permission's actions are "implied by" this object's actions.

If `X` is an `ApplicationStoragePermission`, and `Y` is any `Permission`, then `X.implies(Y)` returns true if and only if all of the following hold:

- Y is an instance of ApplicationStoragePermission.
- X and Y have the same run-time type (i.e. X.getClass() == Y.getClass()).
- X and Y were both constructed with correctly encoded parameters.
- X and Y both have the same organization ID, or X has the organization ID "*".
- X contains all the actions requested by Y. The order of the comma-separated actions list does not affect the results of this check.

Overrides:

`implies` in class `Permission`

Parameters:

`permission` - the permission to check against.

Returns:

`true` if the specified permission is implied by this object, `false` if not.

### newPermissionCollection()

```
public java.security.PermissionCollection newPermissionCollection()
```

Returns an empty `PermissionCollection` for `ApplicationStoragePermission` objects.

Overrides:

`newPermissionCollection` in class `Permission`

Returns:

a new `PermissionCollection` object for `ApplicationStoragePermissions`.

# org.dvb.application.storage

# ExtendedAppAttributes

## Declaration

```
public interface ExtendedAppAttributes extends org.dvb.application.AppAttributes
```

## All Superinterfaces:

`org.dvb.application.AppAttributes`

## Description

The `ExtendedAppAttributes` interface provides additional attributes that are useful when application can be stored in the GEM terminal.

`org.dvb.application.AppAttributes` objects that are returned from the `org.dvb.application.AppsDatabase` shall implement this interface if and only if the terminal supports storing applications and the total amount of memory for stored services and/or cached applications is greater than zero.

Since:

MHP1.1.

## Methods

### canAddToStoredService()

```
public boolean canAddToStoredService()
```

Returns `true` if this application is signalled as capable of being added to a stored service. I.e. if this application could be added to a stored service without the `InvalidApplicationException` being thrown.

For broadcast applications, returns `true` if and only if this application is signalled as being storable and as capable of running stand alone. For applications that are part of a stored service, this function returns `true`.

Returns:

`true` if this application is signalled as capable of being added to a stored service.

Since:

MHP1.1.2.

### canCache()

```
public boolean canCache()
```

Returns `true` if this application is signalled as capable of being cached. I.e. if this application could be added to a cache without the `InvalidApplicationException` being thrown.

For broadcast applications, returns `true` if and only if this application is signalled as being storable. For applications that are part of a stored service, this function returns `true`.

Returns:

`true` if this application is signalled as capable of being cached.

Since:

MHP1.1.2.

### getCurrentVersionNumber()

```
public int getCurrentVersionNumber()
```

Returns the optional version number currently signalled for this application. (E.g. in the AIT). If no version number is signalled, -1 shall be returned.

Returns:

the version number signalled for this application.

Since:

MHP1.1.

### isStartable()

```
public boolean isStartable()
```

This method determines whether the application is startable or not. An Application is not startable if any of the following apply.

- The application is transmitted on a remote connection, and either does not have an application storage descriptor, is not cached, or is not signalled as launchable completely from cache.
- The caller of the method does not have the Permissions to start it.
- if the application is signalled with a control code which is neither AUTOSTART nor PRESENT.
- if the application is signalled with an application storage descriptor where not_launchable_from_broadcast is "1", and the application is not stored or cached.

If none of the above apply, then the application is startable.

The value returned by this method does not depend on whether the application is actually running or not.

Overrides:

`isStartable` in interface `AppAttributes`

Returns:

true if an application is startable, false otherwise.

Since:

MHP1.0.

### isStorageRequired()

```
public boolean isStorageRequired()
```

Returns `true` if this application is signalled as not launchable from broadcast. I.e. for a broadcast application this method will return `true` if and only if the application is signalled with an application storage descriptor where not_launchable_from_broadcast is "1". If this application is part of a stored service, this function returns `true`.

Note that the return value of this method does not depend on whether or not the application is currently stored or cached.

Returns:

`true` if and only if this application is signalled as not launchable from broadcast.

Since:

MHP1.1.2.

### isStored()

```
public boolean isStored()
```

Returns `true` if this application is currently stored or cached on the terminal. Returns `true` if and only if the application is signalled as being storable and all the files signalled as having critical priority have been stored or cached by the terminal.

Returns:

`true` if this application is currently stored or cached.

Since:

MHP1.1.2.

# org.dvb.application.storage InvalidApplicationException

### Declaration

```
public class InvalidApplicationException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.application.storage.InvalidApplicationException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

When an application is being installed into a service, this is thrown when:

- The application does not include an application_storage_descriptor.
- The application is not identified as able to run stand-alone in its application_storage_descriptor.

When an application is being stored in a cache this is thrown when:

- The application does not include an application_storage_descriptor.
- The application is not signalled as part of the same service as the calling application.

Since:

MHP1.1.

### Constructors

### InvalidApplicationException()

```
public InvalidApplicationException()
```

Constructs an `InvalidApplicationException` with no detail message.

### InvalidApplicationException(String)

```
public InvalidApplicationException(java.lang.String s)
```

Construct a `InvalidApplicationException` with a detail message.

Parameters:

`s` - detail message.

# org.dvb.application.storage InvalidDescriptionFileException

### Declaration

```
public class InvalidDescriptionFileException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.application.storage.InvalidDescriptionFileException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown when an application descriptio file is invalid.

### Constructors

### InvalidDescriptionFileException()

```
public InvalidDescriptionFileException()
```

Creates a new instance of `InvalidDescriptionFileException` without detail message.

### InvalidDescriptionFileException(String)

```
public InvalidDescriptionFileException(java.lang.String msg)
```

Constructs an instance of `InvalidDescriptionFileException` with the specified detail message.

Parameters:

`msg` - the detail message.

# org.dvb.application.storage NotEnoughResourcesException

### Declaration

```
public class NotEnoughResourcesException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.application.storage.NotEnoughResourcesException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown if the GEM terminal does not have enough resources to install the application. E.g. not enough storage space.

### Since:

MHP1.1.

### Constructors

#### NotEnoughResourcesException()

```
public NotEnoughResourcesException()
```

Constructs a `NotEnoughResourcesException` with no detail message.

#### NotEnoughResourcesException(String)

```
public NotEnoughResourcesException(java.lang.String s)
```

Constructs a `NotEnoughResourcesException` with a detail message

### Parameters:

`s` - detail message.

# org.dvb.application.storage ServiceAlreadyExistsException

### Declaration

```
public class ServiceAlreadyExistsException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.application.storage.ServiceAlreadyExistsException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown if a StoredApplicationService already exists with the identifiers the application tried to create a new one.

### Since:

MHP1.1.

### Constructors

#### ServiceAlreadyExistsException()

```
public ServiceAlreadyExistsException()
```

Constructs a `ServiceAlreadyExistsException` with no detail message.

#### ServiceAlreadyExistsException(String)

```
public ServiceAlreadyExistsException(java.lang.String s)
```

Constructs a `ServiceAlreadyExistsException` with a detail message.

### Parameters:

`s` - detail message.

# org.dvb.application.storage

# StoredApplicationService

### Declaration

```
public interface StoredApplicationService extends javax.tv.service.Service,
ApplicationStorageController
```

### All Superinterfaces:

```
ApplicationStorageController, javax.tv.service.Service
```

### Description

Defines the information about a stored application service.

Stored application services can be created by applications that have a permission to store applications for an organisation_id.

Stored application services are uniquely identified within the terminal by the combination of the organisation_id and service_id.

### Since:

MHP1.1.

### Methods

#### getLocator()

```
public javax.tv.locator.Locator getLocator()
```

Gets the locator for this stored application service. This locator is opaque and platform specific. It shall be an instance of `org.davic.net.Locator` or a subclass. It is not required to be an instance of any other public locator class in the platform (e.g. `org.davic.net.dvb.DvbLocator` ).

### Overrides:

`getLocator` in interface `Service`

Returns:

a locator for this stored application service.

### getName()

```
public java.lang.String getName()
```

Returns the name of the stored application service as defined when the stored application service was created.

Overrides:

`getName` in interface `Service`

Returns:

the name of the stored application service.

### getOrganisationId()

```
public int getOrganisationId()
```

Return the organisation id of this stored application service.

Returns:

the organisation id of this stored application service.

### getServiceId()

```
public int getServiceId()
```

Return the service id of this stored application service.

Returns:

the service id of this stored application service.

### getServiceInformationType()

```
public javax.tv.service.ServiceInformationType getServiceInformationType()
```

Returns the service information format of this object. This shall always return `ServiceInformationType.UNKNOWN`.

Returns:

the service information format.

### getServiceType()

```
public javax.tv.service.ServiceType getServiceType()
```

Returns the type of this service. For stored application services, this method shall always return `StoredApplicationServiceType.STORED_APPLICATION_SERVICE`.

Overrides:

`getServiceType` in interface `Service`

Returns:

service type of this service.

### hasMultipleInstances()

```
public boolean hasMultipleInstances()
```

Indicates whether the service represented by this Service is available on multiple transports. For stored application services, this shall always return false.

Overrides:

`hasMultipleInstances` in interface `Service`

Returns:

false.

isSelectable()

```
public boolean isSelectable()
```

Test whether this service is selectable. StoredApplicationServices are selectable if they contain at least one autostart application and are not selectable if they do not contain any autostart applications. Applications which offer the end-user a choice of services to select should not include services which are not selectable in that list.

Returns:

false if the service is not selectable otherwise true.

Since:

MHP 1.1.3.

removeService()

```
public void removeService()
throws UserRejectedInstallException
```

Remove the whole stored application service from the terminal. Removal of the service results in the removal of all the applications from within that service.

If there are any applications installed in the stored application service, then the application calling this method should be prepared for the platform consulting the end user of the GEM terminal for permission to remove the stored service.

The platform shall not prompt the user for permission if no applications are installed in the service.

If the end user is asked and does not give permission to remove the service, none of the applications in the service shall be removed.

Throws:

`java.lang.SecurityException` - if the caller does not have an `ApplicationStoragePermission` with action "deleteService" and an organisation_id which matches that of this service. Also thrown if the caller does not have ApplicationStoragePermission with action "removeService" for the organisation IDs of every application installed in this service.

`UserRejectedInstallException` - If the user chose not to remove the applications in the service.

retrieveDetails(SIRequestor)

```
public javax.tv.service.SIRequest retrieveDetails(javax.tv.service.SIRequestor requestor)
```

This method retrieves additional information about the service. This shall result in failure with the `SIRequestFailure-Type DATA_UNAVAILABLE`.

Overrides:

`retrieveDetails` in interface `Service`

Parameters:

`requestor` - The SIRequestor to be notified when this operation completes.

Returns:

A SIRequest object identifying this request.

# org.dvb.application.storage StoredApplicationServiceFactory

## Declaration

```
public abstract class StoredApplicationServiceFactory

java.lang.Object
  |
  +--org.dvb.application.storage.StoredApplicationServiceFactory
```

## Description

This factory creates new Service objects representing stand-alone stored application services. Services thus created shall appear in the list of services maintained by the SIManager until removed using `StoredApplicationService.remove` or some GEM terminal specific mechanism. i.e. they shall be returned by `filterServices` both when passed an instance of `ServiceTypeFilter` constructed with the type `StoredApplicationServiceType.STORED_APPLICATION_SERVICE` and when passed null to list all known services.

### Since:

MHP1.1.2.

## Constructors

### StoredApplicationServiceFactory()

```
protected StoredApplicationServiceFactory()
```

This constructor is provided for the use of implementations and specifications which extend GEM. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

## Methods

### createStoredApplicationService(int, int, String)

```
public abstract org.dvb.application.storage.StoredApplicationService
createStoredApplicationService(int organisation_id, int service_id, java.lang.String serviceName)
throws ServiceAlreadyExistsException
```

Creates a new stored application service.

### Parameters:

`organisation_id` - the organisation_id of the organisation to whom this service belongs to

`service_id` - unique identifier for this service within the organisation

`serviceName` - a name for the service that can be displayed to the end user to identify this service

### Returns:

the stored application service created.

### Throws:

`serviceAlreadyExistsException` - thrown if a stored application service with the same organisation_id and service_id already exists in the terminal.

`java.lang.SecurityException` - thrown if the application calling this method does not have an Application-StoragePermission with action "createService" for the organisation_id passed in as the parameter.

### getInstance()

```
public static org.dvb.application.storage.StoredApplicationServiceFactory getInstance()
```

Get the singleton instance of this class, or null if and only if this GEM implementation does not support stand-alone stored applications.

#### Returns:

a factory.

# org.dvb.application.storage StoredApplicationServiceType

## Declaration

```
public class StoredApplicationServiceType extends javax.tv.service.ServiceType
```

```
java.lang.Object
  |
  +--javax.tv.service.ServiceType
        |
        +--org.dvb.application.storage.StoredApplicationServiceType
```

## Description

The service type used for stored application services.

#### Since:

MHP 1.1.2.

#### See Also:

`javax.tv.service.navigation.ServiceTypeFilter`

## Fields

### STORED_APPLICATION_SERVICE

```
public static final javax.tv.service.ServiceType STORED_APPLICATION_SERVICE
```

The service type for a stored application.

#### Since:

MHP 1.1.2.

## Constructors

### StoredApplicationServiceType(String)

```
protected StoredApplicationServiceType(java.lang.String name)
```

Creates a new StoredServiceType object.

#### Parameters:

`name` - The string name of this type (e.g. "STORED_APPLICATION_SERVICE").

#### Since:

MHP 1.1.2.

# org.dvb.application.storage UserRejectedInstallException

### Declaration

```
public class UserRejectedInstallException extends java.lang.Exception

java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.application.storage.UserRejectedInstallException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown if the request to install an application was rejected. This includes both rejection by the end user and rejection by an ApplicationStorageHandler.

### Since:

MHP1.1.

### Constructors

#### UserRejectedInstallException()

```
public UserRejectedInstallException()
```

Constructs a `UserRejectedInstallException` with no detail message.

#### UserRejectedInstallException(String)

```
public UserRejectedInstallException(java.lang.String s)
```

Constructs a `UserRejectedInstallException` with a detail message

### Parameters:

`s` - detail message.

# Annex AH (normative): Internet client APIs

# Package org.dvb.internet

## Description

Provides a mechanism for GEM applications to control internet clients that may be present on a GEM such as a web browser or email client.

This package provides a mechanism for GEM applications to control internet clients that may be present on a GEM such as a web browser or email client. The API makes no assumptions about whether an internet client and downloaded GEM application can run simultaneously, and in some cases starting an internet client may result in the calling application being killed.

The API consists of two main class hierarchies. `InternetClientService` and its clients provide a mechanism to get a locator for a resident internet client so that it can be started, and also provide any functionality that does not require the client to be started. This can include adding bookmarks or getting the capabilities of the clients. The second major part of the API is the `InternetClient` class and its subclasses. These provide an interface to a running instance of an internet client and support the operations that can only be carried out when the client is running.

The internet client API uses the JavaTV service navigation API for accessing resident applications. Any effect that the invocation of a resident internet client has on the lifecycle of a downloaded application is dependent on the service context in which the resident application is started (using `ServiceContext.select()` ). If the internet client is started in the same service context as the downloaded application, then the downloaded application will be terminated, following the usual semantics of the service selection API. The basic concepts are introduced below and then a few scenarios are explored to illustrate combinations or broadcast application versus resident application execution.

## Concepts

The first implication of resident applications is that it should be feasible for broadcast applications to isolate, browse, and select such services. The collection of `javax.tv.service.*` packages provides the framework for the solution to this requirement.

The class `ServiceType` in the `javax.tv.service` package defines various kinds of services. This is extended in this API to provide additional service types for Internet clients. The `javax.tv` framework lets a client filter the services known to a platform to create a `ServiceCollection`. The client would first create an instance of `ServiceType-Filter`.

```
public ServiceTypeFilter(ServiceType type)
```

where the `ServiceType` in the signature would be the instance for the resident service in question. The client would then filter the services known to the platform to isolate the resident services of this type. The `ServiceCollection` interface provides the filter operation.

```
public ServiceCollection createServiceCollection(ServiceFilter filter)
```

Given the `ServiceCollection`, the client can iterate over the resident services to find a specific resident service. The last step is to `select()` the resident service. The `ServiceContext` interface of the `javax.tv.service.selection` package provides the operation:

```
public void select(Service selection)
throws java.lang.SecurityException
```

The select() method is asynchronous and returns before attempting to allocate resources to execute the service.

If the implementation determines that it can not realize the service due to scarce resources, it forwards a `SelectionFailedEvent` (of the `javax.tv.service. selection` package) with the reason code `INSUFFICIENT_RESOURCES`. The `ServiceContextListener` interface fields the event:

```
public void receiveServiceContextEvent(ServiceContextEvent e)
```

Thus the client should implement an instance of the `ServiceContextListener` interface and register interest in service context events. The `ServiceContext` interface provides the mechanism:

```
public void addListener(ServiceContextListener listener)
throws java.lang.IllegalStateException
```

The semantics of a `ServiceContext` are that at most a single `Service` can execute within a `ServiceContext` at a time. Since the premise is that a broadcast service selects a resident service, there are two scenarios of interest.

The first scenario presumes that the broadcast service wants to select the resident service but not survive. It expects the selection to cause its termination. The broadcast service in this case should create an instance of `ServiceContextFactory`. The operation on `ServiceContextFactory` itself is:

```
public static ServiceContextFactory getInstance()
```

The broadcast application then requests the service contexts that are visible to the broadcast application:

```
public abstract ServiceContext getServiceContext(XletContext ctx)
```

which returns *its* `ServiceContext`. The broadcast application selects the resident application within its service context:

```
public void select(Locator selection)
throws InvalidLocatorException, java.lang.SecurityException,
java.lang.IllegalStateException
```

which causes the broadcast application to terminate.

The premise of the second scenario is that the broadcast application wants to survive the selection of the resident service. In this case the broadcast again creates a `ServiceContextFactory` but then creates a `ServiceContext` which is not its `ServiceContext`:

```
public abstract ServiceContext createServiceContext()
throws InsufficientResourcesException,
java.lang.SecurityException
```

The broadcast application selects the resident application. Since the operation is on a `ServiceContext` that is not its `ServiceContext`, the broadcast application survives the selection.

One subtle aspect of the invocation patterns is that the platform does not know whether the broadcast application will attempt to select a resident application versus another broadcast application until the broadcast application invokes the `select()` operation. For example, assume the platform supports a single broadcast application plus a single resident application. Further assume that, after the broadcast application creates a second `ServiceContext`, it selects a second broadcast application rather than a resident application. If the locator is valid, the operation succeeds, but the broadcast application then receives a `SelectionFailedEvent`.

Likewise assume the platform supports multiple broadcast applications but not (for the present) a resident application. If the broadcast application selects a resident application, and if the locator is valid, the operation succeeds, but the broadcast application then receives a `SelectionFailedEvent`. (The second case is rather improbable. If the platform does not support resident applications, it is not clear how the broadcast application could obtain a valid locator to a broadcast application. The platform would have to support resident applications in concept, but not at the instant the broadcast applications selects a resident application.)

| Class Summary | |
|---|---|
| **Interfaces** | |
| EmailClient | Interface supporting the operations required on an email client. |
| EmailClientService | Service representing the resident email client |
| InternetClient | Base interface for the internet clients. |
| InternetClientListener | Interface for objects that wish to receive `InternetClientEvents` |
| InternetClientService | The base class for the interface to resident applications that are supported by the internet access profile. |
| UsenetClient | This interface supports the operations required on a Usenet news client. |
| UsenetClientService | Service representing the resident usenet news client |
| WWWBrowser | This interface provides support for the operations required on an WWW browser. |
| WWWBrowserService | Service representing a resident WWW browser |
| **Classes** | |
| CancelledByUserEvent | Event indicating that an operation on an internet client failed because the operation was canceled by the user |
| HomePagePermission | This class is for permissions related to the ability for a GEM application to set the home page of a WWW browser in the internet access profile. |
| InternetClientEvent | Base class for all status events from internet clients. |
| InternetClientFailureEvent | Event indicating that an operation on an internet client failed. |
| InternetClientSuccessEvent | Event indicating that an operation on an internet client succeeded |
| InternetServiceFilter | `InternetServiceFilter` represents a service type for a particular kind of internet client. |
| InternetServiceType | Class representing the additional service types available in the Internet Access profile. |
| PermissionDeniedEvent | Event indicating that an operation on an internet client failed due to the specified URL not being accessible by the client due to access controls on the server. |
| URLUnavailableEvent | Event indicating that an operation on an internet client failed due to the specified URL not being available. |
| **Exceptions** | |
| ClientNotRunningException | Exception thrown when a method of `InternetClient` or one of its subclasses is called while the client is not running, for instance if the client has been terminated by the user. |
| EntryExistsException | Exception generated when an application tries to add a bookmark or address book entry that already exists |

# org.dvb.internet

# CancelledByUserEvent

### Declaration

```
public class CancelledByUserEvent extends InternetClientFailureEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.internet.InternetClientEvent
              |
              +--org.dvb.internet.InternetClientFailureEvent
                    |
                    +--org.dvb.internet.CancelledByUserEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Event indicating that an operation on an internet client failed because the operation was canceled by the user.

### Constructors

#### CancelledByUserEvent(Object, URL)

```
public CancelledByUserEvent(java.lang.Object source, java.net.URL url)
```

Construct a new `CancelledByUserEvent`.

Parameters:

`source` - the source of the event.

`url` - the URL to which the event relates.

# org.dvb.internet ClientNotRunningException

### Declaration

```
public class ClientNotRunningException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.internet.ClientNotRunningException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Exception thrown when a method of `InternetClient` or one of its subclasses is called while the client is not running, for instance if the client has been terminated by the user.

### Constructors

#### ClientNotRunningException()

```
public ClientNotRunningException()
```

Construct a `ClientNotRunningException` with no detail message.

#### ClientNotRunningException(String)

```
public ClientNotRunningException(java.lang.String reason)
```

Construct a `ClientNotRunningException` with the specified detail message.

Parameters:

`reason` - the reason why the exception was thrown.

# org.dvb.internet EmailClient

### Declaration

```
public interface EmailClient extends InternetClient
```

All Superinterfaces:

InternetClient, javax.tv.service.selection.ServiceContentHandler

Description

Interface supporting the operations required on an email client.

Methods

createMessage(String, String, String)

```
public void createMessage(java.lang.String to, java.lang.String subject,
java.lang.String messageBody)
throws ClientNotRunningException
```

Create a new email message. If any of the parameters are an empty string, the user will be prompted for the missing information.

This is an asynchronous operation, whose success or failure will be indicated by an InternetClientSuccessEvent or InternetClientFailureEvent or one of their subclasses.

Parameters:

to - the address to which the email should be sent.

subject - the subject for the email.

messageBody - the body of the message.

Throws:

java.lang.NullPointerException - if any of the to address, subject or message body are null.

ClientNotRunningException - if the client is not currently running.

java.lang.IllegalArgumentException - if the destination address is an empty string.

# org.dvb.internet

# EmailClientService

Declaration

```
public interface EmailClientService extends InternetClientService
```

All Superinterfaces:

InternetClientService, javax.tv.service.Service

Description

Service representing the resident email client.

Methods

addToAddressBook(String, String)

```
public void addToAddressBook(java.lang.String address, java.lang.String name)
throws EntryExistsException, IOException
```

Add an entry to the address book. As a side-effect an entry previously added by this method may be lost. Implementations may restrict the number of entries a single GEM application or source of applications may add.

Parameters:

address - the address to be added to the address book.

`name` - the name that should be associated with that address.

Throws:

`EntryExistsException` - if an entry with both the same name and the same address already exists in the address book.

`java.lang.IllegalArgumentException` - if the string passed as the email address does not conform to the internet email address format.

IOException - if no more address book entries can be added due to a lack of storage space or a limitation in the client.

`java.io.IOException`.

### getUserEmailAddress()

```
public java.lang.String getUserEmailAddress()
```

Get the email address of the user. The returned value shall be the same as that obtained by reading the value of the "User @" preference (see org.dvb.user.GeneralPreferences for details).

Returns:

the email address of the user, or null if no email address is set or the email address is not available to the client.

Throws:

`java.lang.SecurityException` - if the caller does not have a `UserPreferencePermission` with the name "read".

### setInitialMessage(String, String, String)

```
public void setInitialMessage(java.lang.String to, java.lang.String subject,
java.lang.String messageBody)
```

Set the initial recipient, subject and& message body to be used when the email client starts. This URL is specific to this instance of EmailClientService and will not impact any other instance and is only valid for the lifetime of this instance. Calling this method and then selecting the EmailClientService instance is equivalent to selecting the EmailClientService instance, obtaining the EmailClient and then calling the createMessage method there. If the application calling this method is still running when the message is sent (or fails) and has registered an InternetClientListener then the appropriate InternetClientEvent shall be sent corresponding to the success or failure of the operation to send the message.

Parameters:

`to` - the address to which the email should be sent.

`subject` - the subject for the email.

`messageBody` - the body of the message.

Throws:

`java.lang.NullPointerException` - if any of the to address, subject or message body are null.

`java.lang.IllegalArgumentException` - if the destination address is an empty string.

# org.dvb.internet EntryExistsException

### Declaration

```
public class EntryExistsException extends java.lang.Exception

java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.internet.EntryExistsException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Exception generated when an application tries to add a bookmark or address book entry that already exists.

### Constructors

#### EntryExistsException()

```
public EntryExistsException()
```

Construct a `EntryExistsException` with no detail message.

#### EntryExistsException(String)

```
public EntryExistsException(java.lang.String message)
```

Construct a `EntryExistsException` with the specified detail message.

### Parameters:

`message` - the reason why the exception was thrown.

# org.dvb.internet HomePagePermission

### Declaration

```
public class HomePagePermission extends java.security.BasicPermission

java.lang.Object
  |
  +--java.security.Permission
        |
        +--java.security.BasicPermission
              |
              +--org.dvb.internet.HomePagePermission
```

### All Implemented Interfaces:

```
java.security.Guard, java.io.Serializable
```

### Description

This class is for permissions related to the ability for a GEM application to set the home page of a WWW browser in the internet access profile. If an application has this permission then shall be able to set the home page.

### Constructors

#### HomePagePermission(String)

```
public HomePagePermission(java.lang.String name)
```

Creates a new `HomePagePermission`. The name parameter is not used and must be set to empty string "". Implementations of this version of GEM shall ignore the name, but it could be taken into use in future versions of GEM.

Parameters:

`name` - the name of the `HomePagePermission`. Not used, shall be "".

#### HomePagePermission(String, String)

```
public HomePagePermission(java.lang.String name, java.lang.String actions)
```

Creates a new `HomePagePermission`. The name and actions parameters are not used. Implementations of this version of GEM shall ignore the name and actions, but they could be taken into use in future versions of GEM. This constructor exists for use by the Policy object to instantiate new Permission objects.

Parameters:

`name` - the name of the `HomePagePermission`. Not used, shall be "".

`actions` - Not used, shall be null.

# org.dvb.internet

# InternetClient

### Declaration

```
public interface InternetClient extends javax.tv.service.selection.ServiceContentHandler
```

### All Superinterfaces:

```
javax.tv.service.selection.ServiceContentHandler
```

### All Known Subinterfaces:

`EmailClient`, `UsenetClient`, `WWWBrowser`

### Description

Base interface for the internet clients. Access to those methods common to all running instances of the client (e.g. operations on bookmark lists) are all carried out through the service objects associated with the `InternetClient` object in question. These are accessed using the `getService()` method.

### Methods

#### addInternetClientListener(InternetClientListener)

```
public void addInternetClientListener(org.dvb.internet.InternetClientListener l)
```

Add a listener for `InternetClientEvents`. If the listener is already registered, or the client is not running, then calls to this method have no effect.

Parameters:

`l` - the listener to be added.

### getService()

```
public org.dvb.internet.InternetClientService getService()
```

Get the service object which matches this internet client. In the case of a web browser, for example, this would be an instance of the WWWBrowserService class.

#### Returns:

the service which matches the `InternetClient` object.

### getServiceContentLocators()

```
public javax.tv.locator.Locator[] getServiceContentLocators()
```

Reports the portions of the service on which this handler operates.

#### Overrides:

`getServiceContentLocators` in interface `ServiceContentHandler`

#### Returns:

An array of length 1, containing the locator representing the internet client. This shall be the same locator returned by calls to `getService().getLocator()`.

### removeInternetClientListener(InternetClientListener)

```
public void removeInternetClientListener(org.dvb.internet.InternetClientListener l)
```

Remove a listener for `InternetClientEvents`. If the listener is not registered, or the client is not running, then calls to this method have no effect.

#### Parameters:

`l` - the listener to be added.

# org.dvb.internet

# InternetClientEvent

#### Declaration

```
public class InternetClientEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.internet.InternetClientEvent
```

#### All Implemented Interfaces:

```
java.io.Serializable
```

#### Direct Known Subclasses:

```
InternetClientFailureEvent, InternetClientSuccessEvent
```

#### Description

Base class for all status events from internet clients.

### Constructors

#### InternetClientEvent(Object, URL)

```
public InternetClientEvent(java.lang.Object source, java.net.URL url)
```

Construct a new `InternetClientEvent`.

#### Parameters:

`source` - the source of the event.

`url` - the URL to which the event relates.

### Methods

#### getUrl()

```
public java.net.URL getUrl()
```

Get the URL for which this event was generated. In the case that a URL was not specified when the event was constructed, null shall be returned.

#### Returns:

an instance of `java.net.URL` or null.

# org.dvb.internet

# InternetClientFailureEvent

### Declaration

```
public class InternetClientFailureEvent extends InternetClientEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.internet.InternetClientEvent
              |
              +--org.dvb.internet.InternetClientFailureEvent
```

### All Implemented Interfaces:

`java.io.Serializable`

### Direct Known Subclasses:

`CancelledByUserEvent`, `PermissionDeniedEvent`, `URLUnavailableEvent`

### Description

Event indicating that an operation on an internet client failed. Typically, internet clients will post subclasses of this event detailing the reason why the operation failed.

### Constructors

#### InternetClientFailureEvent(Object, URL)

```
public InternetClientFailureEvent(java.lang.Object source, java.net.URL url)
```

Construct a new `InternetClientFailureEvent`

Parameters:

`source` - the source of the event.

`url` - the URL to which the event relates.

# org.dvb.internet InternetClientListener

Declaration

```
public interface InternetClientListener extends java.util.EventListener
```

All Superinterfaces:

`java.util.EventListener`

Description

Interface for objects that wish to receive `InternetClientEvents`.

Methods

receiveInternetClientEvent(InternetClientEvent)

```
public void receiveInternetClientEvent(org.dvb.internet.InternetClientEvent event)
```

The method to be called when an `InternetClientEvent` is received.

Parameters:

`event` - the received `InternetClientEvent`

# org.dvb.internet InternetClientService

Declaration

```
public interface InternetClientService extends javax.tv.service.Service
```

All Superinterfaces:

`javax.tv.service.Service`

All Known Subinterfaces:

`EmailClientService`, `UsenetClientService`, `WWWBrowserService`

Description

The base class for the interface to resident applications that are supported by the internet access profile.

The lifecycle of an application which implements this interface or its subclasses is for a broadcast service. The application is started by selecting the appropriate service (using the `Locator` object returned by calls to `getLocator()`). If this service is selected in the service context which contains the executing application, any currently presented content will be stopped and the application will be destroyed before the client is launched. Calling `destroy()` or `stop()` on the service context in which the client is running will cause the client to be terminated.

Methods in this API will not affect the lifecycle of the calling application.

### Methods

#### canRunApplication()

```
public boolean canRunApplication()
```

Returns true if the application can run without having to stop the downloaded GEM application.

**Returns:**

true if the application can be run without stopping the calling application, or false otherwise.

#### getName()

```
public java.lang.String getName()
```

Returns a short service name or an acronym. In the case of subclasses of `InternetClient`, the returned value is implementation dependent

**Overrides:**

`getName` in interface `Service`

**Returns:**

A string representing this service's short name.

#### getServiceType()

```
public javax.tv.service.ServiceType getServiceType()
```

Returns the type of this service. In the case of internet clients, one of the service types defined in the `InternetServiceType` class shall be returned.

**Overrides:**

`getServiceType` in interface `Service`

**Returns:**

The service type of this Service.

#### getSupportedClientServices()

```
public org.dvb.internet.InternetClientService[] getSupportedClientServices()
```

Returns all `InternetClientService`s supported by the same application as this one. This `InternetClientService` is included in the array.

**Returns:**

an array of `InternetClientServices`

#### hasMultipleInstances()

```
public boolean hasMultipleInstances()
```

This method indicates whether the service represented by this Service is available on multiple transports. This method has no effect in the case of an `InternetClient`

**Overrides:**

`hasMultipleInstances` in interface `Service`

**Returns:**

FALSE always for `InternetClient` instances.

### retrieveDetails(SIRequestor)

```
public javax.tv.service.SIRequest retrieveDetails(javax.tv.service.SIRequestor requestor)
```

This method will always fail when called for an `InternetClient`. The requestor will always be notified of a failure of type DATA_UNAVAILABLE.

#### Overrides:

`retrieveDetails` in interface `Service`

#### Parameters:

`requestor` - - The `SIRequestor` to be notified when this retrieval operation completes.

#### Returns:

An `SIRequest` object identifying the request.

# org.dvb.internet

# InternetClientSuccessEvent

### Declaration

```
public class InternetClientSuccessEvent extends InternetClientEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.internet.InternetClientEvent
              |
              +--org.dvb.internet.InternetClientSuccessEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Event indicating that an operation on an internet client succeeded.

### Constructors

### InternetClientSuccessEvent(Object, URL)

```
public InternetClientSuccessEvent(java.lang.Object source, java.net.URL url)
```

Construct a new `InternetClientSuccessEvent`

#### Parameters:

`source` - the source of the event.

`url` - the URL to which the event relates.

# org.dvb.internet

# InternetServiceFilter

### Declaration

```
public final class InternetServiceFilter extends javax.tv.service.navigation.ServiceFilter

java.lang.Object
  |
  +--javax.tv.service.navigation.ServiceFilter
        |
        +--org.dvb.internet.InternetServiceFilter
```

### Description

`InternetServiceFilter` represents a service type for a particular kind of internet client. A `ServiceList` resulting from this filter will include only services providing access to the specified type of internet client.

### Fields

#### EMAIL_CLIENT

```
public static final int EMAIL_CLIENT
```

Constant identifying an email client service.

#### NEWS_CLIENT

```
public static final int NEWS_CLIENT
```

Constant identifying a usenet news client service.

#### WWW_CLIENT

```
public static final int WWW_CLIENT
```

Constant identifying a WWW client service.

### Constructors

#### InternetServiceFilter(int)

```
public InternetServiceFilter(int service_type)
```

Constructs the filter based on a particular type of internet client service. The types of service required are those defined by the constants in this class. Support for other values is platform dependent. Platforms not supporting services of the type specified shall return an empty `ServiceList` when instances of this class constructed using that type are used.

**Parameters:**

`service_type` - the type of service required.

### Methods

#### accept(Service)

```
public boolean accept(javax.tv.service.Service service)
```

Tests if a particular service represents an internet client of the type specified in the constructor of this instance.

**Overrides:**

`accept` in class `ServiceFilter`

**Parameters:**

`service` - A Service to be evaluated against the filtering algorithm.

Returns:

true if service satisfies the filtering algorithm; false otherwise.

# org.dvb.internet

# InternetServiceType

## Declaration

```
public class InternetServiceType extends javax.tv.service.ServiceType
```

```
java.lang.Object
  |
  +--javax.tv.service.ServiceType
        |
        +--org.dvb.internet.InternetServiceType
```

## Description

Class representing the additional service types available in the Internet Access profile. When this service type is used in a `java.tv.service.navigation.ServiceTypeFilter` for filtering available services on their type, all internet client services shall be returned. Applications wishing to obtain a specific sub-type of internet client service should use `InternetServiceFilter`.

See Also:

`InternetServiceFilter`

## Fields

### INTERNET_CLIENT

```
public static final javax.tv.service.ServiceType INTERNET_CLIENT
```

WWW service type.

## Constructors

### InternetServiceType(String)

```
protected InternetServiceType(java.lang.String name)
```

This protected constructor is provided for implementation use and to enable future evolution of this or other specifications. It shall not be called by inter-operable applications.

Parameters:

`name` - the name of the service type.

# org.dvb.internet PermissionDeniedEvent

### Declaration

```
public class PermissionDeniedEvent extends InternetClientFailureEvent

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.internet.InternetClientEvent
              |
              +--org.dvb.internet.InternetClientFailureEvent
                    |
                    +--org.dvb.internet.PermissionDeniedEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Event indicating that an operation on an internet client failed due to the specified URL not being accessible by the client due to access controls on the server.

### Constructors

#### PermissionDeniedEvent(Object, URL)

```
public PermissionDeniedEvent(java.lang.Object source, java.net.URL url)
```

Construct a new `PermissionDeniedEvent`.

### Parameters:

`source` - the source of the event.

`url` - the URL to which the event relates.

# org.dvb.internet URLUnavailableEvent

### Declaration

```
public class URLUnavailableEvent extends InternetClientFailureEvent

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.internet.InternetClientEvent
              |
              +--org.dvb.internet.InternetClientFailureEvent
                    |
                    +--org.dvb.internet.URLUnavailableEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Event indicating that an operation on an internet client failed due to the specified URL not being available. This could be because a server is not available or because a file is not available on that server.

### Constructors

#### URLUnavailableEvent(Object, URL)

```
public URLUnavailableEvent(java.lang.Object source, java.net.URL url)
```

Construct a new `URLUnavailableEvent`.

#### Parameters:

`source` - the source of the event.

`url` - the URL to which the event relates.

# org.dvb.internet

# UsenetClient

### Declaration

```
public interface UsenetClient extends InternetClient
```

### All Superinterfaces:

`InternetClient`, `javax.tv.service.selection.ServiceContentHandler`

### Description

This interface supports the operations required on a Usenet news client.

Any URLs passed to methods in this interface should correspond to the usenet news URL format specified in RFC 1738 [61].

### Methods

#### selectGroup(URL)

```
public void selectGroup(java.net.URL group)
throws ClientNotRunningException
```

Select and display messages in the specified newsgroup. If the news client is not running, then this method will not cause it to be started and the call will fail.

This is an asynchronous operation, whose success or failure will be indicated by an `InternetClientSuccessEvent` or `InternetClientFailureEvent` or one of their subclasses.

#### Parameters:

`group` - the URL of the group. This may or may not include the address of a news server.

#### Throws:

`java.lang.SecurityException` - if the caller does not have a `SocketPermission` for the host part of the specified URL.

`java.lang.IllegalArgumentException` - if the specified URL does not correspond to the Usenet news URL format specified in RFC 1738 [61].

`ClientNotRunningException` - if the client is not currently running.

### selectMessage(URL)

```
public void selectMessage(java.net.URL message)
throws ClientNotRunningException
```

Select and display a message with the given message ID. If the news client is not running, then this method will not cause it to be started and the call will fail.

This is an asynchronous operation, whose success or failure will be indicated by an `InternetClientSuccessEvent` or `InternetClientFailureEvent` or one of their subclasses.

#### Parameters:

`message` - the URL of the message. This may or may not include the address of a news server.

#### Throws:

`java.lang.SecurityException` - if the caller does not have a `SocketPermission` for the host part of the specified URL

`java.lang.IllegalArgumentException` - if the specified URL does not include a Usenet news message ID or does not correspond to the Usenet news URL format specified in RFC 1738 [61].

`ClientNotRunningException` - if the client is not currently running.

# org.dvb.internet

# UsenetClientService

### Declaration

```
public interface UsenetClientService extends InternetClientService
```

### All Superinterfaces:

`InternetClientService`, `javax.tv.service.Service`

### Description

Service representing the resident usenet news client.

### Methods

### setInitialGroup(URL)

```
public void setInitialGroup(java.net.URL group)
```

Set the initial group to be displayed when the news client starts. This URL is specific to this instance of UsenetClient-Service and will not impact any other instance and is only valid for the lifetime of this instance. Calling this method and then selecting the UsenetClientService instance is equivalent to selecting the UsenetClientService instance, obtaining the UsenetClient and then calling the selectMessage method there. If the application calling this method is still running when the message is sent (or fails) and has registered an InternetClientListener then the appropriate InternetClientEvent shall be sent corresponding to the success or failure of the operation to send the message. Calls to setInitialGroup shall cancel previous calls to setInitialMessage and vice-versa on the same UsenetClientService instance.

#### Parameters:

`group` - the URL of the message. This may or may not include the address of a news server.

#### Throws:

`java.lang.IllegalArgumentException` - if the specified URL does not correspond to the Usenet news URL format specified in RFC 1738 [61].

### setInitialMessage(URL)

```
public void setInitialMessage(java.net.URL message)
```

Set the initial message to be used when the news client starts. This URL is specific to this instance of UsenetClient-Service and will not impact any other instance and is only valid for the lifetime of this instance. Calling this method and then selecting the UsenetClientService instance is equivalent to selecting the UsenetClientService instance, obtaining the UsenetClient and then calling the selectMessage method there. If the application calling this method is still running when the message is sent (or fails) and has registered an InternetClientListener then the appropriate InternetClientEvent shall be sent corresponding to the success or failure of the operation to send the message. Calls to setInitialGroup shall cancel previous calls to setInitialMessage and vice-versa on the same UsenetClientService instance.

#### Parameters:

`message` - the URL of the message. This may or may not include the address of a news server.

#### Throws:

`java.lang.IllegalArgumentException` - if the specified URL does not include a Usenet news message ID or does not correspond to the Usenet news URL format specified in RFC 1738 [61].

### subscribe(String)

```
public void subscribe(java.lang.String newsgroup)
throws IOException
```

Add a newsgroup to the list currently subscribed newsgroups. If the newsgroup is already in the list, then this method has no effect. As a side-effect a newsgroup previously subscribed to by this method may be unsubscribed. Implementations may restrict the number of newsgroups a single GEM application or source of applications may subscribe to.

#### Parameters:

`newsgroup` - the name of the newsgroup that should be subscribed to.

#### Throws:

IOException - if no more newsgroups can be added due to a lack of storage space.

`java.io.IOException.`

# org.dvb.internet

# WWWBrowser

### Declaration

```
public interface WWWBrowser extends InternetClient
```

### All Superinterfaces:

```
InternetClient, javax.tv.service.selection.ServiceContentHandler
```

### Description

This interface provides support for the operations required on an WWW browser.

Any URLs passed to methods in this interface should correspond to the HTTP or FTP URL schemes specified in RFC 1738 [61]. Other schemes or URL formats may be supported, but these are implementation-specific and additional supported schemes should be discovered by querying the capabilities of the web browser if they are required.

### Methods

#### goToURL(URL)

```
public void goToURL(java.net.URL url)
throws ClientNotRunningException
```

Direct the web browser to display the page at the specified URL. If the web browser is not already running, then this method will not cause the web browser to be started and the call will fail.

This is an asynchronous operation, whose success or failure will be indicated by an `InternetClientSuccessEvent` or `InternetClientFailureEvent` or one of their subclasses.

#### Parameters:

`url` - the URL to visit.

#### Throws:

`java.lang.SecurityException` - if the caller does not have a `SocketPermission` for the host part of the specified URL.

`java.lang.IllegalArgumentException` - if the URL scheme is not supported by the web browser.

`ClientNotRunningException` - if the client is not currently running.

# org.dvb.internet

# WWWBrowserService

### Declaration

```
public interface WWWBrowserService extends InternetClientService
```

### All Superinterfaces:

`InternetClientService`, `javax.tv.service.Service`

### Description

Service representing a resident WWW browser.

### Methods

#### addBookmark(URL, String)

```
public void addBookmark(java.net.URL bookmarkUrl, java.lang.String name)
throws EntryExistsException, IOException
```

Add a bookmark to the list of bookmarks in the current application. As a side-effect a bookmark previously added by this method may be lost. Implementations may restrict the number of bookmarks a single GEM application or source of applications may add.

#### Parameters:

`bookmarkUrl` - the URL that should be added to the bookmarks list.

`name` - the name that should be displayed for that URL in the bookmarks list.

#### Throws:

`EntryExistsException` - if a bookmark with both the same name and the same URL already exists in the bookmarks list.

`java.lang.IllegalArgumentException` - if the URL scheme is not supported by the application.

IOException - if no more bookmarks can be added due to a lack of storage space or other limitation in the client.

`java.io.IOException.`

### areFramesSupported()

```
public boolean areFramesSupported()
```

Check whether frames are supported by the browser and enabled.

Returns:

true if the browser supports frames and frame support is enabled by the user, false otherwise.

### getAcceptedMediaTypes()

```
public java.lang.String[] getAcceptedMediaTypes()
```

Returns an array of supported MIME types, e.g. "application/vnd.rn-realmedia" The returned MIME types shall include at least one entry with type "text/html". Each entry of this type shall have a 'version' parameter that indicates the version of HTML that is supported, for example text/html; version = "4.0". A browser may claim to support an HTML version while only implementing a subset of the features - this is implementation-dependent. A browser that wishes to explicitly indicate support for multiple HTML versions shall return multiple entries of type "text/html" with different values for the 'version' parameter.

Returns:

an array of MIME types supported by the web browser.

### getSupportedPlugins()

```
public java.lang.String[] getSupportedPlugins()
```

Returns an array of names of installed plug-ins. The name in each string is defined by the plug-in provider.

Returns:

an array of plugin names.

### getUserAgent()

```
public java.lang.String getUserAgent()
```

Returns the string used in the HTTP "User-Agent" header.

Returns:

the string identifying the user agent.

### setHomepage(URL)

```
public void setHomepage(java.net.URL defaultUrl)
```

Set the home page for the web browser.

Parameters:

`defaultUrl` - the URL to be used as the default when the application is launched with no starting URL.

Throws:

`java.lang.SecurityException` - if the caller does not have a `HomePagePermission`.

`java.lang.IllegalArgumentException` - if the URL scheme is not supported by the application. Different classes which implement or extend this interface may implement different schemes.

### setInitialURL(URL)

```
public void setInitialURL(java.net.URL initialUrl)
```

Set the initial URL to be used when the WWW browser starts. This URL is specific to this instance of WWWBrowser-Service and will not impact any other instance and is only valid for the lifetime of this instance. Calling this method and then selecting the WWWBrowserService instance is equivalent to selecting the WWWBrowserService instance, obtaining the WWWBrowser and then calling the goToURL method there. If the application calling this method is still running when the specified initial page is displayed (or fails) and has registered an InternetClientListener then the appropriate InternetClientEvent shall be sent corresponding to the success or failure of the operation to display the specified initial URL.

Parameters:

`initialUrl` - the URL to use.

Throws:

`java.lang.IllegalArgumentException` - if the URL scheme is not supported by the application. Different classes which implement or extend this interface may implement different schemes.

`java.lang.SecurityException` - if the caller does not have a `SocketPermission` for the host part of the specified URL.

# Annex AI (normative):
# DVB Extensions for cryptography

# Package

# org.dvb.security

### Description

Enables applications to login to a PKCS11 token for non key related operations including providing PIN codes.

The AuthProvider class is needed to log into a PKCS11 token for non key related operations. In that case, there is no other way to send the PIN code. The KeyStoreBuilder class is needed to install a callback handler which is called when a PIN code is required to create a KeyStore. This is an alternative to providing the PIN code when a KeyStore is instantiated. It enables an application to log into the token for key operations without using a KeyStoreBuilder.

| Class Summary | |
|---|---|
| **Interfaces** | |
| KeyStoreProtectionParameters | A marker interface for keystore protection parameters. |
| **Classes** | |
| AuthProvider | This class defines login and logout for a provider. |
| DVBKeyStore | Extends KeyStore to allow loading the keystore using protection parameters. |
| KeyStoreBuilder | An instance of this class encapsulates the information needed to instance and initialize a KeyStore object. |
| KeyStoreCallbackHandlerProtection | A protection parameter encapsulating a CallbackHandler. |
| **Exceptions** | |
| KeyStoreException | Generic KeyStore exception. |
| LoginException | Basic login exception. |

# org.dvb.security

# AuthProvider

### Declaration

```
public abstract class AuthProvider extends java.security.Provider
```

```
java.lang.Object
  |
  +--java.util.Dictionary
        |
        +--java.util.Hashtable
              |
              +--java.util.Properties
                    |
                    +--java.security.Provider
                          |
                          +--org.dvb.security.AuthProvider
```

### All Implemented Interfaces:

```
java.lang.Cloneable, java.util.Map, java.io.Serializable
```

### Direct Known Subclasses:

```
org.dvb.security.pkcs11.DVBPKCS11Provider
```

### Description

This class defines login and logout for a provider. While callers may invoke login directly, the provider may also invoke login on behalf of callers if it determines that a login must be perfomed prior to certain operations.

### Constructors

#### AuthProvider(String, double, String)

```
protected AuthProvider(java.lang.String name, double version, java.lang.String info)
```

Creates a new instance of AuthProvider.

### Methods

#### login(Principal, CallbackHandler)

```
public abstract void login(java.security.Principal identity,
org.dvb.auth.callback.CallbackHandler handler)
throws LoginException
```

log in to this provider.

Throws:

LoginException

#### logout()

```
public abstract void logout()
throws LoginException
```

logout from this provider.

Throws:

LoginException

#### setCallbackHandler(CallbackHandler)

```
public abstract void setCallbackHandler(org.dvb.auth.callback.CallbackHandler handler)
```

set a call back handler.

# org.dvb.security

# DVBKeyStore

### Declaration

```
public class DVBKeyStore extends java.security.KeyStore
```

```
java.lang.Object
  |
  +--java.security.KeyStore
        |
        +--org.dvb.security.DVBKeyStore
```

### Description

Extends KeyStore to allow loading the keystore using protection parameters.

Since:

MHP 1.1.2.

### Constructors

#### DVBKeyStore(KeyStoreSpi, Provider, String)

```
protected DVBKeyStore(java.security.KeyStoreSpi keyStoreSpi, java.security.Provider provider,
java.lang.String type)
```

Creates a KeyStore object of the given type, and encapsulates the given provider implementation (SPI object) in it.

#### Parameters:

`keyStoreSpi` - the provider implementation.

`provider` - the provider.

`type` - the keystore type.

### Methods

#### load(KeyStoreProtectionParameters)

```
public final void load(org.dvb.security.KeyStoreProtectionParameters p)
throws IOException, NoSuchAlgorithmException, CertificateException
```

Loads this keystore using the given protection parameters.

#### Parameters:

`p` - protection parameters to use.

#### Throws:

`java.lang.IllegalArgumentException` - if the parameter p is not recognized.

`java.io.IOException` - if there is an I/O or format problem with the keystore data.

`java.security.NoSuchAlgorithmException` - if the algorithm used to check the integrity of the keystore cannot be found.

`java.security.cert.CertificateException` - if any of the certificates in the keystore could not be loaded.

# org.dvb.security

# KeyStoreBuilder

### Declaration

```
public abstract class KeyStoreBuilder

java.lang.Object
  |
  +--org.dvb.security.KeyStoreBuilder
```

### Description

An instance of this class encapsulates the information needed to instance and initialize a KeyStore object. That process is triggered when the getKeyStore method is called. This makes it possible to decouple configuration from KeyStore object creation and delay password prompt until it is needed.

### Constructors

#### KeyStoreBuilder()

```
protected KeyStoreBuilder()
```

Creates a new instance of KeyStoreBuilder.

### Methods

#### getKeyStore()

```
public abstract java.security.KeyStore getKeyStore()
throws KeyStoreException
```

Returns the KeyStore described by this object.

Throws:

KeyStoreException - if an error occurred, e.g. if an error occurred in the constructor or the load method of the KeyStore.

#### newInstance(String, Provider, KeyStoreProtectionParameters)

```
public static org.dvb.security.KeyStoreBuilder newInstance(java.lang.String type,
java.security.Provider provider, org.dvb.security.KeyStoreProtectionParameters protection)
```

Returns a new builder object. Each call to the `getKeyStore` method on the returned builder will return a new org.dvb.security.DVBKeyStore object of type type. Its `load` method is invoked with the protection parameter used to construct this KeyStoreBuilder.

Parameters:

`type` - the type of the KeyStore to be constructed. The type parameter is concatenated with the string "KeyStore." and then passed to the get method of the specified Provider in order to obtain the fully qualified name of the KeyStoreSpi implementation. For more details, see "How to Implement a Provider for the JavaTM Cryptography Architecture".

`provider` - the provider from which the keyStore is to be instantiated.

`protection` - the protection parameter securing the Keystore.

Throws:

`java.lang.IllegalArgumentException` - if protection is an application defined class.

`java.lang.NullPointerException` - if type, provider or protection are null.

# org.dvb.security

# KeyStoreCallbackHandlerProtection

### Declaration

```
public class KeyStoreCallbackHandlerProtection implements KeyStoreProtectionParameters

java.lang.Object
  |
  +--org.dvb.security.KeyStoreCallbackHandlerProtection
```

### All Implemented Interfaces:

KeyStoreProtectionParameters

### Description

A protection parameter encapsulating a CallbackHandler.

### Constructors

#### KeyStoreCallbackHandlerProtection(CallbackHandler)

```
public KeyStoreCallbackHandlerProtection(org.dvb.auth.callback.CallbackHandler handler)
```

Creates a new instance of KeyStoreCallbackHandlerProtection.

### Methods

#### getCallbackHandler()

```
public org.dvb.auth.callback.CallbackHandler getCallbackHandler()
```

Returns the callback handler.

# org.dvb.security KeyStoreException

### Declaration

```
public class KeyStoreException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.security.KeyStoreException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Generic KeyStore exception.

### Constructors

#### KeyStoreException()

```
public KeyStoreException()
```

Creates a new instance of `KeyStoreException` without detail message.

#### KeyStoreException(String)

```
public KeyStoreException(java.lang.String msg)
```

Constructs an instance of `KeyStoreException` with the specified detail message.

### Parameters:

`msg` - the detail message.

# org.dvb.security KeyStoreProtectionParameters

### Declaration

```
public interface KeyStoreProtectionParameters
```

### All Known Implementing Classes:

`KeyStoreCallbackHandlerProtection`

### Description

A marker interface for keystore protection parameters.

# org.dvb.security

# LoginException

### Declaration

```
public class LoginException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.security.LoginException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Basic login exception.

### Constructors

### LoginException()

```
public LoginException()
```

Creates a new instance of `LoginException` without detail message.

### LoginException(String)

```
public LoginException(java.lang.String msg)
```

Constructs an instance of `LoginException` with the specified detail message.

### Parameters:

`msg` - the detail message.

# Package

# org.dvb.auth.callback

### Description

Enables applications to ask the end-user for PIN codes. Applications will request the PIN code by implementing the interface CallbackHandler from this package. This handler will receive a PasswordCallback object when the implementation requires the PIN code.

NOTE: In order to prevent other applications listening in, it is recommended that applications use org.dvb.event to obtain exclusive access to the number keys when asking for the PIN code.

| Class Summary | |
|---|---|
| **Interfaces** | |
| Callback | Implementations of this interface are passed to a CallbackHandler, allowing underlying security services the ability to interact with a calling application to retrieve specific authentication data such as usernames and passwords, or to display certain information, such as error and warning messages. |
| CallbackHandler | An application implements a CallbackHandler and passes it to underlying security services so that they may interact with the application to retrieve specific authentication data, such as usernames and passwords, or to display certain information, such as error and warning messages. |
| **Classes** | |
| PasswordCallback | Underlying security services instantiate and pass a PasswordCallback to the handle method of a CallbackHandler to retrieve password information. |
| **Exceptions** | |
| UnsupportedCallbackException | Thrown when a callback handler is asked to handle a callback which it cannot handle. |

# org.dvb.auth.callback

# Callback

### Declaration

```
public interface Callback
```

### All Known Implementing Classes:

PasswordCallback

### Description

Implementations of this interface are passed to a CallbackHandler, allowing underlying security services the ability to interact with a calling application to retrieve specific authentication data such as usernames and passwords, or to display certain information, such as error and warning messages. Callback implementations do not retrieve or display the information requested by underlying security services. Callback implementations simply provide the means to pass such requests to applications, and for applications, if appropriate, to return requested information back to the underlying security services.

# org.dvb.auth.callback

# CallbackHandler

### Declaration

```
public interface CallbackHandler
```

### Description

An application implements a CallbackHandler and passes it to underlying security services so that they may interact with the application to retrieve specific authentication data, such as usernames and passwords, or to display certain information, such as error and warning messages.

### Methods

### handle(Callback[])

```
public void handle(org.dvb.auth.callback.Callback[] callbacks)
throws IOException, UnsupportedCallbackException
```

Called when a callback needs handling.

Parameters:

callbacks - - an array of Callback objects provided by an underlying security service which contains the information requested to be retrieved or displayed.

Throws:

java.io.IOException - if an input or output error occurs when retrieving the password.

UnsupportedCallbackException - if one of the callbacks in the array is not supported by the handler.

# org.dvb.auth.callback

# PasswordCallback

## Declaration

```
public class PasswordCallback implements Callback
```

```
java.lang.Object
  |
  +--org.dvb.auth.callback.PasswordCallback
```

## All Implemented Interfaces:

Callback

## Description

Underlying security services instantiate and pass a PasswordCallback to the handle method of a CallbackHandler to retrieve password information. The CallbackHandler uses this to communicate to the security services a password obtained from the end-user.

## Constructors

### PasswordCallback(String, boolean)

```
public PasswordCallback(java.lang.String prompt, boolean echoOn)
```

Creates a new instance of PasswordCallback.

Parameters:

prompt - the prompt to use.

echoOn - true if the password should be displayed as typed otherwise false.

## Methods

### clearPassord()

```
public void clearPassord()
```

Clear the retrieved password.

### getPassword()

```
public char[] getPassword()
```

Get the retrieved password.

Returns:

the last password previously set by setPassword or null if none has been set.

See Also:

setPassword(char[])

### getPrompt()

```
public java.lang.String getPrompt()
```

Get the prompt to use.

Returns:

the prompt as passed to the constructor.

### isEchoOn()

```
public boolean isEchoOn()
```

Return whether the password should be displayed as being typed.

Returns:

true if the password should be displayed otherwise false.

### setPassword(char[])

```
public void setPassword(char[] password)
```

Set the retrieved password.

Parameters:

`password` - the password to return.

See Also:

getPassword()

# org.dvb.auth.callback UnsupportedCallbackException

## Declaration

```
public class UnsupportedCallbackException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.auth.callback.UnsupportedCallbackException
```

## All Implemented Interfaces:

```
java.io.Serializable
```

## Description

Thrown when a callback handler is asked to handle a callback which it cannot handle.

### Constructors

#### UnsupportedCallbackException(Callback)

```
public UnsupportedCallbackException(org.dvb.auth.callback.Callback callback)
```

Creates a new instance of `UnsupportedCallbackException` without detail message.

Parameters:

`callback` - the callback which cannot be handled.

#### UnsupportedCallbackException(Callback, String)

```
public UnsupportedCallbackException(org.dvb.auth.callback.Callback callback, java.lang.String msg)
```

Constructs an instance of `UnsupportedCallbackException` with the specified detail message.

Parameters:

`callback` - the callback which cannot be handled.

`msg` - the detail message.

### Methods

#### getCallback()

```
public org.dvb.auth.callback.Callback getCallback()
```

Returns the Callback passed in to the constructor.

Returns:

a callback.

# Package org.dvb.net.ssl

### Description

Enables applications to provide keys and certificates for SSL/TLS connections. During the SSL handshake, the SSL engine uses TrustManagers to make sure the certificate chain received from the server is trusted. If client authentication is enabled, the SSL engine will use KeyManagers to find a certificate chain and a private key according to the CertificateRequest message from the server. The CertificateRequest message gives a list of acceptable CA for the server.

The init method in DVBTrustManagerFactory and DVBKeyManagerFactory is used by applications to provide a set of KeyStoreBuilders to these factory classes. Without these two classes, applications can only provide the PIN code at the time the KeyManagerFactory is created. Hence if the token is replaced, the existing KeyManagerFactory cannot be used and the application would need to create a new instance to use with the new token.

| Class Summary | |
|---|---|
| **Classes** | |
| DVBKeyManagerFactory | This class is used to create a KeyManagerFactory initialized with an array of KeyStoreBuilder instances. |
| DVBKeyManagerFactorySpi | This class defines the Service Provider Interface for the DVBKeyManager-Factory class. |
| DVBTrustManagerFactory | This class is used to create a KeyManagerFactory initialized with an array of KeyStoreBuilder instances. |
| DVBTrustManagerFactorySpi | This class defines the Service Provider Interface for the DVBTrustManager-Factory class. |

# org.dvb.net.ssl DVBKeyManagerFactory

### Declaration

```
public class DVBKeyManagerFactory extends javax.net.ssl.KeyManagerFactory

java.lang.Object
  |
  +--javax.net.ssl.KeyManagerFactory
        |
        +--org.dvb.net.ssl.DVBKeyManagerFactory
```

### Description

This class is used to create a KeyManagerFactory initialized with an array of KeyStoreBuilder instances. The (inherited) method getInstance shall return an instance of DVBKeyManagerFactory where provider is an instance of DVBPKCS11Provider.

### Constructors

#### DVBKeyManagerFactory(DVBKeyManagerFactorySpi, Provider, String)

```
protected DVBKeyManagerFactory(org.dvb.net.ssl.DVBKeyManagerFactorySpi factorySpi,
java.security.Provider provider, java.lang.String algorithm)
```

Creates a new instance of DVBKeyManagerFactory.

### Methods

#### init(KeyStoreBuilder[])

```
public final void init(org.dvb.security.KeyStoreBuilder[] builders)
```

This method is used to initialize the factory with an array of KeyStoreBuilder instances.

### Parameters:

`builders` - an array of KeyStoreBuilder instances.

# org.dvb.net.ssl DVBKeyManagerFactorySpi

### Declaration

```
public abstract class DVBKeyManagerFactorySpi extends javax.net.ssl.KeyManagerFactorySpi

java.lang.Object
  |
  +--javax.net.ssl.KeyManagerFactorySpi
        |
        +--org.dvb.net.ssl.DVBKeyManagerFactorySpi
```

### Description

This class defines the Service Provider Interface for the DVBKeyManagerFactory class.

### Constructors

#### DVBKeyManagerFactorySpi()

```
public DVBKeyManagerFactorySpi()
```

Creates a new instance of DVBKeyManagerFactorySpi.

### Methods

#### engineInit(KeyStoreBuilder[])

```
protected abstract void engineInit(org.dvb.security.KeyStoreBuilder[] builders)
```

This method is used to initialize the factory with an array of KeyStoreBuilder instances.

Parameters:

`builders` - an array of KeyStoreBuilder.

# org.dvb.net.ssl

# DVBTrustManagerFactory

### Declaration

```
public class DVBTrustManagerFactory extends javax.net.ssl.TrustManagerFactory
```

```
java.lang.Object
  |
  +--javax.net.ssl.TrustManagerFactory
        |
        +--org.dvb.net.ssl.DVBTrustManagerFactory
```

### Description

This class is used to create a KeyManagerFactory initialized with an array of KeyStoreBuilder instances. The inherited method getInstance shall return an instance of DVBTrustManagerFactory when the provider is an instance of DVBPKCS11Provider.

### Constructors

#### DVBTrustManagerFactory(DVBTrustManagerFactorySpi, Provider, String)

```
protected DVBTrustManagerFactory(org.dvb.net.ssl.DVBTrustManagerFactorySpi factorySpi,
java.security.Provider provider, java.lang.String algorithm)
```

Creates a new instance of DVBTrustManagerFactory.

### Methods

#### init(KeyStoreBuilder[])

```
public final void init(org.dvb.security.KeyStoreBuilder[] builders)
```

This method is used to initialize the factory with an array of KeyStoreBuilder instances.

Parameters:

`builders` - an array of KeyStoreBuilder instances.

# org.dvb.net.ssl

# DVBTrustManagerFactorySpi

### Declaration

```
public abstract class DVBTrustManagerFactorySpi extends javax.net.ssl.TrustManagerFactorySpi

java.lang.Object
  |
  +--javax.net.ssl.TrustManagerFactorySpi
        |
        +--org.dvb.net.ssl.DVBTrustManagerFactorySpi
```

### Description

This class defines the Service Provider Interface for the DVBTrustManagerFactory class.

### Constructors

#### DVBTrustManagerFactorySpi()

```
public DVBTrustManagerFactorySpi()
```

Creates a new instance of DVBTrustManagerFactorySpi.

### Methods

#### engineInit(KeyStoreBuilder[])

```
protected abstract void engineInit(org.dvb.security.KeyStoreBuilder[] builders)
```

This method is used to initialize the factory with an array of KeyStoreBuilder instances.

### Parameters:

`builders` - an array of `KeyStoreBuilder`s.

# Package

# org.dvb.security.pkcs11

### Description

Provides information about available slots and tokens and enables the selection of the slot to use.

PKCS11 smart card products in the market show significant variation. In practice, this variation is significant enough that a defining a default PKCS11 driver to be resident in all GEM terminals is impractical. DVBPKCS11Provider implementations will use the " Non-CA smart card API" (http://java.sun.com/products/satsa/) to communicate with the smart cards.

In this package, when PKCS#11 library functions are mentioned, it is to indicate that the call this method shall cause effect similar to the named PKCS#11 library equivalent.

The methods getSlotList and getTokenInfo of the class DVBPKCS11Provider can be used to get the list of slots and tokens available. The default slot used will be defined by the java property "dvb.security.pkcs11.defaultSlotId". If an application needs to use another slot, it should call the method DVBPKCS11Provider.setSlotId(int slotId) to change the slot used by the provider. The slot can only be changed when the provider is not logged into the token.

| Class Summary | |
|---|---|
| **Interfaces** | |
| SlotInfo | This interface is used to retrieve information about a PKCS11 slot. |
| TokenInfo | This interface is used to get information about a PKCS11 token. |
| **Classes** | |
| DVBPKCS11Provider | This class implements a PKCS11 security provider. |

# org.dvb.security.pkcs11

# DVBPKCS11Provider

## Declaration

```
public abstract class DVBPKCS11Provider extends org.dvb.security.AuthProvider
```

```
java.lang.Object
  |
  +--java.util.Dictionary
        |
        +--java.util.Hashtable
              |
              +--java.util.Properties
                    |
                    +--java.security.Provider
                          |
                          +--org.dvb.security.AuthProvider
                                |
                                +--org.dvb.security.pkcs11.DVBPKCS11Provider
```

## All Implemented Interfaces:

```
java.lang.Cloneable, java.util.Map, java.io.Serializable
```

## Description

This class implements a PKCS11 security provider. It can be used in the following security related packages:

- In java.security for MessageDigest, SecureRandom, Signature, KeyPairGenerator and KeyStore related classes.
- In javax.crypto (JCE) for encryption and decryption.

Providers have a slot identifier associated with them identifying each smart card reader slot. These are numbered starting from zero. For details, see the PKCS 11 specification.

## Constructors

### DVBPKCS11Provider(String, double, String, AppID)

```
protected DVBPKCS11Provider(java.lang.String name, double version, java.lang.String info,
org.dvb.application.AppID appID)
throws SecurityException
```

Creates a new instance of DVBPKCS11Provider.

## Throws:

`java.lang.SecurityException` - when called with an appID not allowed by the DVBPKCS11 implementation.

## Methods

### getSlotId()

```
public abstract int getSlotId()
```

This method returns the PKCS11 slot identifier currently associated with this provider. By default the slot identifier is defined by the property "dvb.security.pkcs11.defaultSlotId" It can be changed by calling setSlotId.

Returns:

a slot identifier.

getSlotList(boolean)

```
public org.dvb.security.pkcs11.SlotInfo[] getSlotList(boolean tokenPresent)
```

This method is used to get the list of PKCS11 Slot available for this provider. It is equivalent to the PKCS11 C_GetSlotList function to get the list of slot and C_GetSlotInfo to retreive slot information.

Parameters:

`tokenPresent` - boolean indicating if the returned list includes only the slots with a token present or all slots.

Returns:

the list of slot.

getTokenInfo(int)

```
public abstract org.dvb.security.pkcs11.TokenInfo getTokenInfo(int slotId)
throws IllegalArgumentException
```

This method is used to retreive information about a PKCS11 token in a given slot. It is equivalent to the PKCS11 C_GetTokenInfo function to get this information.

Returns:

a TokenInfo object which gives a subset of the PKCS11 CK_TOKEN_INFO.

Throws:

`java.lang.IllegalArgumentException` - if the slot does not exist or there is no token in the slot.

login(Principal, CallbackHandler)

```
public abstract void login(java.security.Principal identity,
org.dvb.auth.callback.CallbackHandler handler)
throws LoginException, NullPointerException
```

This method is used to explicitly log into a PKCS11 token. A call to this method will be equivalent to a call to the PKCS11 function C_Login. The PKCS11 user type will always be CKU_USER.

Overrides:

`login` in class `AuthProvider`

Parameters:

`identity` - This parameter is not used. It is kept to be compatible with the J2SE 5.0.

`handler` - This parameter is used to get the pin code needed to login. The callbackHandler will get a PasswordCallback in which it should put the pin code. This parameter may be null in which case the handler that was previously set by setCallbackHandler is used.

Throws:

`org.dvb.security.LoginException` - This exception is under the conditions when C_Login would return an error return an error.

`java.lang.NullPointerException` - if the CallbackHandler parameter is null and either no previous call to setCallbackHandler has occurred or the last call to that method set the handler to null.

### logout()

```
public abstract void logout()
throws LoginException
```

This method is called to explicitly log out from a PKCS11 token. A call to this method will be equivalent to a call to the PKCS11 funtion C_Logout.

#### Overrides:

`logout` in class `AuthProvider`

#### Throws:

`org.dvb.security.LoginException` - This exception is thrown under the conditions when C_Logout would return an error.

### setCallbackHandler(CallbackHandler)

```
public abstract void setCallbackHandler(org.dvb.auth.callback.CallbackHandler handler)
```

This method is used to set a default callback handler for the provider.

#### Overrides:

`setCallbackHandler` in class `AuthProvider`

#### Parameters:

`handler` - a Callback handler that will be used to get the pin code when the login method is called with a null handler.

### setSlotId(int)

```
public abstract void setSlotId(int slotId)
throws IOException, IllegalArgumentException
```

This method can be used to change the slot identifier used by the provider. The slot can only be changed when the provider is not logged into the token.

#### Parameters:

`slotId` - a slot identifier.

#### Throws:

`java.io.IOException` - this exception is thrown if this method is called when the provider is logged into the token.

`java.lang.IllegalArgumentException` - if the slot does not exist.

# org.dvb.security.pkcs11

# SlotInfo

### Declaration

```
public interface SlotInfo
```

### Description

This interface is used to retrieve information about a PKCS11 slot. The information returned here is the equivalent to the one returned by the PKCS11 function C_GetSlotInfo.

### Methods

#### getFlags()

```
public int getFlags()
```

This method returns the flags of the PKCS11 CK_SLOT_INFO.

#### getManufacturerID()

```
public java.lang.String getManufacturerID()
```

This method returns the ManufacturerID of the PKCS11 CK_SLOT_INFO.

#### getSlotDescription()

```
public java.lang.String getSlotDescription()
```

This method returns the slotDescription of the PKCS11 CK_SLOT_INFO.

#### getSlotID()

```
public int getSlotID()
```

This method returns the slot identifier associated this slot information.

# org.dvb.security.pkcs11

# TokenInfo

### Declaration

```
public interface TokenInfo
```

### Description

This interface is used to get information about a PKCS11 token. The information returned here is the equivalent to the one returned by the PKCS11 function C_GetTokenInfo.

### Methods

#### getFlags()

```
public int getFlags()
```

This method returns the flags of the PKCS11 CK_TOKEN_INFO.

#### getLabel()

```
public java.lang.String getLabel()
```

This method returns the label of the PKCS11 CK_TOKEN_INFO.

#### getManufacturerID()

```
public java.lang.String getManufacturerID()
```

This method returns the manufacturerID of the PKCS11 CK_TOKEN_INFO.

#### getModel()

```
public java.lang.String getModel()
```

This method returns the model of the PKCS11 CK_TOKEN_INFO.

### getSerialNumber()

```
public java.lang.String getSerialNumber()
```

This method returns the serial Number of the PKCS11 CK_TOKEN_INFO.

# Annex AJ (normative): Cryptographics service provider installation

## AJ.1 Introduction (informative)

PKCS11 smart card products in the market show significant variation. In practice, this variation is significant enough that a defining a default PKCS11 driver to be resident in all GEM terminals is impractical. GEM defines a mechanism to permit the downloading of cryptographic service providers in the form of Java classes. Specifically, a cryptographic service provider is a set of Java classes that provide a concrete implementation of security related functions.

To implement a cryptographic provider for the smart card, the following is needed:

a) Implement the subclasses of the needed SPI classes.
The following SPI (Service Provider Interface) classes are available :

```
java.security.SignatureSpi,
java.security.MessageDigestSpi,
java.security.KeyPairGeneratorSpi,
java.security.SecureRandomSpi,
java.security.AlgorithmParameterGeneratorSpi,
java.security.AlgorithmParametersSpi,
java.security.KeyFactorySpi,
java.security.cert.CertificateFactorySpi,
java.security.KeyStoreSpi,
javax.crypto.CipherSpi,
javax.crypto.MacSpi,
org.dvb.net.ssl.DVBKeyManagerFactorySpi,
org.dvb.net.ssl.DVBTrustManagerFactorySpi,
```

To access the smart card, the provider implementation must use the Non-CA smart card API as defined in clause 11.9.4, "Non-CA smart card API".

b) Implement the master provider class which is a subclass of org.dvb.security.pkcs11.DVBPKCS11Provider. The main purpose of this class is to give the list of the SPI classes implemented by the provider. This list is given via a set of properties initialized in the constructor of the class.

The present document does not define a mechanism by which applications may request instances of java.security.SecurityPermission. Hence applications will not be able to use the methods addProvider and removeProvider from the class java.security.Security to install and remove providers.

## AJ.2 The org.dvb.security.provider package

# Package org.dvb.security.provider

### Description

Enables the use of providers (as supported by the org.dvb.spi package) as cryptographic service providers (as supported by the java.security package).

| Class Summary | |
|---|---|
| **Interfaces** | |
| CryptographicServiceProviderProvider | This xlet-bound provider permits a cryptographic service provider to be packaged and managed as part of the GEM provider framework as supported through the org.dvb.spi package. |
| **Classes** | |
| ProviderManager | This class gives access to those cryptographic service providers visible to an application. |

# org.dvb.security.provider CryptographicServiceProviderProvider

### Declaration

```
public interface CryptographicServiceProviderProvider extends org.dvb.spi.XletBoundProvider
```

### All Superinterfaces:

org.dvb.spi.Provider, org.dvb.spi.XletBoundProvider

### Description

This xlet-bound provider permits a cryptographic service provider to be packaged and managed as part of the GEM provider framework as supported through the org.dvb.spi package. Xlet-bound providers shall not be visible to applications through the methods of the java.security.Security class.

### Methods

#### getProvider()

```
public org.dvb.security.pkcs11.DVBPKCS11Provider getProvider()
```

Obtain a DVBPKCS11Provider to be used by this Xlet.

### Returns:

a DVBPKCS11Provider.

# org.dvb.security.provider ProviderManager

### Declaration

```
public class ProviderManager

java.lang.Object
  |
  +--org.dvb.security.provider.ProviderManager
```

### Description

This class gives access to those cryptographic service providers visible to an application. The cryptographic service providers visible to an application shall include the following:

- The installed providers as returned by java.security.Security.getProviders.
- Xlet bound providers which are bound to the calling Xlet.

Where providers of the same name are visible to an application through both the above mechanisms, the Xlet bound provider shall always be used by this class, even if the installed provider has a higher version number. Installed providers can always be accessed through java.security.Security.getProviders.

### Constructors

#### ProviderManager()

```
protected ProviderManager()
```

Creates a new instance of ProviderManager. This constructor is provided for the use of implementations and specifications which extend this specification. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

### Methods

#### getInstance()

```
public static org.dvb.security.provider.ProviderManager getInstance()
```

Gets the singleton instance of the ProviderManager.

**Returns:**

the ProviderManager.

#### getProvider(String)

```
public java.security.Provider getProvider(java.lang.String name)
```

Returns a visible provider with the specified name. If no provider with that name is visible to the calling application then null shall be returned.

**Parameters:**

`name` - the name of the provider.

**Returns:**

a provider or null.

#### getProviders()

```
public java.security.Provider[] getProviders()
```

Returns an array containing all the providers visible to the calling application.

**Returns:**

an array of providers.

# Annex AK (normative): Extended service selection API

# Package org.dvb.service.selection

### Description

Extensions to the Java TV service selection API.

| Class Summary | |
|---|---|
| Interfaces<br>DvbServiceContext | An extension to ServiceContext permitting the discovery of the network interface (if any) being used by a service context. |

# org.dvb.service.selection DvbServiceContext

### Declaration

```
public interface DvbServiceContext extends javax.tv.service.selection.ServiceContext
```

### All Superinterfaces:

```
javax.tv.service.selection.ServiceContext
```

### Description

An extension to ServiceContext permitting the discovery of the network interface (if any) being used by a service context. When a service is selected in a service context, the resource priority of the service context shall be initialised to the resource priority of the service where one is set. Where a service does not have a resource priority, the priority of the service context shall be initialised to 128.

### Methods

#### getNetworkInterface()

```
public org.davic.net.tuning.NetworkInterface getNetworkInterface()
```

Return the NetworkInterface reserved by this ServiceContext. The NetworkInterface instance returned shall be .equals() to one of those returned by NetworkInterfaceManager.getNetworkInterfaces().

#### Returns:

a NetworkInterface object or null if this ServiceContext has no NetworkInterface reserved.

#### getPriority()

```
public int getPriority()
```

Get the resource priority for this service context. The priority is a number between 0 and 255.

#### Returns:

the resource priority for this service context.

Since:

MHP 1.2.

### setDefaultVideoTransformation(VideoTransformation)

```
public void setDefaultVideoTransformation(org.dvb.media.VideoTransformation vt)
```

Sets the default video transformation to be applied to the new service following a service selection operation. The identity of the last application to call this method for each service context instance shall be remembered by the GEM terminal. If this is set then the presentation of the video of the new service will use the video transformation of the existing service until it is known whether that application survives. If that application survives then this transformation shall be applied to the video of the new service. If that application does not survive then the video transformation of the new service will be reset to the default for the platform and the value set by the call to this method discarded. This method shall have no effect if the ServiceMediaHandler for the service context is a component based player when service selection happens.

Parameters:

`vt` - the video default transformation or null to reset to the default for the platform.

Since:

MHP 1.1.2.

### setPriority(int)

```
public void setPriority(int priority)
```

Set the resource priority for this service context. The priority is a number between 0 and 255.

Parameters:

`priority` - the resource priority for this service context.

Throws:

`java.lang.IllegalArgumentException` - if the priority is outside this specified range.

Since:

MHP 1.2.

# Annex AL (normative): Extended content referencing API

The class `org.dvb.locator.FrequencyLocator`, defined in this annex, is optional for GEM terminals.

# Package

# org.dvb.locator

### Description

Provides Locators for various concepts not addressed in the org.davic and javax.tv packages.

| Class Summary | |
|---|---|
| Classes | |
| FrequencyLocator | Used to reference a service that does not carry any DVB-SI and hence does not appear in the service list of a receiver. |
| NetworkInterfaceBoundMediaLocator | Class representing a MediaLocator combined with a network interface. |

# org.dvb.locator

# FrequencyLocator

### Declaration

```
public class FrequencyLocator extends org.davic.net.Locator
```

```
java.lang.Object
  |
  +--org.davic.net.Locator
        |
        +--org.dvb.locator.FrequencyLocator
```

### All Implemented Interfaces:

```
javax.tv.locator.Locator
```

### Description

Used to reference a service that does not carry any DVB-SI and hence does not appear in the service list of a receiver. The external form of locators of this class is implementation specific.

### Since:

MHP 1.1.2.

### Constructors

#### FrequencyLocator(byte[], int)

```
public FrequencyLocator(byte[] delivery_system_descriptor, int program_number)
throws InvalidLocatorException
```

Constructor for referencing a service which does not carry any DVB-SI.

Parameters:

`delivery_system_descriptor` - one of the delivery system descriptors defined in clause 6.2.13 ("Delivery system descriptors") of the DVB-SI specification.

`program_number` - the MPEG program_number of the service within the transport stream.

Throws:

`org.davic.net.InvalidLocatorException` - if the delivery_system_descriptor parameter is incorrect - the descriptor tag does not define one of the permitted descriptors, the length of the array is not correct for the descriptor tag, any values within the descriptor are outside any range defined for them.

Methods

getDeliverySystemDescriptor()

```
public byte[] getDeliverySystemDescriptor()
```

Return the delivery system descriptor as passed into the constructor.

Returns:

a byte array containing a DVB-SI delivery system descriptor.

getProgramNumber()

```
public int getProgramNumber()
```

Return the MPEG program number as passed in to the constructor.

Returns:

an MPEG program number.

# org.dvb.locator

# NetworkInterfaceBoundMediaLocator

Declaration

```
public class NetworkInterfaceBoundMediaLocator extends javax.media.MediaLocator
```

```
java.lang.Object
  |
  +--javax.media.MediaLocator
        |
        +--org.dvb.locator.NetworkInterfaceBoundMediaLocator
```

Description

Class representing a MediaLocator combined with a network interface. Used by applications that want to control which network interface is used by JMF when multiple network interfaces exist which are connected to the same actual network, for example a multi-tuner PVR.

JMF players constructed using instances of this class shall only use the specified network interface. If the media referenced by that media locator is not available via that network interface, the JMF player shall behave as if the media is not available at all, even if it is in fact available by another network interface.

Since:

MHP 1.1.2.

### Constructors

#### NetworkInterfaceBoundMediaLocator(MediaLocator, NetworkInterface)

```
public NetworkInterfaceBoundMediaLocator(javax.media.MediaLocator locator,
org.davic.net.tuning.NetworkInterface ni)
```

Construct an instance of this class.

Parameters:

`locator` - the MediaLocator to be combined with a NetworkInterface.

`ni` - a NetworkInterface.

### Methods

#### getNetworkInterface()

```
public org.davic.net.tuning.NetworkInterface getNetworkInterface()
```

Return the network interface aspect of this MediaLocator.

Returns:

the network interface passed into the constructor.

# Annex AM (normative): Smart card reader API

# Package

# org.dvb.smartcard

### Description

Provides access to smart card readers. This access includes information about the status of any smart card in those readers and any changes in that status.

| Class Summary | |
|---|---|
| Interfaces | |
| SmartCardReaderListener | The listener interface to receive smart card reader status changes. |
| Classes | |
| SmartCardReader | Represents a physical smart card reader slot in the terminal. |
| SmartCardReaderEvent | Represents an event generated by a change in the status of a smart card reader. |
| SmartCardReaderManager | The smart card reader manager allow user to know the status of any slot available on the terminal. |

# org.dvb.smartcard

# SmartCardReader

### Declaration

```
public class SmartCardReader

java.lang.Object
  |
  +--org.dvb.smartcard.SmartCardReader
```

### Description

Represents a physical smart card reader slot in the terminal.

### Since:

MHP 1.1.3.

### Constructors

### SmartCardReader()

```
protected SmartCardReader()
```

This constructor is provided for the use of implementations and specifications which extend this specification. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

### Methods

#### addSmartCardReaderListener(SmartCardReaderListener)

```
public void addSmartCardReaderListener(org.dvb.smartcard.SmartCardReaderListener listener)
```

Allows the specified listener to received notifications about changes in the status of the smart card reader.

Parameters:

`listener` - the SmartCardReaderListener that will receive the notifications.

#### getSlotId()

```
public int getSlotId()
```

Each single reader is identified on the terminal with an increasing identifier. Default identifier for single slot terminals is zero.

Returns:

int id associated to the reader.

#### getStatus()

```
public int getStatus()
```

Retrieves current status of the smart card reader. The constant values are the ones defined in SmartCardReaderEvent.

Returns:

int current status of the reader.

#### isSmartCardInserted()

```
public boolean isSmartCardInserted()
```

Allows applications to query if a smart card is inserted in the reader. True is returned also if no ATR is correctly retrieved (i.e. smart card inserted upside-down).

Returns:

boolean true if smart card is inserted in given slot.

#### openRawConnection()

```
public javax.microedition.apdu.APDUConnection openRawConnection()
throws IOException, ConnectionNotFoundException
```

Opens a raw APDU connection without selecting an application or managing channels. The following properties shall apply to raw connections:

- There may be only one raw connection open at the same time with the same smart card.
- The implementation shall open the connection without sending any extra commands.
- A GEM application is allowed to send Application Selection (SELECT FILE by DF NAME) and MANAGE CHANNEL commands.
- For the T=0 protocol, for case 4 and case 2 command APDUs the card may respond with 61 XX or 6C XX statuses. The implementation should not handle these special cases, it should neither send GET RESPONSE, nor resend commands. The GEM application is required to handle these cases.

Returns:

an APDUConnection with the selected smartcard.

Throws:

`java.lang.SecurityException` - if the application does not have APDUPermission.

`javax.microedition.io.ConnectionNotFoundException` - if either the smartcard is not inserted or if opening a connection is not permitted (e.g. for CA cards).

`java.io.IOException` - if an APDU connection is already open or if any other I/O error occurs.

### removeSmartCardReaderListener(SmartCardReaderListener)

```
public void removeSmartCardReaderListener(org.dvb.smartcard.SmartCardReaderListener listener)
```

Removes specified listener: the notifications about changes in the status of the smart card reader will no longer be forwarded to the given listener.

Parameters:

`listener` - the SmartCardReaderListener that was receiving the notifications.

# org.dvb.smartcard

# SmartCardReaderEvent

### Declaration

```
public class SmartCardReaderEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.smartcard.SmartCardReaderEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Represents an event generated by a change in the status of a smart card reader. All available constants are defined within this class.

### Since:

MHP 1.1.3.

### Fields

#### SMART_CARD_ERROR

```
public static final int SMART_CARD_ERROR
```

Smart card is inserted into the reader, there is electrical communication with the smart card but no ATR is retrieved.

#### SMART_CARD_IN

```
public static final int SMART_CARD_IN
```

A smart card is inserted into the reader and an ATR is correctly retrieved. This means the insertion contact is active and the reader is also able to communicate with the smart card.

### SMART_CARD_MUTED

```
public static final int SMART_CARD_MUTED
```

Smart card is inserted into the reader but no ATR is retrieved because no electrical communication is established with the smart card.

### SMART_CARD_OUT

```
public static final int SMART_CARD_OUT
```

Nothing is inserted into the reader, meaning the insertion contact is disabled.

### Constructors

#### SmartCardReaderEvent(Object, int)

```
public SmartCardReaderEvent(java.lang.Object source, int type)
```

Constructor for a smart card reader event notifying the slot has identified a change in its status.

#### Parameters:

`source` - Object the `SmartCardReader` who is generating the event.

`type` - int the event type.

### Methods

#### getType()

```
public int getType()
```

Retrieves the type of `SmartCardReaderEvent` that has been fired. It can be either `SMART_CARD_IN`, `SMART_CARD_OUT`, `SMART_CARD_MUTED` or `SMART_CARD_ERROR`.

#### Returns:

int type of event.

# org.dvb.smartcard

# SmartCardReaderListener

### Declaration

```
public interface SmartCardReaderListener
```

### Description

The listener interface to receive smart card reader status changes.

#### Since:

MHP 1.1.3.

### Methods

#### smartCardReaderEventReceived(SmartCardReaderEvent)

```
public void smartCardReaderEventReceived(org.dvb.smartcard.SmartCardReaderEvent event)
```

Called by the terminal when a change in the status of the smart card reader is notified.

#### Parameters:

`event` - SmartCardReaderEvent the event that was fired.

# org.dvb.smartcard SmartCardReaderManager

## Declaration

```
public final class SmartCardReaderManager

java.lang.Object
  |
  +--org.dvb.smartcard.SmartCardReaderManager
```

## Description

The smart card reader manager allow user to know the status of any slot available on the terminal. The manager can dispatch the change of status in any reader to applications that registered themselves for monitoring it or can synchronously return the current status of the reader. The SmartCardReaderManager is a singleton.

### Since:

MHP 1.1.3.

## Constructors

### SmartCardReaderManager()

```
protected SmartCardReaderManager()
```

This constructor is provided for the use of implementations and specifications which extend this specification. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

## Methods

### getInstance()

```
public static org.dvb.smartcard.SmartCardReaderManager getInstance()
```

Used to get the unique istance of the smart card reader manager.

### Returns:

SmartCardReaderManager the manager singleton instance.

### getNumber()

```
public int getNumber()
```

Retrieves the number of smart card readers provided on the terminal.

### Returns:

int number of card readers.

### getSmartCardReaders()

```
public org.dvb.smartcard.SmartCardReader[] getSmartCardReaders()
```

Allows application to retrieve an array including all the smart card readers provided on the terminal. In case no readers are provided, an array with length zero is returned.

### Returns:

SmartCardReader[] array of smart card readers.

# Annex AN (normative): Provider APIs

# Package

# org.dvb.spi

### Description

This package defines a central registry for all DVB Service Provider Interface (SPI) providers. This encourages a consistent architecture for such providers. Additionally, it gives some support for debugging. For example, a list of installed providers might be useful in a crash log.

In normal operation, xlets using a provider never get a direct reference to that provider. Rather, xlets interact with the terminal through standardized APIs, such as Java TV. The service underlying those underlying APIs can be:

- an integrated part of the implementation; or
- interoperable downloaded code, either in the form of an xlet-bound or a system-bound provider.

NOTE: The Java platform class `java.security.Provider` is unrelated to the providers defined in this package and its subpackages. Additionally, `java.security.Provider` in the Java platform is typically associated with a completely different lifecycle model.

| Class Summary | |
|---|---|
| **Interfaces** | |
| `Provider` | Abstract interface for all DVB providers. |
| `SystemBoundProvider` | Interface for providers that are registered from one Xlet, but provide system-wide services. |
| `XletBoundProvider` | Interface for all Xlet-bound providers. |
| **Classes** | |
| `ProviderPermission` | This class is for applications which wish to be able to install providers. |
| `ProviderRegistry` | Registry of providers. |
| **Exceptions** | |
| `ProviderFailedInstallationException` | Thrown when there is a problem installing a provider. |

# org.dvb.spi

# Provider

### Declaration

```
public interface Provider
```

### All Known Subinterfaces:

```
org.dvb.security.provider.CryptographicServiceProviderProvider,
org.dvb.spi.ict.InteractionChannelTransportProvider, org.dvb.spi.selection.SelectionProvider,
org.dvb.spi.si.full.SIManagerProvider, org.dvb.spi.si.simple.SimpleSIProvider, SystemBoundProvider,
XletBoundProvider
```

### Description

Abstract interface for all DVB providers.

Since:

MHP 1.1.3.

Methods

getName()

```
public java.lang.String getName()
```

Returns the name of this provider. This can be used for debugging purposes, e.g. in a crash log. The name shall be encoded as defined for the permission request file in the main body of the present document. For example "0x0000000B.EMV_PK11.VISA_REVOLVER".

Returns:

the name.

getServiceProviderInterfaces()

```
public java.lang.Class[] getServiceProviderInterfaces()
```

Gives a list of the SPI's implemented by this provider. The list shall be at least one element long, and shall contain only valid SPI interfaces defined by the terminal specification. Unknown interfaces (e.g. application-defined interfaces) shall be rejected, as documented in ProviderRegistry.

Returns:

a list of SPIs.

See Also:

```
ProviderRegistry.registerXletBound(XletBoundProvider),
ProviderRegistry.registerSystemBound(SystemBoundProvider)
```

getVersion()

```
public java.lang.String getVersion()
```

Return the version of this provider. The format of this string is not specified.

Returns:

- the version of this provider.

providerRegistered()

```
public void providerRegistered()
```

Called by the system when this provider is registered.

providerUnregistered()

```
public void providerUnregistered()
```

Called by the system when this provider is unregistered.

# org.dvb.spi ProviderFailedInstallationException

### Declaration

```
public class ProviderFailedInstallationException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.spi.ProviderFailedInstallationException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Thrown when there is a problem installing a provider.

### Since:

MHP 1.1.3.

### See Also:

```
ProviderRegistry.registerSystemBound(SystemBoundProvider),
ProviderRegistry.registerXletBound(XletBoundProvider)
```

### Constructors

#### ProviderFailedInstallationException()

```
public ProviderFailedInstallationException()
```

Constructs an exception with no reason.

#### ProviderFailedInstallationException(String)

```
public ProviderFailedInstallationException(java.lang.String s)
```

Constructs an exception with a reason.

### Parameters:

`s` - the reason why the exception was thrown.

# org.dvb.spi ProviderPermission

### Declaration

```
public class ProviderPermission extends java.security.BasicPermission
```

```
java.lang.Object
  |
  +--java.security.Permission
        |
        +--java.security.BasicPermission
              |
              +--org.dvb.spi.ProviderPermission
```

All Implemented Interfaces:

`java.security.Guard, java.io.Serializable`

Description

This class is for applications which wish to be able to install providers. A ProviderPermission contains a name and an action string.

The permission name shall be either the fully qualified class name of the provider class to be installed or "*" meaning any provider. Applications may have multiple instances of this permission in order to be able to install more than one but not all providers.

The actions list shall either be "xlet" or "system". "xlet" means the right to install the provider as an xlet bound provider and "system" as a system bound provider. No checking shall be performed on whether the specified class name is consistent with the action.

Since:

MHP 1.1.3.

Constructors

ProviderPermission(String)

```
public ProviderPermission(java.lang.String name)
```

Creates a new ProviderPermission with the specified name. The name is the symbolic name of the ProviderPermission.

Parameters:

`name` - the name of the ProviderPermission or "*"

ProviderPermission(String, String)

```
public ProviderPermission(java.lang.String name, java.lang.String actions)
```

Creates a new ProviderPermission object with the specified name. The name is the symbolic name of the Provider-Permission. The actions string should be either "xlet" or "system". This constructor exists for use by the Policy object to instantiate new Permission objects.

Parameters:

`name` - the name of the ProviderPermission or "*".

`actions` - the requested actions.

# org.dvb.spi

# ProviderRegistry

Declaration

```
public class ProviderRegistry
```

```
java.lang.Object
  |
  +--org.dvb.spi.ProviderRegistry
```

Description

Registry of providers.

Since:

MHP 1.1.3.

### Constructors

#### ProviderRegistry()

```
protected ProviderRegistry()
```

This constructor is provided for use by implementations and by other specifications that extend this class. It is not to be used by normal applications.

### Methods

#### getInstalledProviders()

```
public java.lang.String[] getInstalledProviders()
```

Return the names of all installed providers. These are the names returned by the getName methods on those Providers. Provider names shall be encoded as defined for permission request file in the main body of the present document. For example "0x0000000B.EMV_PK11.VISA_REVOLVER".

**Returns:**

the names of all installed providers.

**See Also:**

`Provider.getName()`

#### getInstance()

```
public static org.dvb.spi.ProviderRegistry getInstance()
```

Return the singleton provider registry as seen by the calling application.

**Returns:**

the provider registry.

#### getProviderVersion(String)

```
public java.lang.String getProviderVersion(java.lang.String provider)
```

Return the version of an installed provider.

**Parameters:**

`provider` - the name of a provider as returned by the method getInstalledProviders.

**Returns:**

the version of the specified provider.

**Throws:**

`java.lang.IllegalArgumentException` - if the provider name is not one of those installed, i.e. is not one returned from a call to getInstalledProviders.

**See Also:**

`Provider.getVersion()`

#### registerSystemBound(SystemBoundProvider)

```
public void registerSystemBound(org.dvb.spi.SystemBoundProvider p)
throws ProviderFailedInstallationException
```

Registers a provider. Note that providers might be installed "automatically" by the terminal, e.g. due to signalling.

**Parameters:**

`p` - the provider to register.

Throws:

`java.lang.IllegalArgumentException` - if Provider does not expoort a valid set of services as determined by Provider.getServiceProviderInterfaces().

`ProviderFailedInstallationException` - if the organisation_id in the name of the provider does not match the organisation_id in a certificate which can authenticate the provider class.

`java.lang.SecurityException` - if the caller, for all of the SPIs implemented by the provider, does not have a ProviderPermission whose name is the fully qualified name of the class returned by Provider getService-ProviderInterfaces and whose action is "system".

See Also:

`Provider.getServiceProviderInterfaces()`

### registerXletBound(XletBoundProvider)

```
public void registerXletBound(org.dvb.spi.XletBoundProvider p)
throws ProviderFailedInstallationException
```

Registers a provider. Note that providers might be installed "automatically" by the terminal, e.g. due to signalling.

Parameters:

`p` - the provider to register.

Throws:

`java.lang.IllegalArgumentException` - if Provider does not export a valid set of services as determined by Provider.getServiceProviderInterfaces(), or if the provider does not have have a non-null Xlet context.

`ProviderFailedInstallationException` - if the organisation_id in the name of the provider does not match the organisation_id in a certificate which can authenticate the provider class.

`java.lang.SecurityException` - if the caller, for all of the SPIs implemented by the provider, does not have a ProviderPermission whose name is the fully qualified name of the class returned by Provider getService-ProviderInterfaces and whose action is "xlet".

See Also:

`Provider.getServiceProviderInterfaces()`, `XletBoundProvider.getBoundXletContext()`, `XletBoundProvider.getBoundPBPXletContext()`

### unregister(Provider)

```
public void unregister(org.dvb.spi.Provider p)
```

Unregister a provider. Xlets that "manually" register a provider using one of the register methods of this class shall unregister that provider before returning from a successful destroyXlet call.

Parameters:

`p` - the provider to unregister.

See Also:

`javax.tv.xlet.Xlet.destroyXlet(boolean)`

# org.dvb.spi

# SystemBoundProvider

Declaration

```
public interface SystemBoundProvider extends Provider
```

All Superinterfaces:

```
Provider
```

All Known Subinterfaces:

```
org.dvb.spi.ict.InteractionChannelTransportProvider, org.dvb.spi.selection.SelectionProvider,
org.dvb.spi.si.simple.SimpleSIProvider
```

Description

Interface for providers that are registered from one Xlet, but provide system-wide services. Such providers shall be used by the system in such a way that there is no sharing of mutable object instances between the xlet that installs the provider, and any xlet that uses the services.

Since:

MHP 1.1.3.

See Also:

`XletBoundProvider`

# org.dvb.spi

# XletBoundProvider

Declaration

```
public interface XletBoundProvider extends Provider
```

All Superinterfaces:

```
Provider
```

All Known Subinterfaces:

```
org.dvb.security.provider.CryptographicServiceProviderProvider,
org.dvb.spi.si.full.SIManagerProvider
```

Description

Interface for all Xlet-bound providers. An instance of an Xlet-bound provider provides its services for exactly one xlet: The xlet with which that provider is registered. The classes of the provider are in the classloader hierarchy of the xlet itself. This leads to a very simple mechanism, because there are no issues with instance sharing between the provider and the xlet that uses the provider. However, it has the disadvantage that every xlet that needs the services provided must carry its own copy of the provider.

Since:

MHP 1.1.3.

See Also:

`SystemBoundProvider`

Methods

getBoundPBPXletContext()

```
public javax.microedition.xlet.XletContext getBoundPBPXletContext()
```

A valid provider shall return a non-null value from at least one of the get...XletContext methods.

Returns:

the xlet context of the xlet to which the provider is bound.

getBoundXletContext()

```
public javax.tv.xlet.XletContext getBoundXletContext()
```

A valid provider shall return a non-null value from at least one of the get...XletContext methods.

Returns:

the xlet context of the xlet to which the provider is bound.

# Package org.dvb.spi.selection

Description

This package defines an SPI for selection of services and service components. The providers defined in this package allow the presentation of services whose location is not announced using standard signalling. One a SelectionProvider has been installed, the standard GEM APIs, such as the JavaTV service selection API, can be used to present those services.

Examples of API usage

*Switched Digital*

The following table shows the sequence of API calls involved in a service selection operation, both API calls between an application and the implementation and between the implementation and a SelectionProvider.

The table covers both the case when the service location is known and when the location is unknown. The first column indicates if the step applies when the service location is unknown. The second column indicates whether the step applies when the service location is known. Most steps either apply in both cases at the same point in the sequence or only apply when the service location is unknown. Two steps apply in both cases but at different points in the sequence.

| Unknown Service Location | Known Service Location | Step | API call by application | API call to/from provider |
|---|---|---|---|---|
| Y | Y | Provider is installed | `ProviderRegistry.registerSystemBound` | `Provider.providerRegistered` |
| Y | Y | Provider is initialised | None | `SelectionProvider.init` |
| Y | Y | Implementation asks provider for supported services | None | `SelectionProvider.getServiceList()` |
| Y | Y | Each supported service is merged into the service list of the receiver | None | Methods on each ServiceReference returned from `SelectionProvider.getServiceList()` |
| N | Y | Extract tuning information for service | None | For GEM, `org.dvb.locator.FrequencyLocator.getDeliverySystemDescriptor()` |
| Some time later | | | | |
| Y | Y | Application obtains service object corresponding to locator of service | SIManager.get-Service(Locator-Factory.create-Locator( "dvb://"movie-channel-1.broadcaster-b.com"") | None |

| Unknown Service Location | Known Service Location | Step | API call by application | API call to/from provider |
|---|---|---|---|---|
| Y | Y | Application asks for service to be presented | Service-Context.select(Service) | None |
| Y | N | Create session for the presentation of the requested service. | None | 1) new ServiceReference(..)<br>2) `SelectionProvider.newSession` |
| Y | N | Implementation asks provider to present channel. | None | `SelectionSession.select` |
| Y | N | Provider asks head-end for channel | None | Existing java.net APIs |
| Y | N | Head-end returns reference to channel location | None | Existing java.net APIs |
| Y | N | Provider passes on reference to channel location to implementation | None | 1a) For GEM, new Frequency-Locator(...)<br>1b) For OCAP, new OCAPLocator(int frequency, int programNumber,int modulationFormat, ...)<br>2) Return from call to `SelectionSession.select()` |
| Y | N | Extract tuning information for service | None | For GEM, `org.dvb.locator.FrequencyLocator.getDeliverySystemDescriptor()` |
| Y | Y | Implementation tunes to appropriate frequency etc | None | None |
| N | Y | Create session for the presentation of the requested service. | None | 1) new ServiceReference(..)<br>2) `SelectionProvider.newSession` |
| Y | N | Inform provider that implementation is ready to receive content | None | `SelectionSession.selectionReady` |
| Y | N | Provider tells server to send content | Provider sends message to switched digital server | Existing java.net APIs |
| Y | Y | Content arrives | 1) new Normal-ContentEvent<br>2) ServiceContext-Listener.receive-ServiceContext-Event(..) | None |
| Some time later | | | | |
| Y | Y | Content stops being presented (change to new channel, ...) | Either none or Service-Context.select() with a different service. | `SelectionSession.destroy` |
| Y | Y | Content may stop being delivered | Provider may tell head-end that this content is no longer needed if the head-end is counting the number of users of the content | Existing java.net APIs |

### Video on Demand using Non-Standard RTSP

The table below shows the use of a org.dvb.spi.selection.SelectionProvider with RTSP when content is presented using the GEM service selection API. The rows colour coded in grey are specific to content being presented using the the service selection API. Equivalents exist if JMF is used to present content using a provider.

| Step | API call by application | API call to/from provider | Notes |
|---|---|---|---|
| Provider is installed | `ProviderRegistry.registerSystemBound` | `Provider.providerRegistered` | |
| Provider is initialised | None | `SelectionProvider.init` | |
| Provider registers locator schemes it can select | None | Implementation calls `SelectionProvider.getSupportedLocatorSchemes()` which returns new org.dvb.spi.selection.Locator-Scheme("rtsp", true) | |
| Some time later | | | |
| Application decides which content is to be played back | new RTSPLocator( ... ) | None | |
| Application asks for content to be presented | SIManager.getService(Locator)<br>ServiceContext.select(Service) | None | |
| Implementation decides whether to use built-in RTSP implementation or provider. | None | `LocatorScheme.getProviderFirst()` | |
| Create session for the presentation of the requested content. | None | New ServiceReference(..)<br>`SelectionProvider.newSession` | |
| Provider negotiates RTSP session with server. | None | Provider uses existing java.net APIs to issue SETUP command including GEM terminal IP address and UDP port chosen by the provider to receive the MPEG stream from the RTSP server | |
| Provider tells implementation which port will be used for incoming content | None | Provider returns new `org.dvb.locator.ip.UnicastLocator` from call to `SelectionSession.select()` | |
| Implementation listens on port identified by the UDPLocator returned from the provider and sets up MPEG demux / decoder chain as appropriate | None | None | |
| Server sends content when GEM terminal is ready to receive it. | None | Implementation calls `SelectionSession.selectionReady()`<br>Provider uses existing java.net APIs to send PLAY command to server | |
| Content arrives | 1) new NormalContentEvent<br>2) ServiceContextListener.receive-ServiceContextEvent(..) | None | |
| Some time later | | | |
| Application wants to pause playback | Player.setRate(0) | Implementation calls `SelectionSession.setRate(float)`<br>Provider uses existing java.net APIs to send RTSP pause command to server. | |
| Some time later | | | |
| Application wants to resume playback | Player.setRate(1.0) | Implementation calls `SelectionSession.setRate(float)`<br>Provider uses existing java.net APIs to send RTSP PLAY command to server. | |
| Some time later | | | |
| Application wants playback to jump to new location | Player.setMediaTime() | Implementation calls `SelectionSession.setPosition(long)` | |

| Step | API call by application | API call to/from provider | Notes |
|---|---|---|---|
| | | Provider uses existing java.net APIs to send PLAY command specifying new starting offset. | |
| Some time later | | | |
| Application wants playback to switch into fast forwards | Player.setRate() | Implementation calls `SelectionSession.setRate(float)`<br>Provider uses existing java.net APIs to send PLAY command with desired playscale | |
| Some time later | | | |
| Content stops being presented (change to new channel, ...) | Either none or ServiceContext.select() with a different service. | `SelectionSession.destroy()` | |
| Provider sends TEARDOWN command to server and closes session | None | Existing java.net APIs | |

| Class Summary | |
|---|---|
| **Interfaces** | |
| `SelectionProvider` | A provider of service selection. |
| `SelectionProviderContext` | Platform implementations that wish to receive notifications from a `SelectionProvider` register an instance that implements this interface with the `SelectionProvider`. |
| `SelectionSession` | A session for presentation of one or more services over a period of time by a SelectionProvider. |
| `ServiceDescription` | Represents the description of a service. |
| **Classes** | |
| `KnownServiceReference` | A reference to a service that is reached with the mediation of a SelectionProvider. |
| `LocatorScheme` | This class is used to represent a locator scheme that is supported by a SelectionProvider. |
| `ServiceReference` | A reference to a service that is reached with the mediation of a SelectionProvider. |

# org.dvb.spi.selection

# KnownServiceReference

### Declaration

```
public class KnownServiceReference extends ServiceReference

java.lang.Object
  |
  +--org.dvb.spi.selection.ServiceReference
        |
        +--org.dvb.spi.selection.KnownServiceReference
```

### Description

A reference to a service that is reached with the mediation of a SelectionProvider. This class represents services whose location is already known to the provider without needing to ask a server or head-end.

### Since:

MHP 1.1.3.

### See Also:

`SelectionProvider`

### Constructors

#### KnownServiceReference(String, String, Locator)

```
public KnownServiceReference(java.lang.String transportIndependent,
java.lang.String transportDependent, javax.tv.locator.Locator actualLocation)
```

Create a ServiceReference for the given service identified by the combination of a transport independent and transport dependent identifiers.

#### Parameters:

`transportIndependent` - The transport-independent locator of the service.

`transportDependent` - The transport-dependent locator of the service.

`actualLocation` - The actual location of the service.

### Methods

#### getActualLocation()

```
public javax.tv.locator.Locator getActualLocation()
```

Return the actualLocation as provided when this ServiceReference was constructed.

#### Returns:

a Locator.

#### getLocator()

```
public javax.tv.locator.Locator getLocator()
```

Gives the transport-dependent locator that will be used to represent this service through the Java TV APIs, both in this xlet and in others.

#### Overrides:

`getLocator` in class `ServiceReference`

#### getServiceIdentifier()

```
public java.lang.String getServiceIdentifier()
```

#### Description copied from class:

```
org.dvb.spi.selection.ServiceReference
```

Return the transport independent locator of the service.

#### Overrides:

`getServiceIdentifier` in class `ServiceReference`

#### Returns:

the serviceIdentifier string passed into the constructor.

# org.dvb.spi.selection

# LocatorScheme

### Declaration

```
public class LocatorScheme

java.lang.Object
  |
  +--org.dvb.spi.selection.LocatorScheme
```

### Description

This class is used to represent a locator scheme that is supported by a SelectionProvider.

### Since:

MHP 1.1.3.

### Constructors

### LocatorScheme(String, boolean)

```
public LocatorScheme(java.lang.String scheme, boolean providerFirst)
```

Construct a LocatorScheme.

### Parameters:

`scheme` - the name of the locator scheme which is handled, not including the ":" character.

`providerFirst` - true if this provider shall handle the scheme before any handler in the platform implementation, false if this provider shall handle the scheme after any handler in the platform implementation and only for locators not handled by the platform implementation.

### Methods

### getProviderFirst()

```
public boolean getProviderFirst()
```

Returns whether the provider is to handle the scheme before any platform handler for the scheme.

### Returns:

true if this provider shall handle the scheme before any handler in the platform implementation, false if this provider shall handle the scheme after any handler in the platform implementation and only for locators not handled by the platform implementation.

### getScheme()

```
public java.lang.String getScheme()
```

Returns the scheme.

### Returns:

scheme the name of the locator scheme which is handled, not including the ":" character.

# org.dvb.spi.selection

# SelectionProvider

### Declaration

```
public interface SelectionProvider extends org.dvb.spi.SystemBoundProvider
```

### All Superinterfaces:

```
org.dvb.spi.Provider, org.dvb.spi.SystemBoundProvider
```

### Description

A provider of service selection. For example, a SelectionProvider can be installed in an always-running service-unbound xlet to provide the ability to reach services that do not have standarized SI signalling in an IPTV network.

The process for determining which Provider (if any) shall be used to present a transport independent service is defined by the following steps in the order given.

1) Consult the service list. If the service appears in the service list then make an selection between the available sources of that Service as defined by ServiceContext.select(Service).

2) If there one Provider registered to support the locator scheme of the locator for the Service, ask that Provider to present that service. The service selection operation shall fail if the provider cannot present it.

3) If more than one Provider has registed as supporting the locator scheme of the locator for the Service, offer the Service to each Provider in turn until either one of them does not fail or no more Providers are available in which case the service selection operation shall fail.

NOTE: If the performance penalty of polling multiple Providers for the same locator scheme is undesirable, service providers / operators using multiple Providers should ensure they use different Locator schemes.

Providers should select the transport dependent locators returned such that there is no collision the transport dependent locators returned by other Providers that may offer the same transport independent service. Where a GEM terminal is managed by a service provider / operator, the service provider / operator should enforce this if they permit more than one Provider to be installed.

Where there is a collision and a transport dependent locator can be accessed by more than one Provider, the selection of which to use is implementation dependent.

A SelectionProvider may inform the implementation of the actual location of a service by two mechanisms.

- For services which are already present in the network, the actual location can be passed in ServiceReference objects. These can be provided by the getServiceList method when a provider is first installed and later by the SelectionProviderContext. For these services, the implementation shall use this actual location to present the service without calling the select method. A session shall be created using the ServiceReference used to present the service in order that the provider can be notified when the service is no longer used. The session instance should be created once the platform has started finding the service (e.g. during tuning or after sending an IGMP join request) in order to avoid delaying the presentation of the service.

- For services whose actual location is not passed in a ServiceReference (e.g. services not already present in the network), the implementation shall use the select method to obtain the actual location.

The mechanism by which the actual location of a service is determined may change dynamically, e.g. as the set of services available in the network changes.

### Since:

MHP 1.1.3.

### See Also:

```
org.dvb.spi.Provider.getServiceProviderInterfaces()
```

### Methods

#### getServiceDescriptions(ServiceReference[])

```
public org.dvb.spi.selection.ServiceDescription[]
getServiceDescriptions(org.dvb.spi.selection.ServiceReference[] services)
```

Called by the terminal to request service description information from an SI source. The output shall be returned in an array of the same size and order as the input. If information is not available on any service listed in the input then the corresponding entry in the results array shall be null.

Parameters:

`services` - References identifying the services whose descriptions are requested.

Returns:

an array of service descriptions.

#### getServiceList()

```
public org.dvb.spi.selection.ServiceReference[] getServiceList()
```

Give a list of the services provided by this provider. The services returned from this method shall be merged into the platform service list as returned by javax.tv.service.SIManager.filterServices (null). Where the transport independent identification of a service is equal to one already in the service list then that transport independent service shall acquire an additional transport dependent service. Where the transport independent identification of a service is not equal to one already in the service list, a new service shall be added to the service list and SIChangeEvents generated to appropriate listeners.

The list of services returned shall include both those where the actual location is returned in the ServiceReference and those where the actual location is to be returned from a later call to the select method.

Returns:

a list of ServiceReference instances.

See Also:

`SelectionProviderContext.serviceListChanged(ServiceReference[])`

#### getSupportedLocatorSchemes()

```
public org.dvb.spi.selection.LocatorScheme[] getSupportedLocatorSchemes()
```

Returns the list of locator schemes handled by this provider. This list should not change over time; it is expected that platforms will usually call this method exactly once, after installation of a provider. There is no method to unregister a scheme other than unregistering the provider.

#### init(SelectionProviderContext)

```
public void init(org.dvb.spi.selection.SelectionProviderContext c)
```

Called by the platform to register its handler for events originating in the provider. This method shall be called after providerRegistered and before any other methods on the provider are called. Only one handler can be registered at any one time.

Parameters:

`c` - the handler.

#### newSession(ServiceReference)

```
public org.dvb.spi.selection.SelectionSession newSession(org.dvb.spi.selection.ServiceReference
service)
```

Called by the platform to create a session to manage the presentation of a service. A new session shall be created for each service to be presented by a provider.

Parameters:

`service` - the service whose presentation is managed through this session.

Returns:

a session.

# org.dvb.spi.selection

# SelectionProviderContext

## Declaration

```
public interface SelectionProviderContext
```

## Description

Platform implementations that wish to receive notifications from a `SelectionProvider` register an instance that implements this interface with the `SelectionProvider`.

Since:

MHP 1.1.3.

## Methods

### serviceDescriptionAvailable(ServiceReference[])

```
public void serviceDescriptionAvailable(org.dvb.spi.selection.ServiceReference[] serviceReferences)
```

Called by a source when service description information is available to offer that information to the platform implementation. The new list replaces any previous list.

Parameters:

`serviceReferences` - The ServiceReference instances for which descriptions are available. This array must not change after this method is invoked.

### serviceListChanged(ServiceReference[])

```
public void serviceListChanged(org.dvb.spi.selection.ServiceReference[] serviceReferences)
```

Called by a source when the list of services changes. The new list replaces any previous list.

The services passed into this method shall be compared with the service list returned from the call to getServiceList when this provider was first registered. Where services are added, these shall be merged into the platform service list as defined in the description of the getServiceList method. Where services are removed, if the transport independent service is left with no transport dependent services then it shall be removed from the platform service list. In all cases where the platform service list changes, SIChangeEvents shall be generated to appropriate listeners.

Parameters:

`serviceReferences` - The ServiceReference instances for the available services. This array must not change after this method is invoked.

### updateService(ServiceReference)

```
public void updateService(org.dvb.spi.selection.ServiceReference service)
```

Called by a source to update the details of a service it supports. This includes changing the service between one whose location is already known and one whose location must be found when it is selected. It also includes changing the location of a service whose location is already known. ServiceReferences shall be matched using the transport independent and transport dependent names.

Parameters:

`service` - the new service reference.

Throws:

`java.lang.IllegalArgumentException` - if a service with the same service identifier and transport independent locator has not been previously returned by this source to the implementation.

# org.dvb.spi.selection

# SelectionSession

Declaration

```
public interface SelectionSession
```

Description

A session for presentation of one or more services over a period of time by a SelectionProvider. The first operation on a new session will always be selection of a service in a service context, and all subsequent operations will pertain to the same service context.

Since:

MHP 1.1.3.

Methods

destroy()

```
public void destroy()
```

Called by the platform when the service bound to this session is no longer being used by the implementation. The selection provider should do any needed clean-up here, e.g. informing a server that the service is not needed any more. Once destroy is called, the terminal implementation discards the SelectionSession.

select()

```
public org.davic.net.Locator select()
```

Sets up delivery of a stream representing the service, and delivers a locator for reception of that stream. For example, either a unicast locator or a frequency locator might be returned, under the apprlpriate circumstances.

Returns:

a locator for where the stream can be found.

selectionReady()

```
public void selectionReady()
```

Called when the implementation is ready to receive content on the locator returned by the select method. When using unicast protocols where the content must only be sent when the GEM terminal is ready, it is now safe to request the content be sent.

A SelectionSession might be destroyed after a select, but before this method is called. In this case, this method may not be called. Applications should therefore not expect this method to be called after destroy() is called on this session.

See Also:

`destroy()`

### setPosition(long)

```
public long setPosition(long position)
```

Set the position within the media. If this session does not support trick modes, this method returns -1 and has no effect.

Parameters:

`position` - The position within the program, in milliseconds.

Returns:

The new position, or -1 if position setting isn't supported.

### setRate(float)

```
public float setRate(float newRate)
```

Sets the speed of playback. If this session does not support trick modes, this method returns 1 and has no effect. Calling this method with the value 1.0f shall always succeed, and shall result in a return value of 1.0f.

Parameters:

`newRate` - New playback rate. Implementations shall make a best effort to approximate this rate.

Returns:

The new rate, if known. If not known, the value java.lang.Float.NEGATIVE_INFINITY is returned.

# org.dvb.spi.selection

# ServiceDescription

### Declaration

```
public interface ServiceDescription
```

### Description

Represents the description of a service.

Since:

MHP 1.1.3.

### Methods

### getDeliverySystemType()

```
public javax.tv.service.navigation.DeliverySystemType getDeliverySystemType()
```

Return the type of the delivery system.

Returns:

the type of the delivery system by which this service is delivered.

Throws:

`java.lang.IllegalArgumentException` - if the returned ServiceType is not from a class loaded by the system classloader.

### getLongName(String)

```
public org.dvb.spi.util.MultilingualString getLongName(java.lang.String perferredLanguage)
```

Return the long name of this service.

Returns:

the name.

getServiceType()

```
public javax.tv.service.ServiceType getServiceType()
```

Return the type of this service. The service type returned shall be from a class loaded by the system classloader.

Returns:

the service type of this service.

Throws:

`java.lang.IllegalArgumentException` - if the returned ServiceType is not from a class loaded by the system classloader.

# org.dvb.spi.selection

# ServiceReference

Declaration

```
public class ServiceReference
```

```
java.lang.Object
  |
  +--org.dvb.spi.selection.ServiceReference
```

Direct Known Subclasses:

`KnownServiceReference`

Description

A reference to a service that is reached with the mediation of a SelectionProvider.

Since:

MHP 1.1.3.

See Also:

`SelectionProvider`

Constructors

ServiceReference(String, String)

```
public ServiceReference(java.lang.String transportIndependent, java.lang.String transportDependent)
```

Create a ServiceReference for a service. The two strings shall both be the external form of Locators, for example a transport independent "dvb:" locator string and provider specific locator string.

Parameters:

`transportIndependent` - The transport-independent locator of the service.

`transportDependent` - The transport-dependent locator of the service.

### Methods

#### getLocator()

```
public javax.tv.locator.Locator getLocator()
```

Gives the transport-dependent locator of the service

Returns:

the trasnport dependent string passed into the constructor.

#### getServiceIdentifier()

```
public java.lang.String getServiceIdentifier()
```

Return the transport independent locator of the service

Returns:

the transportIndependent string passed into the constructor.

# Package org.dvb.spi.si.full

### Description

This package defines an interface for providing full service information, as accessed by javax.tv.service.SIManager. This package defines an xlet-bound provider. Since it's an xlet-bound manager, every xlet that needs the services of this kind of provider needs to carry its own implementation of the provider. It is envisaged that the provider defined in the package `org.dvb.si.simple` will be used more frequently.

| Class Summary | |
|---|---|
| **Interfaces** | |
| SIManagerProvider | This xlet-bound provider allows one to take complete control of the view of SI exposed to the xlet to which it is bound. |

# org.dvb.spi.si.full SIManagerProvider

### Declaration

```
public interface SIManagerProvider extends org.dvb.spi.XletBoundProvider
```

### All Superinterfaces:

```
org.dvb.spi.Provider, org.dvb.spi.XletBoundProvider
```

### Description

This xlet-bound provider allows one to take complete control of the view of SI exposed to the xlet to which it is bound. The author of the provider is able to directly implement Java TV's SIManager interface, and provide that instance to the implementation. The xlet to which this provider is bound will get this instance of SIManager when it calls the Java TV method to retrieve an SIManager.

### See Also:

```
javax.tv.service.SIManager.createInstance()
```

### Methods

#### createSIManager(SIManager)

```
public javax.tv.service.SIManager createSIManager(javax.tv.service.SIManager parent)
```

Gets an SI manager. This will typically be given directly to the xlet (when it calls SIManager.createInstance()). This method will be called each time the xlet calls SIManager.createInstance().

Parameters:

`parent` - The "system" SIManager that would have been given to the xlet, if a provider had not been installed. The provider's SIManager may consult its parent, e.g. for terrestrial services in a hybrid terminal.

# Package org.dvb.spi.si.simple

### Description

This package defines interfaces for providing service information, as accessed by SIManager. This package defines a system-bound provider. As a system-bound provider, only one provider needs to be registered with the GEM terminal, and this one provider services multiple xlets. It provides the minimal SI believed to be adequate to support service-bound xlets that rely on the standard Java TV SI access mechanisms.

| Class Summary | |
|---|---|
| **Interfaces** | |
| `ProgramDescription` | Encapsulates the description of a program. |
| `ScheduleDescription` | Encapsulates the description of a schedule for a service (sometimes colloquially known as a "TV channel"). |
| `SimpleSIProvider` | This interface describes a source of information about the schedules and program events in a broadcast channel. |
| `SimpleSIProviderListener` | An instance of this class is given to a SimpleSIProvider when registered so that the provider can notify the platform of events that happen. |
| **Classes** | |
| `ProgramReference` | This class represents a reference to a program within a service that is accessed using a SelectionProvider. |
| `ProgramReferenceWithTimes` | Wraps a program reference with its start and end times. |

# org.dvb.spi.si.simple ProgramDescription

### Declaration

```
public interface ProgramDescription
```

### Description

Encapsulates the description of a program. One instance of ProgramDescription represents a static snapshot of the information about a single program.

The program descriptions may be relatively large. Implementations with limited memory resources should be careful not to ask for many descriptions at one time and not to keep references to descriptions longer than needed.

### Methods

#### getDescription(String)

```
public org.dvb.spi.util.MultilingualString getDescription(java.lang.String preferredLanguage)
```

Return the description of the program.

Parameters:

`preferredLanguage` - the preferred language for the description.

Returns:

the description.

#### getEndTime()

```
public java.util.Date getEndTime()
```

Return the end time for the program.

Returns:

the end time.

#### getName(String)

```
public org.dvb.spi.util.MultilingualString getName(java.lang.String preferredLanguage)
```

Return the name of the program.

Parameters:

`preferredLanguage` - the preferred language for the name.

Returns:

the name.

#### getStartTime()

```
public java.util.Date getStartTime()
```

Return the start time for the program.

Returns:

the start time.

# org.dvb.spi.si.simple

# ProgramReference

### Declaration

```
public final class ProgramReference
```

```
java.lang.Object
  |
  +--org.dvb.spi.si.simple.ProgramReference
```

### Description

This class represents a reference to a program within a service that is accessed using a SelectionProvider.

See Also:

org.dvb.spi.selection.SelectionProvider, SimpleSIProvider.getProgramReferences(ServiceReference), ProgramDescription

### Constructors

#### ProgramReference(ServiceReference, String)

```
public ProgramReference(org.dvb.spi.selection.ServiceReference service,
java.lang.String programIdentifier)
```

Constructs a reference to a program.

Parameters:

service - the service containing the program.

programIdentifier - an identifier for the program within the service.

### Methods

#### getProgramIdentifier()

```
public java.lang.String getProgramIdentifier()
```

Get the identifier of the program.

Returns:

a program identifier.

#### getServiceReference()

```
public org.dvb.spi.selection.ServiceReference getServiceReference()
```

Get the service containing the program as passed to the constructor.

Returns:

a service.

# org.dvb.spi.si.simple

# ProgramReferenceWithTimes

### Declaration

```
public final class ProgramReferenceWithTimes

java.lang.Object
  |
  +--org.dvb.spi.si.simple.ProgramReferenceWithTimes
```

### Description

Wraps a program reference with its start and end times.

### Constructors

#### ProgramReferenceWithTimes(ProgramReference, Date, Date)

```
public ProgramReferenceWithTimes(org.dvb.spi.si.simple.ProgramReference r, java.util.Date start,
java.util.Date end)
```

Construct the program reference with times.

Parameters:

r - the program reference.

start - the (inclusive) start time of the program.

end - the (exclusive) end time of the program.

Methods

getEndTime()

```
public java.util.Date getEndTime()
```

Return the end time of the program as passed in to the constructor.

Returns:

a time.

getProgramReference()

```
public org.dvb.spi.si.simple.ProgramReference getProgramReference()
```

Return the program reference as passed in to the constructor.

Returns:

a program reference.

getStartTime()

```
public java.util.Date getStartTime()
```

Return the start time of the program as passed in to the constructor.

Returns:

a time.

# org.dvb.spi.si.simple

# ScheduleDescription

Declaration

```
public interface ScheduleDescription
```

Description

Encapsulates the description of a schedule for a service (sometimes colloquially known as a "TV channel"). One instance of ScheduleDescription represents a static snapshot of one time-bounded part of the schedule information. The schedule shall be sorted from earliest to latest within the schedule.

Methods

getPrograms()

```
public java.util.List getPrograms()
```

Return identifiers for the programs in the service, along with the start and end times.

Returns:

a list of ProgramReferenceWithTime instances to the programs in the service sorted in time order from first to last. This list instance must not change after this method is invoked.

# org.dvb.spi.si.simple

# SimpleSIProvider

### Declaration

```
public interface SimpleSIProvider extends org.dvb.spi.SystemBoundProvider
```

### All Superinterfaces:

```
org.dvb.spi.Provider, org.dvb.spi.SystemBoundProvider
```

### Description

This interface describes a source of information about the schedules and program events in a broadcast channel.

This class and related classes and interfaces are intended to be used as follows:

- When new program data is available, the SI source calls the method on SimpleSIProviderContext to tell the platform which entities have data available about them.
- The platform can, at any time, ask the provider for information on certain entities. This may be in response to the provider telling the platform that new information is available. It may be in response to an application asking for information about an entity whose description is not available in the terminal.

In this API, collections of data are represented with random-access java.util.List type.

This class involves several methods that pass an instance of List to the terminal. Once a list instance has been give to the terminal, it must not be changed. The results of changing a list that has been given to the terminal via this API are undefined, and may include termination of the enclosing xlet.

### See Also:

`org.dvb.spi.SystemBoundProvider`, `org.dvb.spi.selection.SelectionProvider`, `java.util.List`, `java.util.ArrayList`

### Methods

#### getProgramDescriptions(ProgramReference[], String[])

```
public org.dvb.spi.si.simple.ProgramDescription[]
getProgramDescriptions(org.dvb.spi.si.simple.ProgramReference[] programReferences,
java.lang.String[] preferredLanguages)
```

Called by a terminal to request program description information from a provider. The output shall be returned in an array of the same size and order as the input. If the schedule is not available on any service listed in the input then the corresponding entry in the results array shall be null.

#### Parameters:

`programReferences` - references identifying the programs whose schedules are requested.

`preferredLanguages` - The preferred language(s) for program names and descriptions. This is a hint of the language(s) likely to be requested of the descriptions themselves.

#### Returns:

an array of program descriptions.

#### getProgramReferences(ServiceReference)

```
public org.dvb.spi.si.simple.ProgramReference[]
getProgramReferences(org.dvb.spi.selection.ServiceReference r)
```

Called by the terminal to request program information from a provider for a given service.

Parameters:

r - the service for which program information is requested.

Returns:

references for all of the known programs for the given service.

### getScheduleDescriptions(ServiceReference[], Date, Date)

```
public org.dvb.spi.si.simple.ScheduleDescription[]
getScheduleDescriptions(org.dvb.spi.selection.ServiceReference[] services, java.util.Date startDate,
java.util.Date endDate)
```

Called by the terminal to request schedule information from a provider. The output shall be returned in an array of the same size and order as the input. If the schedule is not available on any service listed in the input then the corresponding entry in the results array shall be null.

Parameters:

services - references identifying the services whose schedules are requested.

startDate - the starting date and time from which to return schedule descriptions (inclusive).

endDate - the finishing date and time up to which schedule descriptons should be returned (exclusive).

Returns:

an array of schedule descriptions.

### registerListener(SimpleSIProviderListener)

```
public void registerListener(org.dvb.spi.si.simple.SimpleSIProviderListener l)
```

Called by the platform in order to obtain events from this provider.

Parameters:

l - the listener to register.

### unregisterListener(SimpleSIProviderListener)

```
public void unregisterListener(org.dvb.spi.si.simple.SimpleSIProviderListener l)
```

Called by the platform when it no longer wishes to receive events from this provider.

Parameters:

l - the listener to remove.

# org.dvb.spi.si.simple

# SimpleSIProviderListener

### Declaration

```
public interface SimpleSIProviderListener
```

### Description

An instance of this class is given to a SimpleSIProvider when registered so that the provider can notify the platform of events that happen.

See Also:

SimpleSIProvider.registerListener(SimpleSIProviderListener)

### Methods

#### scheduleDescriptionAvailable(List)

```
public void scheduleDescriptionAvailable(java.util.List serviceReferences)
```

Called by providers when schedule information is available to offer that information to the terminal.

Parameters:

`serviceReferences` - A list of ServiceReference instances for which descriptions are available. This list instance must not change after this method is invoked.

# Package org.dvb.spi.util

### Description

Utility classes used by other providers.

| Class Summary | |
|---|---|
| **Classes** | |
| `MultilingualString` | A string that has an associated language code. |

# org.dvb.spi.util MultilingualString

### Declaration

```
public final class MultilingualString
```

```
java.lang.Object
  |
  +--org.dvb.spi.util.MultilingualString
```

### Description

A string that has an associated language code.

Since:

MHP 1.1.3.

### Constructors

#### MultilingualString(String, String)

```
public MultilingualString(java.lang.String s, java.lang.String languageCode)
```

Initialize a new MultilingualString.

Parameters:

`s` - The string that is in the given language.

`languageCode` - The language code encoded as per ISO 639-2 [84].

### Methods

#### getLanguageCode()

```
public java.lang.String getLanguageCode()
```

Return the language code.

Returns:

The language code encoded as per ISO 639-2 [84].

#### getString()

```
public java.lang.String getString()
```

Return the string.

Returns:

The string that is in the given language.

#### toString()

```
public java.lang.String toString()
```

Return a string representation of this object.

Overrides:

`toString` in class `Object`

Returns:

a string.

# Package org.dvb.spi.ict

### Description

This package enables the GEM return channel application download mechanism to be used with non-standard, proprietary or market specific protocols.

| Class Summary | |
|---|---|
| **Interfaces** | |
| AsyncResourceLoadListener | This interface should be implemented by classes within the GEM implementation that will be used as listeners for asynchronous loading of resources by interaction channel transport service providers. |
| InteractionChannelTransportProvider | This interface will be implemented by system providers of resource transport via the interaction channel. |
| ResourceTransportObject | This interface represents a reference to a resource that will be downloaded using an interaction channel transport service provider. |
| ResourceUpdateListener | This interface will be implemented by listeners that are part of the GEM implementation so that they are notified of updates to resources downloaded by an interaction channel tranport service provider. |
| **Classes** | |
| AsyncResourceLoadEvent | Base class for events related to progress of loading an event by an interaction channel transport service provider. |
| LoadFailedEvent | Event indicating that the load of a resource failed. |
| LoadFinishedEvent | Event indicating that the load of a resource was completed successfully. |
| ResourceObjectUpdateEvent | Event object used to notify update listeners that a network resource has changed (i.e. a new version is available). |
| **Exceptions** | |
| ResourceLoadException | This exception is thrown when an attempt to load a resource by an interaction channel transport service provider fails. |
| ResourceNotFoundException | This exception is thrown when an attempt is made to load a resource via interaction channel transport, and the provider determines that the resource does not exist. |

# org.dvb.spi.ict

# AsyncResourceLoadEvent

### Declaration

```
public class AsyncResourceLoadEvent extends java.util.EventObject

java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.spi.ict.AsyncResourceLoadEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Direct Known Subclasses:

```
LoadFailedEvent, LoadFinishedEvent
```

### Description

Base class for events related to progress of loading an event by an interaction channel transport service provider.

### Constructors

#### AsyncResourceLoadEvent(ResourceTransportObject, boolean)

```
protected AsyncResourceLoadEvent(org.dvb.spi.ict.ResourceTransportObject source, boolean loaded)
```

Create a new AsyncResourceLoadEvent object.

### Parameters:

`source` - the ResourceTransportObject that is the source of this event.

`loaded` - flag indicating whether the resource is loaded as of the receipt of this event.

### Methods

#### isLoaded()

```
public boolean isLoaded()
```

Return a flag indicating whether the resource is loaded.

#### Returns:

boolean.

# org.dvb.spi.ict AsyncResourceLoadListener

### Declaration

```
public interface AsyncResourceLoadListener extends java.util.EventListener
```

### All Superinterfaces:

```
java.util.EventListener
```

### Description

This interface should be implemented by classes within the GEM implementation that will be used as listeners for asynchronous loading of resources by interaction channel transport service providers.

Note that in the case where a DVB-J application calls asynchronousLoad() on a DSMCCObject that refers to an interaction channel file system file, the GEM implementation will be respsonsible for mapping receiveEvent method calls from this method to calls on the corresponding AsynchronousLoadingEventListener (i.e. applications will not be able to reference the service provider classes directly).

### Methods

#### receiveEvent(AsyncResourceLoadEvent)

```
public void receiveEvent(org.dvb.spi.ict.AsyncResourceLoadEvent e)
```

Notify this listener of an update in the progress of loading a network resource. The specified event object contains detailed information about the update.

#### Parameters:

`e` - the event object containing details of the update.

# org.dvb.spi.ict InteractionChannelTransportProvider

### Declaration

```
public interface InteractionChannelTransportProvider extends org.dvb.spi.SystemBoundProvider
```

### All Superinterfaces:

```
org.dvb.spi.Provider, org.dvb.spi.SystemBoundProvider
```

### Description

This interface will be implemented by system providers of resource transport via the interaction channel. It will be used by the GEM implementation when accessing resources identified by references (URIs) encountered within the interaction channel protocol descriptor. In addition, this provider may be accessed indirectly from DVB-J applications via the java.io.File and org.dvb.dsmcc.DSMCCObject classes.

### Methods

#### createResourceTransportObject(String)

```
public org.dvb.spi.ict.ResourceTransportObject createResourceTransportObject(java.lang.String uri)
```

Create new resource access object in preparation for downloading and using the contents of a network resource via the interaction channel.

Parameters:

`uri` - URI identifying the resource to be accessed, the URI scheme of which is guaranteed to be one of those returned from getSupportedURISchemes.

Returns:

ResourceTransportObject - the object used to track download of the resource.

#### exists(String)

```
public boolean exists(java.lang.String uri)
```

Return a flag indicating whether the specified resource exists. The expectation is that this method will provide a more efficient indication of existence than creating a ResourceTransportObject and initiating a download, for service provider implementations that support caching of directory lookup information.

Parameters:

`uri` - URI identifying the resource whose existence to check, the URI scheme of which is guaranteed to be one of those returned from `getSupportedURISchemes()`

Returns:

boolean - Flag indicating whether the specified resource exists. Note that there is no guarantee that the resource will subsequently be loaded successfully, since the return value will be determined just from the directory information. It is possible that the resource will be found not to exist during the process of downloading it.

Throws:

`java.lang.UnsupportedOperationException` - if this provider does not support optimized existence checks (i.e. if `existsSupported()` returns `false`).

#### existsSupported()

```
public boolean existsSupported()
```

Return a flag indicating whether this provider supports existence checks using the exists() method on the provider implementation. This is intended as a short circuit by which the GEM implementation may avoid the overhead of creating a ResourceTransportObject instance in cases where the provider is capable of determining the existence of a resource without actually trying to load it (e.g. broadcast transport mechanisms where there may be cached directory lookup information for the resources). The goal is to optimize the performance of the logical file system search path.

Returns:

boolean - Flag indicating whether fast existence checks are supported via the `exists()` method.

### getSupportedURISchemes()

```
public java.lang.String[] getSupportedURISchemes()
```

Return the list of URI schemes supported by this provider. This method will be called by the GEM implementation exactly once immediately subsequent to registration of the provider. The GEM implementation will only call createResourceTransport on this provider for URI schemes that are returned by this method. In the case where multiple providers return the same URI scheme, the provider that will be used will be the one that most recently registered.

Note that if http or https are contained in the returned list they will be ignored. In other words, the service provider cannot override the GEM implementation's internal implementation of the HTTP and HTTPS protocols.

#### Returns:

String[] - list of supported URI schemes, each of which must consist of alphabetic characters only (no trailing ":").

### prefetch(String, int)

```
public void prefetch(java.lang.String uri, int priority)
```

Provides a hint to the provider that the resource identified by the specified URI will probably be accessed in the near future, so that if available resources permit and the provider supports prefetching it can take whatever steps are appropriate to speed up the time required to load the file when the subsequent request to load the file is made via the normal call to `ResourceTransportObject.synchronousLoad()` or `ResourceTransportObject.asynchronousLoad()`. This could include anything from retrieving directory information to actually downloading the file to memory or local physical file system, if space is available.

#### Parameters:

`uri` - URI identifying the resource to be accessed, the URI scheme of which is guaranteed to be one of those returned from getSupportedURISchemes.

`priority` - hint to the provider about the relative priority of this resources with respect to other resources prefetch hints for this provider (higher = more important).

#### Throws:

`java.lang.UnsupportedOperationException` - if this provider does not support prefetching (i.e. if `prefetchSupported()` returns `false`).

### prefetchSupported()

```
public boolean prefetchSupported()
```

Return a flag indicating whether this provider supports prefetch. Note that a return value of `true` does not guarantee that a call to `prefetch()` for a particular resource will have any effect. It is merely an indication of whether the provider supports prefetch hints in general.

#### Returns:

boolean - Flag indicating whether this provider supports prefetch hints via the `prefetch()` method.

# org.dvb.spi.ict
# LoadFailedEvent

### Declaration

```
public final class LoadFailedEvent extends AsyncResourceLoadEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.spi.ict.AsyncResourceLoadEvent
              |
              +--org.dvb.spi.ict.LoadFailedEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

Event indicating that the load of a resource failed.

### Constructors

#### LoadFailedEvent(ResourceTransportObject, Exception)

```
public LoadFailedEvent(org.dvb.spi.ict.ResourceTransportObject source, java.lang.Exception reason)
```

Construct a new event indicating failure to load a resource.

### Parameters:

`source` - The ResourceTransportObject that was the source of the failure.

`reason` - Exception indicating the reason for the failure (may be null).

### Methods

#### getReason()

```
public java.lang.Exception getReason()
```

Return the reason for the load failure.

### Returns:

Exception - The exception object indicating the reason for the failure (may be null).

# org.dvb.spi.ict
# LoadFinishedEvent

### Declaration

```
public final class LoadFinishedEvent extends AsyncResourceLoadEvent
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.spi.ict.AsyncResourceLoadEvent
              |
              +--org.dvb.spi.ict.LoadFinishedEvent
```

### All Implemented Interfaces:

`java.io.Serializable`

### Description

Event indicating that the load of a resource was completed successfully.

### Constructors

#### LoadFinishedEvent(ResourceTransportObject)

```
public LoadFinishedEvent(org.dvb.spi.ict.ResourceTransportObject source)
```

Construct a new event indicating that the load of a resource was completed successfully.

### Parameters:

`source` - The ResourceTransportObject that was the source of this event.

#### LoadFinishedEvent(ResourceTransportObject, boolean)

```
public LoadFinishedEvent(org.dvb.spi.ict.ResourceTransportObject source, boolean loaded)
```

Construct a new event indicating that the load of a resource was completed successfully.

### Parameters:

`source` - The ResourceTransportObject that was the source of this event.

`loaded` - flag indicating whether the resource is loaded as of the receipt of this event.

# org.dvb.spi.ict

# ResourceLoadException

### Declaration

```
public class ResourceLoadException extends java.lang.Exception
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.spi.ict.ResourceLoadException
```

### All Implemented Interfaces:

`java.io.Serializable`

### Direct Known Subclasses:

`ResourceNotFoundException`

### Description

This exception is thrown when an attempt to load a resource by an interaction channel transport service provider fails.

### Constructors

#### ResourceLoadException()

```
public ResourceLoadException()
```

Create a ResourceLoadException.

### ResourceLoadException(String)

```
public ResourceLoadException(java.lang.String message)
```

Create a ResourceLoadException.

#### Parameters:

`message` - describes the failure.

### ResourceLoadException(String, Throwable)

```
public ResourceLoadException(java.lang.String message, java.lang.Throwable cause)
```

Create a ResourceLoadException.

#### Parameters:

`message` - describes the failure.

`cause` - details the cause of the failure.

### ResourceLoadException(Throwable)

```
public ResourceLoadException(java.lang.Throwable cause)
```

Create a ResourceLoadException.

#### Parameters:

`cause` - details the cause of the failure.

# org.dvb.spi.ict

# ResourceNotFoundException

### Declaration

```
public class ResourceNotFoundException extends ResourceLoadException
```

```
java.lang.Object
  |
  +--java.lang.Throwable
        |
        +--java.lang.Exception
              |
              +--org.dvb.spi.ict.ResourceLoadException
                    |
                    +--org.dvb.spi.ict.ResourceNotFoundException
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

This exception is thrown when an attempt is made to load a resource via interaction channel transport, and the provider determines that the resource does not exist.

### Constructors

### ResourceNotFoundException(String)

```
public ResourceNotFoundException(java.lang.String uri)
```

Create a ResourceNotFoundException.

Parameters:

uri - the URI for the resource that was not found.

ResourceNotFoundException(String, String)

```
public ResourceNotFoundException(java.lang.String uri, java.lang.String message)
```

Create a ResourceNotFoundException.

Parameters:

message - describes the failure.

# org.dvb.spi.ict ResourceObjectUpdateEvent

Declaration

```
public class ResourceObjectUpdateEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.spi.ict.ResourceObjectUpdateEvent
```

All Implemented Interfaces:

```
java.io.Serializable
```

Description

Event object used to notify update listeners that a network resource has changed (i.e. a new version is available).

Constructors

ResourceObjectUpdateEvent(ResourceTransportObject, int)

```
public ResourceObjectUpdateEvent(org.dvb.spi.ict.ResourceTransportObject source, int newVersion)
```

Create a ResourceObjectUpdateEvent indicating that a new version of the resource has been detected. Note that the new version of the resource is not automatically downloaded; that is left to the user of this object.

Parameters:

source - the ResourceTransportObject whose version has changed.

newVersion - the new version number.

Methods

getNewVersion()

```
public int getNewVersion()
```

Return the newly available resource version number (i.e. the version that would be downloaded now if the resource were loaded again).

Returns:

int

# org.dvb.spi.ict

# ResourceTransportObject

### Declaration

```
public interface ResourceTransportObject
```

### Description

This interface represents a reference to a resource that will be downloaded using an interaction channel transport service provider. It exposes a set of methods that correspond with those in org.dvb.dsmcc.DSMCCObject, so that the applications can use DSMCCObject to reference resources that will be downloaded by the GEM implementation via the service provider implementation.

Like DSMCCObject, ResourceTransportObject instances can either be in a loaded or unloaded state. When in a loaded state, the contents of the resource can be accessed either directly from memory as a byte array or else from a physical file in a local file system (could also be a RAM-based file system).

Note that unlike DSMCCObject instances, resource transport objects are only used to download resources with associated content (i.e. not for URIs that represent directories).

### Fields

#### LOAD_TYPE_FILE

```
public static final int LOAD_TYPE_FILE
```

Resource load type that indicates the file has been loaded to a local physical file system.

#### LOAD_TYPE_MEMORY

```
public static final int LOAD_TYPE_MEMORY
```

Resource load type that indicates the file has been loaded in memory as a byte array.

### Methods

#### abort()

```
public void abort()
```

This method is used to abort a load in progress. It can be used to abort either a synchronousLoad or an asynchronous-Load.

#### Throws:

`java.lang.IllegalStateException` - if load of the resource has not already been started.

#### addUpdateListener(ResourceUpdateListener)

```
public void addUpdateListener(org.dvb.spi.ict.ResourceUpdateListener listener)
```

Add a listener to receive notifications of updates to this resource. Note that update notifications will not be fired until after the object has succcessfully entered the loaded state for the first time.

Once an object has successfully entered the loaded state once, this event shall continue to be fired when changes are detected regardless of further transitions in or out of the loaded state.

#### Parameters:

`listener` - the listener to be added.

### asynchronousLoad(AsyncResourceLoadListener)

```
public void asynchronousLoad(org.dvb.spi.ict.AsyncResourceLoadListener listener)
```

Load the resource asynchronously. The specified listener will be notified of the loading progress, including subsequent successful completion or failure.

Parameters:

`listener` - the listener to be notified.

### exists()

```
public boolean exists()
```

Return a flag indicating whether the resource exists. Note that some providers may be able to determine resource existence without downloading the resource using cached directory lookup information, but in general this may not be the case. In cases where the provider must download the resource to determine resource existence, this call may result in a synchronous load operation being performed before returning.

Returns:

boolean.

### getLength()

```
public long getLength()
```

Return the length in bytes of content of this resource. Note that some providers may be able to determine the size without downloading the resource using cached directory lookup informatino, but in general this may not be the case. In cases where the provider must download the resource to determine the length, this call may result in a synchronous load operation being performed before returning. In addition, since this information may be populated from a directory, it is not guaranteed to be accurate; the most authoritative information is specified after download from either `getResource-Bytes().length` or `getResourceFile().length()`, depending on the load type.

Returns:

long the length in bytes of the resource content.

### getMimeType()

```
public java.lang.String getMimeType()
```

Return the MIME type of the resource. Note that some providers may be able to determine the MIME type without downloading the resource using cached directory lookup information, but in general this may not be the case. In cases where the provider must download the resource to determine the MIME type, this call may result in a synchronous load operation being performed before returning.

Returns:

String.

### getResourceBytes()

```
public byte[] getResourceBytes()
```

Return a byte array containing the downloaded resource. This method should only be called if the resource is loaded and `getResourceLoadType()` returns `LOAD_TYPE_MEMORY`.

Returns:

byte [] the resource data.

Throws:

`java.lang.IllegalStateException` - if the resource is not loaded or is loaded to a file.

### getResourceFile()

```
public java.io.File getResourceFile()
```

Return the resource file containing the downloaded resource. This method should only be called if the resource is loaded and `getResourceLoadType()` returns `LOAD_TYPE_FILE`.

The provider must ensure that the returned file is readable by the GEM implementation.

#### Returns:

File - the resource load file.

#### Throws:

`java.lang.IllegalStateException` - if the resource is not loaded or is loaded to a byte array in memory.

### getResourceLoadType()

```
public int getResourceLoadType()
```

Return a value indicating where the associated resource has been downloaded. Possible values include:

- `LOAD_TYPE_MEMORY` - indicates that the resource was loaded into memory as a byte array, and can be retrieved using the `getResourceBytes()` method. `LOAD_TYPE_FILE` - indicates that the resource was loaded to a file on a local physical file system, and can be retrieved using the `getResourceFile()` method.

#### Returns:

int - the load type.

#### Throws:

`java.lang.IllegalStateException` - if the resource is not loaded.

### getURI()

```
public java.lang.String getURI()
```

Return the URI identifying the resource to be downloaded.

#### Returns:

String the URI for this resource.

### isLoaded()

```
public boolean isLoaded()
```

Return a flag indicating whether the resource has been succesfully downloaded.

### isMetadataLoaded()

```
public boolean isMetadataLoaded()
```

Return a flag indicating whether the metadata for this resource (existence flag, content length, and MIME type) is available for retrieval via the corresponding accessor methods. Depending on the specifics of the provider implementation and transport mechanism, it may be that this information is available without needing to call the `loadMetadata` method; that method should be called only when this method returns `false`.

#### Returns:

boolean - Flag indicating whether the metadata for this resource has been loaded and is ready for retrieval via relevant accessor methods.

### loadMetadata(AsyncResourceLoadListener)

```
public void loadMetadata(org.dvb.spi.ict.AsyncResourceLoadListener listener)
throws ResourceLoadException
```

Load the resource metadata (existence flag, content length, and MIME type) for this resource. Note that some providers will be able to determine the metadata information without downloading the resource content by using cached directory lookup information, but in general this may not be the case. In cases where the provider must download the resource to find the directory information, this call may have an equivalent effect to calling `asynchronousLoad()`. If a load request is already in progress, this method may also block until the outstanding load request is complete.

This method may fail either asynchronously with notification event sent to the listener or else synchronously during this method call.

Once the metadata is loaded, the following methods may be called:

- `exists()`
- `getMimeType()`
- `getLength()`

Parameters:

`listener` - A listener to be notified of asynchronous loading events.

Throws:

`ResourceLoadException`

### removeUpdateListener(ResourceUpdateListener)

```
public void removeUpdateListener(org.dvb.spi.ict.ResourceUpdateListener listener)
```

Remove a listener that is currently receiving notifications of updates to this resource. If the specified listener has not previously been added, then calling this method has no effect.

Parameters:

`listener` - the listener to be removed.

### synchronousLoad()

```
public void synchronousLoad()
throws ResourceLoadException
```

Load the resource. When this method returns, the resource will have been succesfully downloaded. If the download fails, then a ResourceLoadException will be thrown.

Throws:

`ResourceLoadException`

### unload()

```
public void unload()
```

Tells the service provider that the downloaded resource object is no longer used, and therefore any resources allocated to this object can be freed. This method puts the ResourceTransportObject back into the unloaded state.

Throws:

`java.lang.IllegalStateException` - if the resource is not loaded.

# org.dvb.spi.ict

# ResourceUpdateListener

### Declaration

```
public interface ResourceUpdateListener
```

### Description

This interface will be implemented by listeners that are part of the GEM implementation so that they are notified of updates to resources downloaded by an interaction channel tranport service provider.

Note that in the case where a DVB-J application registers an ObjectChangeEventListener on a DSMCCObject that refers to an interaction channel file system file, the GEM implementation will be respsonsible for mapping update method calls from this method to calls to the ObjectChangeEventListener (i.e. applications will not be able to reference the service provider classe directly).

### Methods

#### resourceObjectUpdated(ResourceObjectUpdateEvent)

```
public void resourceObjectUpdated(org.dvb.spi.ict.ResourceObjectUpdateEvent e)
```

Notify the listener that a downloaded resource object has been updated. Details of the update are described in the specified event object.

### Parameters:

- `e` - the event object describing the change.

# Annex AO (normative): Services and the service list

## AO.1 Service list creation

The first entry point for services is the SIManager object. This object is responsible for aggregating services and delivering filtered lists of available services. It is assumed that when this object is instantiated it immediately begins to build the list of available services e.g. via the SD&S process to discover available Service Providers and their Service Offerings. The inputs to the list of available services shall be as follows:

- One input to the list of available services shall be all the services listed in the broadcast discovery information records for the selected service provider(s) - both "TS Full SI" and "TS - Optional SI". At least one of the service providers signalled in the service provider discovery records shall be selected. Support for simultaneously selecting more than one such service provider is optional however if more than one such service provider is selected simultaneously then services from all selected providers shall be used as input to the list of available services. Hence if the same service is available from multiple selected service providers then they shall appear in the service list as a single transport independent service.
- On GEM terminals supporting both classic DVB and IP network interfaces, a second input to the list of available services shall be the list discovered by the classical service scan (see clause Z.5, "Multiple service contexts in a GEM platform" and MHP [1], clause O.2.1, "javax.tv.service.Service").
- A third input to the list of available services shall be those available through currently registered selection providers (org.dvb.spi.selection.SelectionProvider).
- If services are discovered via the broadband content guide, these should not form an additional input to the list of available services.

The inputs to the list of available services shall be combined as follows:

- If more than one input contains the same service (identified by textual service identifier), the services shall be combined into a single entry in the available services list - even if they have different original_network_id and/or service_id.
- If more than one input contains the same service (identified by original network identifier and service identifier), the services shall be combined into a single entry in the available services list. The transport_stream_id shall be ignored for the purposes of this comparison. This comparison shall only happen between services without a textual_service_identifier.

## AO.2 SIManager

### AO.2.1 Introduction (informative)

The SIManager provides a single view over all the available SI sources for the GEM terminal. On GEM terminals supporting both classic DVB and IP network interfaces, information from both sources is merged into this single view.

## AO.2.2 Method definition

### AO.2.2.1 Methods without modified or extended semantics

The following list of methods have no modifications or extensions to their semantics defined by GEM.

- filterServices.
- getService.
- getRatingDimension.
- retrieveServiceDetails.
- registerInterest.
- retrieveProgramEvent.
- retrieveSIElement.

These methods shall support references to IP delivered content and services according to their specification.

For the methods retrieveProgramEvent and retrieveSIElement, on GEM terminals where BCG is not supported, where information from BCG is required in order to achieve the specified behaviour, the method call shall fail as specified for that method when that information is not available - even if that information would be available to GEM terminals supporting BCG.

### AO.2.2.2 getTransports

The getTransports method shall return a Transport object for the broadband (IPTV or OTT) content delivery mechanism which implements BouquetCollection, NetworkCollection and TransportStreamCollection. This is in addition to Transport objects for those classical DVB content delivery mechanisms included in the GEM terminal. The Transport object for IPTV/OTT shall be as follows:

- The BouquetCollection shall include all bouquets defined in the package discovery information record.
- The TransportStreamCollection is not required to be implemented.
- The NetworkCollection is not required to be implemented.

### AO.2.2.3 getSupportedDimensions

GEM does not require any rating dimensions to be available for IPTV/OTT delivered content.

See MHP [1], clause O.2.6.1 for the behaviour of this method in GEM terminals including a classical DVB network interface.

# AO.3 Services

This clause considers four different types of services:

- Services only signalled in SD&S (in the TS Full SI Broadcast Discovery Information Record) where the information about the service comes from the stream itself. These will be both transport independent and transport dependent.
- Services only signalled in SD&S (in the TS Partial SI Broadcast Discovery Information Record) where the information about the service comes from the SD&S entry for the service (except as defined by the primary SI source attribute - see clause AO.3.2.1, "General"). These will be both transport independent and transport dependent.

- Services only signalled in DVB-SI which shall be supported as defined in MHP [1], clause O.2.1, "javax.tv.service.Service".
- Services signalled in more than one mechanism which by definition are only transport independent.

A fifth type of service is found in GEM terminals supporting the broadband content guide option as defined in clause 15, "Detailed platform profile definitions". These are services signalled in SD&S (in the TS Partial SI Broadcast Discovery Information Record) for which information is also available via the BCG. These are addressed in more detail in clause AQ.2 "javax.tv.service.Service and javax.tv.service.navigation.ServiceDetails".

## AO.3.1 Services only signalled in TS Full SI Broadcast Discovery Information Record

### AO.3.1.1 getLocator

When called on a transport independent service, the getLocator method shall return a transport independent locator for the service as defined in clause 14.9, "Content referencing for IPTV/ OTT".

When called on a transport dependent service, the getLocator method shall return a transport dependent locator for the service as defined in clause 14.9, "Content referencing for IPTV/ OTT".

### AO.3.1.2 getName

The getName method shall return the name as defined in MHP [1], clause O.2.1.1, "getName". If no name is available (e.g. the service concerned has never been received by the GEM terminal) then a string of length zero shall be returned.

### AO.3.1.3 getServiceType

The getServiceType method shall return the value defined in MHP [1], clause O.2.1.2, "getServiceType". If no type is available (e.g. the service concerned has never been received by the GEM terminal) then UNKNOWN shall be returned.

### AO.3.1.4 retrieveDetails

When the service concerned is being received by the GEM terminal, the retrieveDetails method shall retrieve the ServiceDetails as defined in AP.4.1, "javax.tv.service.navigation.ServiceDetails". Otherwise this method shall behave as defined in clause AP.2.2, "Information available only when a stream is actually received".

### AO.3.1.5 hasMultipleInstances

The hasMultipleInstances method shall return true if and only if the service's entry in the broadcast discovery information record includes both IGMP and RTSP protocols.

## AO.3.2 Services only signalled in TS Partial SI Broadcast Discovery Information Record

### AO.3.2.1 General

Implementations of the Java TV APIs shall ignore DVB-SI tables if present in services signalled with TS Partial SI Broadcast Discovery Information records. There is only a single source of SI for such services and that is SD&S.

NOTE 1: The "primary SI source" attribute is ignored.

NOTE 2: Applications wishing to access any DVB-SI information in such services may do this using the org.dvb.si package.

### AO.3.2.2 getLocator

When called on a transport independent service, the getLocator method shall return a transport independent locator for the service as defined in clause 14.9, "Content referencing for IPTV/ OTT".

When called on a transport dependent service, the getLocator method shall return a transport dependent locator for the service as defined in clause 14.9, "Content referencing for IPTV/ OTT".

### AO.3.2.3 getName

When called on a transport independent service, the getName method returns the name of the service as stored in the GEM terminal. Depending on the GEM terminal implementation, the end user may have the possibility to edit these names according to his preferences. In the absence of editing by the end user, one of the multilingual service names as defined in clause AP.2.1, "dvb:MultilingualType" shall be returned.

When called on a transport dependent service, one of the multilingual service names as defined in clause AP.2.1, "dvb:MultilingualType" shall be returned.

### AO.3.2.4 getServiceType

The getServiceType method shall return the service type signalled in the discovery information record mapped into Java as defined in MHP [1], clause O.2.3, "javax.tv.service.ServiceType".

### AO.3.2.5 retrieveDetails

The retrieveDetails method shall return a ServiceDetails instance containing information from the discovery information record as described in clause AP.5.1, "javax.tv.service.navigation.ServiceDetails".

### AO.3.2.6 hasMultipleInstances

The hasMultipleInstances method shall return true if and only if the service's entry in the broadcast discovery information record includes both IGMP and RTSP protocols.

## AO.3.3 Services signalled in more than one mechanism

These services are always transport independent by definition. When a transport independent service of this type is transformed into transport dependent services, those transport dependent services.

### AO.3.3.1 getLocator

If all instances of the service are identified by the same textual service identifier then the method getLocator shall return a transport independent dvb: locator including that textual service identifier otherwise it shall return a transport independent dvb: locator with the DVB triplet.

The following apply to the transport independent locator returned from this method:

- The Locator.hasMultipleTransformations method shall return true.
- The LocatorFactory.transformLocator method shall return one transport dependent locator for each actual service instance.

### AO.3.3.2 getName

The getName method returns the name of the service as stored in the GEM terminal. Depending on the GEM terminal implementation, the end user may have the possibility to edit these names according to his preferences. In the absence of editing by the end user, an implementation specific choice shall be made of one of between the mechanisms in which the service is signalled. The name shall be returned according to the specification for the chosen mechanism - MHP [1], clause O.2.1.1, "getName" or one of the multilingual service names as defined in clause AP.2.1, "dvb:MultilingualType".

### AO.3.3.3 getServiceType

An implementation specific choice shall be made of one of between the mechanisms in which the service is signalled. The type returned shall be according to the specification for the chosen mechanism - MHP [1], clause O.2.1.2, "getServiceType" or clause AO.3.2.4, "getServiceType".

### AO.3.3.4 retrieveDetails

This method shall behave as specified.

### AO.3.3.5 hasMultipleInstances

The hasMultipleInstances method shall return true.

# Annex AP (normative): Mapping between Java TV and service discovery and selection

## AP.1 Introduction (informative)

The following clause defines the mapping between the Java TV SI API and the service discovery and selection. This section applies to services signalled via SD&S. It is divided into 5 clauses:

- generic issues which apply to multiple methods in the API;
- mappings which apply to services regardless of whether they are signalled via TS Full SI Broadcast Discovery Information Record or the TS Partial SI Broadcast Discovery Information Record;
- mappings which only apply to services signalled via TS Full SI Broadcast Discovery Information Record;
- mappings which only apply to services signalled via TS Partial SI Broadcast Discovery Information Record.

## AP.2 Generic issues

### AP.2.1 dvb:MultilingualType

For fields whose value is signalled via a MultilingualType element, if the language returned by javax.tv.service.SIManager.getPreferredLanguage corresponds to a signalled language then the text description in that language shall be used. Otherwise an implementation dependent selection between the available languages shall be made.

### AP.2.2 Information available only when a stream is actually received

The following shall apply for information used in this mapping that is only guaranteed to be available when the GEM terminal is receiving the stream concerned.

- When the service concerned is being received by the GEM terminal, the information shall be returned from the received service.
- When the service concerned is not being received by the GEM terminal, the information may be returned if the GEM terminal has it in cache from when the service was previously received.
- If the service is not being received and the information is not cached, the request shall fail as defined for the method concerned when data is unavailable. (This means an SIRequestFailureType of DATA_UNAVAILABLE for methods passed an SIRequestor). In this case, the GEM terminal shall not request the network to supply the service concerned.

# AP.3 Mappings independent of Discovery Information Record Type

## AP.3.1 javax.tv.service.navigation.ServiceProviderInformation

The name of the service provider shall be retrieved from the multilingual service provider name attribute of the service provider discovery record for the service provider. The name returned shall be selected from those available as specified by clause AP.2.1, "dvb:MultilingualType".

## AP.3.2 javax.tv.service.navigation.StreamType

When the service concerned is being received by the GEM terminal, the stream type shall be as defined in MHP [1], clause O.2.4. When the service concerned is not being received by the GEM terminal but information on the stream types of the components of that service are cached in the GEM terminal then the cached data shall be returned. Otherwise StreamType.UNKNOWN shall be returned.

## AP.3.3 javax.tv.service.navigation.ServiceComponent

This shall be supported as defined in MHP [1], clause O.2.2, "javax.tv.service.navigation.ServiceComponent".

## AP.3.4 javax.tv.service.transport.Network

This is not required to be implemented but if implemented, shall be supported as defined in MHP [1], clause O.2.14, "javax.tv.service.transport.Network".

NOTE: This is not required to be implemented because NetworkCollection is not required to be implemented.

## AP.3.5 javax.tv.service.transport.Transport

### AP.3.5.1 getDeliverySystemType

The getDeliverySystemType method shall return the delivery system type defined in org.dvb.service.navigation.IPDeliverySystemType.

## AP.3.6 javax.tv.service.transport.Bouquet

Each package in the Package Discovery Information record shall be represented as an instance of Bouquet.

### AP.3.6.1 getBouquetID

The getBouquetID method shall return the package id from the Package Discovery Information record.

### AP.3.6.2 getName

The getName method shall return one of the multilingual package names selected as defined in clause AP.2.1, "dvb:MultilingualType".

## AP.3.7 javax.tv.service.transport.BouquetCollection

There is no requirement to monitor for changes more frequently than once per day. When changes are detected, BouquetChangeEvents will be generated when packages are added or removed. BouquetChangeEvents with SIChangeType MODIFY shall only be generated when it is detected that the contents of the service list for a package changes.

## AP.3.8 javax.tv.service.transport.TransportStream

This shall be supported as defined in MHP [1], clause O.2.15, "javax.tv.service.transport.TransportStream" except that the transport stream ID returned shall always be taken from the broadcast discovery information record.

## AP.3.9 javax.tv.service.transport.TransportStreamCollection

This is not required to be implemented. If it is implemented, there is no requirement to monitor for changes more frequently than once per day.

## AP.3.10 javax.tv.service.transport.NetworkCollection

This is not required to be implemented. If it is implemented, there is no requirement to monitor for changes more frequently than once per day.

NOTE: Where broadcast services are being relayed for carriage over an IP network, it is unclear whether any re-writing of DVB-SI network_ids will be done. Hence the network_ids may be more varied than would be the case in a classical DVB environment.

## AP.3.11 javax.tv.service.ServiceInformationType

The present document defines additional service information types in org.dvb.service.IPServiceInformationType.

## AP.3.12 javax.tv.service.ServiceType

The translation from DVB-SI encoding shall be as defined in MHP [1], clause O.2.3, "javax.tv.service.ServiceType".

## AP.3.13 javax.tv.service.navigation.DeliverySystemType

The delivery system type returned shall be org.dvb.service.navigation.IPDeliverySystemType.IP as defined in the present document.

## AP.3.14 ServiceNumber

Support for ServiceNumber is not defined in GEM.

## AP.3.15 javax.tv.service.navigation.CAIdentification

Support for CAIdentification is not defined in GEM.

## AP.3.16 javax.tv.service.navigation.FavoriteServicesName

This is not a signalling issue but is an attribute of the navigator UI.

### AP.3.17 javax.tv.service.transport.Network

This is not required to be implemented but if implemented, shall be supported as defined in MHP [1], clause O.2.14, "javax.tv.service.transport.Network".

## AP.4 Mappings specific to TS Full SI Broadcast Discovery Information Record

### AP.4.1 javax.tv.service.navigation.ServiceDetails

These shall be supported as defined in MHP [1], clause O.2.8, "javax.tv.service.navigation.ServiceDetails".

When the service concerned is being received by the GEM terminal, the GEM terminal shall monitor the SI of the service and post events to ServiceComponentChangeListeners as specified. When it is not being received, no SI monitoring is possible and hence no events will be posted.

Information on the components of a service shall be supported as defined by clause AP.2.2, "Information available only when a stream is actually received" Information available only when a stream is actually received.

ServiceDetails objects shall implement the ServiceProviderInformation interface.

The Locator returned by getLocator shall be a transport dependent locator for this specific service.

## AP.5 Mappings specific to TS Partial SI Broadcast Discovery Information Record

### AP.5.1 javax.tv.service.navigation.ServiceDetails

ServiceDetails objects shall implement the ServiceProviderInformation interface.

#### AP.5.1.1 getDeliverySystemType

The delivery system type returned shall be org.dvb.service.navigation.IPDeliverySystemType.IP as defined in the present document.

#### AP.5.1.2 getLongName

The getLongName method shall return one of the multilingual service names as defined in clause AP.2.1, "dvb:MultilingualType".

#### AP.5.1.3 getService

The getService method shall return the appropriate service object for this service details.

#### AP.5.1.4 getLocator

The getLocator method shall return a transport dependent locator for this specific service as defined in clause 14.9, "Content referencing for IPTV/ OTT".

### AP.5.1.5 getServiceType

The getServiceType method shall return the service type signalled in the discovery information record decoded according to MHP [1], clause O.2.3, "javax.tv.service.ServiceType".

### AP.5.1.6 retrieveComponents

Information on the service components shall be returned based on the signalling in the PMT of the transport stream carrying the service. Clause AP.2.2, "Information available only when a stream is actually received" Information available only when a stream is actually received shall apply.

### AP.5.1.7 retrieveServiceDescription

The retrieveServiceDescription method shall return one of the optional multilingual service descriptions as defined by clause AP.2.1, "dvb:MultilingualType" or fail with an SIRequestFailureType of DATA_UNAVAILABLE if none of those descriptions are present. Clause AP.2.2, "Information available only when a stream is actually received". Information available only when a stream is actually received shall apply.

### AP.5.1.8 addServiceComponentChangeListener

If the service concerned is currently being received by the GEM terminal then the PMT of the transport stream carrying the service shall be monitored for changes to service components. Otherwise monitoring the SI for changes to service components is impossible.

## AP.5.2 javax.tv.service.navigation.ServiceDescription

The service description shall be one of the descriptions in the Description element of the SI element. The choice of which description to return shall be as defined by clause AP.2.1, "dvb:MultilingualType".

## AP.5.3 javax.tv.service.navigation.ServiceComponent

### AP.5.3.1 getName

The getName method shall return an empty string

### AP.5.3.2 getAssociatedLanguage

The getAssociatedLanguage method shall return an empty string

### AP.5.3.3 getStreamType

The getStreamType method shall return a StreamType as defined in clause AP.3.2, "javax.tv.service.navigation.StreamType".

### AP.5.3.4 getService

The getService method shall return the appropriate service.

# Annex AQ (normative): Mapping between Java TV and broadband content guide

The following clauses define the mapping between the Java TV SI API and the TV-Anytime metadata specified by the broadband content guide - TS 102 539 [81].

## AQ.1 Finding the BCG Provider

The Java TV API shall use the SD&S Service(s)DescriptionLocation field to locate the BCG provider to use. The SD&S information may specify a BCG provider for a range of services or one per service.

If present in the SD&S information the "preferred" attribute shall be used to identify which BCG provider takes precedence. If not present then GEM does not define which of the signalled providers to use.

## AQ.2 javax.tv.service.Service and javax.tv.service.navigation.ServiceDetails

In GEM terminals supporting the broadband content guide, clause 6.6 of TS 102 539 [81] shall apply.

NOTE: This means information from the BCG takes precedence over information from SD&S. Information from SD&S is only used when fields which are optional in BCG are not present.

## AQ.3 javax.tv.service.guide.ProgramSchedule

This maps to a collection of <BroadcastEvent> and <ScheduleEvent> elements in the <ProgramLocationTable> for a particular service. <ScheduleEvent> elements are contained within one to many <Schedule> elements.

### AQ.3.1 retrieveCurrentProgramEvent

The retrieveCurrentProgramEvent method shall return a ProgramEvent based on the instance of <ScheduleEvent> or <BroadcastEvent> that has <StartTime> and <EndTime> overlapping the current time.

### AQ.3.2 retrieveFutureProgramEvents

The retrieveFutureProgramEvents method shall return all ProgramEvents based on instances of <ScheduleEvent> or <BroadcastEvent> that have <StartTime> and <EndTime> between the specified begin and end time.

### AQ.3.3 retrieveNextProgramEvent

The retrieveNextProgramEvent method shall return a ProgramEvent based on an instance of <ScheduleEvent> or <BroadcastEvent> that follows the current event in the schedule in terms of <StartTime>.

### AQ.3.4 retrieveProgramEvent

The retrieveProgramEvent method shall return the ProgramEvent that matches the Locator as defined by clause 11.7.11, "TV-Anytime content referencing and metadata".

### AQ.3.5 addListener

No additional semantics are defined in the present document.

NOTE 1: TV-Anytime bi-directional updates are based on polling.

NOTE 2: <ScheduleEvent> elements can only be updated in TV-Anytime as part of a whole <Schedule> element, whereas <BroadcastEvent> elements can be updated independently.

### AQ.3.6 removeListener

No additional semantics are defined in the present document.

### AQ.3.7 getServiceLocator

The getServiceLocator method shall return a Locator based on the TV-Anytime <ServiceURL> obtained from the Service Information Table entry which matches the serviceId of the <Schedule> or <BroadcastEvent> element.

## AQ.4 javax.tv.service.guide.ProgramEvent

ProgramEvent shall map either to a TV-Anytime <ScheduleEvent> or to a <BroadcastEvent> element within the <ProgramLocationTable>.

### AQ.4.1 getLocator

The getLocator method shall return a Locator for the ProgramEvent obtained by resolving the CRID of the <ScheduleEvent> or <BroadcastEvent>. This locator shall comply with clause 11.7.11, "TV-Anytime content referencing and metadata".

### AQ.4.2 getStartTime

The getStartTime method shall return the time specified in the <PublishedStartTime> of a <ScheduleEvent> or <BroadcastEvent> in the <ProgramLocationTable>.

### AQ.4.3 getEndTime

The getEndTime method shall return the time specified in the <PublishedEndTime> of a <ScheduleEvent> or <BroadcastEvent> element in the <ProgramLocationTable>. If not present then the <PublishedStartTime> and <PublishedDuration> shall be used to calculate this.

### AQ.4.4 getDuration

The getDuration method shall return the time specified in the <PublishedDuration> of a <ScheduleEvent> or <BroadcastEvent> element in the <ProgramLocationTable>.

If <PublishedDuration> is not present then the <PublishedStartTime> and <PublishedEndTime> shall be used to calculate this.

### AQ.4.5 getName

The getName method shall return the String specified in the <Title> element of <InstanceDescription>, contained in either a <ScheduleEvent> or <BroadcastEvent> element in the <ProgramLocationTable>.

If <InstanceDescription> is not present then the <Title> element from the <BasicContentDescription> element in the <ProgramInformationTable> shall be returned

### AQ.4.6 retrieveDescription

The retrieveDescription method shall return the String specified in the short <Synopsis> element of <InstanceDescription>, contained in either a <ScheduleEvent> or <BroadcastEvent> element in the <ProgramLocationTable>.

If <InstanceDescription> is not present then the short <Synopsis> element from the <BasicContentDescription> element in the <ProgramInformationTable> shall be used.

If the short <Synopsis> is not present then the long <Synopsis> shall be used.

### AQ.4.7 getRating

The getRating method shall return a ContentRatingAdvisory based on the TV-Anytime <ParentalGuidance> element from the <ContentDescription> element contained in the <ProgramInformationTable>.

The ContenRatingAdvisory shall only represent ratings that are part of the rating dimensions returned by the method SIManager.getSupportedDimensions.

If there is no TV-Anytime rating information present within the supported dimensions then this method shall return null.

### AQ.4.8 getService

No additional semantics are defined in GEM.

### AQ.4.9 retrieveComponents

No mapping is defined in GEM.

## AQ.5 javax.tv.service.guide.ProgramEventDescription

The ProgramEventDescription shall be constructed from the short <Synopsis> element of <InstanceDescription>, contained in either a <ScheduleEvent> or <BroadcastEvent> element in the <ProgramLocationTable>.

If <InstanceDescription> is not present then the short <Synopsis> element from the <BasicContentDescription> element in the <ProgramInformationTable> shall be used.

If the short <Synopsis> is not present then the long <Synopsis> shall be used.

# Annex AR (normative): XML encoding for AIT

Signaling of external applications is optional for IPTV targets.

## AR.1 Introduction

This annex defines an XML encoding for the AIT in addition to the already existing MPEG-2 table and section based encoding defined in MHP [1], clause 10, "Application Signalling". Since the first intended use for this XML encoding is in conjunction with the SD&S defined in TS 102 034 [80], this encoding follows the same format and re-uses already defined elements and types.

The following features of the MPEG-2 encoding are not supported in the XML encoding:

- Application types different from DVB-J and DVB-HTML are not currently taken into account.
- DVB-HTML specific descriptors are not defined in GEM.
- Signalling of applications transported in DSM-CC object carousel is not supported.

The semantics of the fields defined in this annex shall be identical to those of the corresponding fields in the existing MPEG-2 table and section based encoding as defined by GEM and extended by OCAP [5].

## AR.2 Extensions to defined SD&S elements

### AR.2.1 Package

Add to Package, an optional list of applications and an optional list of externally authorised applications:

```
<xsd:element name="applicationList" type="dvb:mhp:ApplicationList" minOccurs="0"/>

<xsd:element name="ExternalapplicationAuthorisedList"
type="dvb:mhp:ExternalApplicationIdentifier**" minOccurs="0"/>

<xsd:complexType name="PackagedServices">
    <xsd:complexContent>
        <xsd:extension base="dvb:OfferingBase">
            <xsd:sequence>
                <xsd:element name="Package" maxOccurs="unbounded">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element name="PackageName" type="dvb:MultilingualType"
                            maxOccurs="unbounded"/>
                            <xsd:element name="IPService" maxOccurs="unbounded">
                                <xsd:complexType>
                                    <xsd:sequence>
                                        <xsd:element name="TextualID" type="dvb:TextualIdentifier"/>
                                        <xsd:element name="DescriptionLocation"
type="dvb:DescriptionLocation"
                                        minOccurs="0"/>
                                    </xsd:sequence>
                                </xsd:complexType>
                            </xsd:element>
                            <xsd:element name="CountryAvailability" type="dvb:CountryAvailability"
                            minOccurs="0" maxOccurs="unbounded"/>
                            <xsd:element name="PackageDescription" type="dvb:DescriptionLocation"
                            minOccurs="0" maxOccurs="unbounded"/>
                            <xsd:element name="applicationList" type="dvb:mhp:ApplicationList"
                            minOccurs="0"/>
                            <xsd:element name="ExternalapplicationAuthorisedList"
                            type="dvb:mhp:ExternalApplicationIdentifier" minOccurs="0"/>
                        </xsd:sequence>
```

```
                        <xsd:attribute name="Id" type="dvb:Hexadecimal16bit" use="required"/>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
```

The application list signalled in the package is aimed to contain the whole set of applications which are available for all the IPServices signalled in such package.

## AR.2.2 IP Service

Add the following elements, as optional:

```
<xsd:element name="applicationList" type="dvbmhp:ApplicationList" minOccurs="0"/>
<xsd:element name="ExternalapplicationAuthorisedList"
type="dvb:ExternalApplicationIdentifier" minOccurs="0"/>
```

Therefore the IPService definition is modified as follows:

```
<xsd:complexType name="IPService">
    <xsd:sequence>
        <xsd:element name="ServiceLocation" type="dvb:ServiceLocation"/>
        <xsd:element name="TextualIdentifier" type="dvb:TextualIdentifier"/>
        <xsd:element name="DVBTriplet" type="dvb:DVBTriplet"/>
        <xsd:element name="MaxBitrate" type="xsd:positiveInteger" minOccurs="0"/>
        <xsd:element name="SI" type="dvb:SI" minOccurs="0"/>
        <xsd:element name="applicationList" type="dvb:MHPApplicationList" minOccurs="0"/>
        <xsd:element name="ExternalapplicationAuthorisedList"
        type="dvb:ExternalApplicationIdentifier" minOccurs="0"/>
        <xsd:element name="VideoCoding" type="dvb:VideoCoding" minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="AudioCoding" type="dvb:AudioCoding" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
```

# AR.3 New XML element definitions

## AR.3.1 ApplicationList

```
<xsd:complexType name="ApplicationList">
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
        <xsd:element name="application" type="dvb:mhp:Application"/>
    </xsd:sequence>
</xsd:complexType>
```

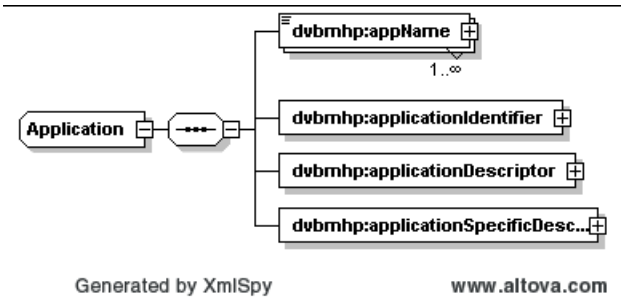
## AR.3.2 Application

```
<xsd:complexType name="Application">
        <xsd:sequence>
            <xsd:element name="appName" type="dvb:MultilingualType" maxOccurs="unbounded"/>
            <xsd:element name="applicationIdentifier" type="dvb:mhp:ApplicationIdentifier"/>
            <xsd:element name="applicationDescriptor" type="dvb:mhp:ApplicationDescriptor"/>
            <xsd:element name="applicationSpecificDescriptor"
            type="dvb:mhp:ApplicationSpecificDescriptor"/>
            <xsd:element name="providerExportDescriptor"
            type="dvb:mhp:exportDescriptor"/>
            <xsd:element name="providerUsageDescriptor"
            type="dvb:mhp:usageDescriptor"/>
        </xsd:sequence>
</xsd:complexType>
```

A GEM application can be completely described by:

- An Application Name which can be multilingual. (appName).
- An Unique identification (applicationIdentifier).

- A generic descriptor which is common and mandatory for both types of applications (DVB-J & DVB-HTML) defined in the present document. This generic descriptor is valid also for other type of applications, e.g OCAP.
- An Application Specific descriptor which will depend upon the signalled application, e.g. DVB-J, DVB-HTML.

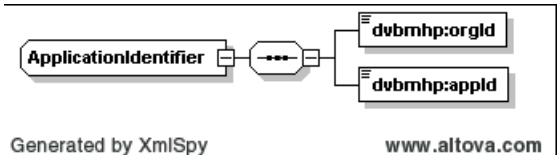



# AR.3.3 ApplicationIdentifier

```
<xsd:complexType name="ApplicationIdentifier">
    <xsd:sequence>
        <xsd:element name="orgId" type="xsd:unsignedInt"/>
        <xsd:element name="appId" type="xsd:unsignedShort"/>
    </xsd:sequence>
</xsd:complexType>
```

As defined in clause 10.4.3, "Content of the Application Description" of the present document, an application is uniquely identified by:

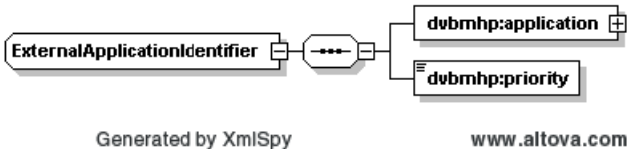
- OrgId, organisation identifier, it is a globally unique value identifying the organisation that is responsible for the application.
- AppId, application identifier, it is allocated by the organisation registered with the organisation identifier who decides the policy for allocation within the organisation.




# AR.3.4 ExternalApplicationIdentifier

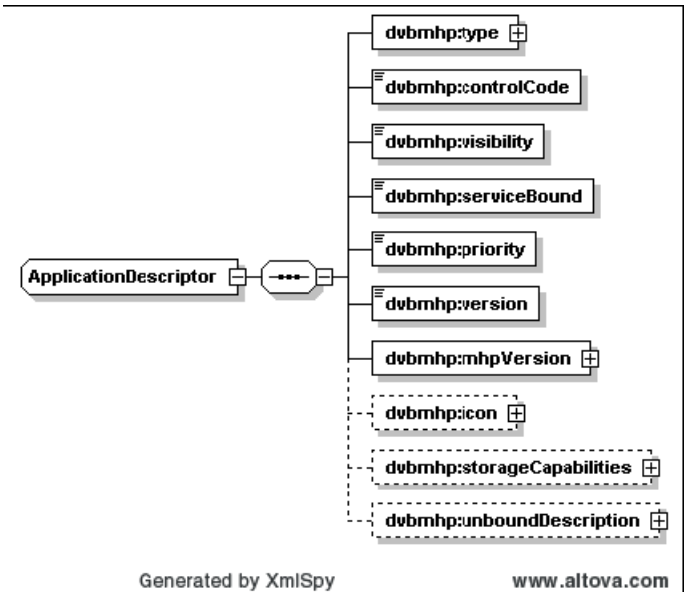
```
<xsd:complexType name="ExternalApplicationIdentifier">
    <xsd:sequence>
        <xsd:element name="application" type="dvb:mhp:ApplicationIdentifier"/>
        <xsd:element name="priority" type="dvb:Hexadecimal8bit"/>
    </xsd:sequence>
</xsd:complexType>
```

As defined in clause 10.7.5, "External application authorization descriptor" of MHP [1], an external application Identifier is used to signal an application which is allowed to continue running in the current service even if it is not signalled in it.



# AR.3.5 ApplicationDescriptor

```
<xsd:complexType name="ApplicationDescriptor">
    <xsd:sequence>
        <xsd:element name="type" type="dvb:mhp:ApplicationType"/>
        <xsd:element name="controlCode" type="dvb:mhp:ApplicationControlCode"/>
        <xsd:element name="visibility" type="dvb:mhp:VisibilityDescriptor"/>
        <xsd:element name="serviceBound" type="xsd:boolean" default="true"/>
        <xsd:element name="priority" type="dvb:Hexadecimal8bit"/>
        <xsd:element name="version" type="dvb:Version"/>
        <xsd:element name="mhpVersion" type="dvb:mhp:MhpVersion"/>
        <xsd:element name="icon" type="dvb:mhp:IconDescriptor" minOccurs="0"/>
        <xsd:element name="storageCapabilities" type="dvb:mhp:StorageCapabilities" minOccurs="0"/>
        <xsd:element name="unboundDescription" type="dvb:mhp:UnboundApplicationDescriptor"
        minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>
```




This contents of this complex type are mostly those defined in clause 10.7.3, "Application descriptor" of MHP [1].

The simple type elements are defined as follows:

visibility: this element specifies whether the application is suitable to be offered to the end-user for them to decide if the application should be launched. Therefore "false" value means that application shall not be visible either to applications via an application listing API or to users via the navigator with the exception of any error reporting or logging facility, etc., whereas a "true" value means that application shall not be visible to users but shall be visible to applications via an application listing API.

serviceBound: whether the application is bound to a service or not as defined by the `service_bound_flag`.

priority: This field identifies a relative priority between the applications signalled in this service.

- Where there is more than one application with the same Application identification this priority shall be used to determine which application is started.
- Where there are insufficient resources to continue running a set of applications, this priority shall be used to determine which applications to stop or pause.
- A larger integer value indicates higher priority.
- If two applications have the same application identification and the same priority, the GEM may make an implementation-dependent choice on which to start.

version: Version of the application which starts at zero and increments by one each time any of the files listed in the "Application description file" change or the contents of the "Application description file" itself changes. Used values shall never reused, in the event that the number range is exhausted a new application id shall be used.

mhpVersion: version of the GEM as defined in 10.7.3 "Application descriptor" of MHP [1].

icon: to bind an icon to the application.

storageCapabilities: this optional element shall be added for giving the receiver the required information to store / cache the application.

unboundDescription: this descriptor only shall be used when serviceBound is set to false, i.e. it is an unbound application.

# AR.3.6 VisibilityDescriptor

```
<xsd:simpleType name="VisibilityDescriptor">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="NOT_VISIBLE_ALL"/>
        <xsd:enumeration value="NOT_VISIBLE_USERS "/>
        <xsd:enumeration value="VISIBLE_ALL"/>
    </xsd:restriction>
</xsd:simpleType>
```

The following correspond to the values of the field defined in MHP [1], table 84, "Definition of visibility states for applications".

NOT_VISIBLE_ALWAYS: This application shall not be visible either to applications via an application listing API or to users via the navigator with the exception of any error reporting or logging facility, etc.

NOT_VISIBLE_USERS: This application shall not be visible to users but shall be visible to applications via an application listing API.

VISIBLE_ALL: This application can be visible to users and shall be visible to applications via the application listing API.

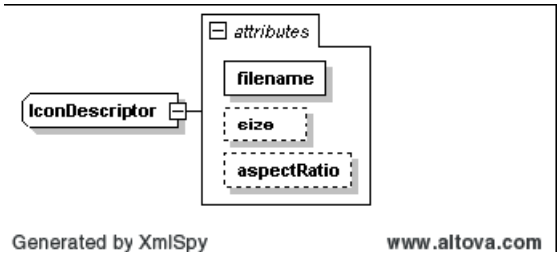
# AR.3.7 IconDescriptor

```
<xsd:complexType name="IconDescriptor">
    <xsd:attribute name="filename" type="xsd:string" use="required"/>
    <xsd:attribute name="size" type="xsd:unsignedShort" use="optional"/>
    <xsd:attribute name="aspectRatio" type="dvb:AspectRatio" use="optional"/>
</xsd:complexType>
```

As defined in clause 10.7.4.2, "Application icons descriptor" of MHP [1] the icon Descriptor serves to signal the presence of an icon representing the application. Size and aspectRatio attributes are defined as optional since they can be determined as defined in MHP [1], table 88, "Definition of different icon flags".

E.g.

```
<icon filename="dvb.icon.1"/> , size = 32x32 pixel square
```




# AR.3.8 AspectRatio

```
<xsd:simpleType name="AspectRatio">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="4_3"/>
        <xsd:enumeration value="16_9 "/>
    </xsd:restriction>
```

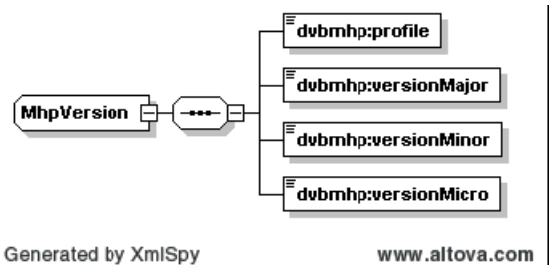
```
</xsd:simpleType>
```

These aspect ratios are the set of aspect ratios used in MHP [1], table 88, "Definition of different icon flags".

## AR.3.9 MhpVersion

```
<xsd:complexType name="MhpVersion">
    <xsd:sequence minOccurs="1">
        <xsd:element name="profile" type=" dvb:Hexadecimal16bit "/>
        <xsd:element name="versionMajor" type=" dvb:Hexadecimal8bit "/>
        <xsd:element name="versionMinor" type=" dvb:Hexadecimal8bit "/>
        <xsd:element name="versionMicro" type=" dvb:Hexadecimal8bit "/>
    </xsd:sequence>
</xsd:simpleType>
```

See MHP [1] clause 10.7.3, "Application descriptor".




## AR.3.10 StorageCapabilities

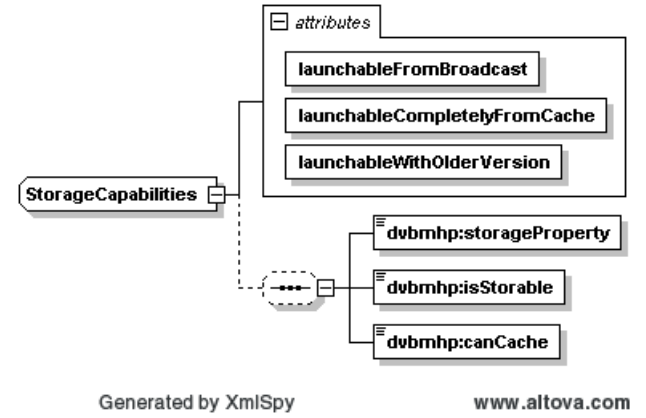
```
<xsd:complexType name="StorageCapabilities">
    <xsd:sequence minOccurs="0">
        <xsd:element name="storageProperty" type="dvb:StorageType"/>
        <xsd:element name="isStorable" type="xsd:boolean"/>
        <xsd:element name="canCache" type="xsd:boolean"/>
    </xsd:sequence>
    <xsd:attribute name="launchableFromBroadcast" type="xsd:boolean" use="required"/>
    <xsd:attribute name="launchableCompletelyFromCache" type="xsd:boolean" use="required"/>
    <xsd:attribute name="launchableWithOlderVersion" type="xsd:boolean" use="required"/>
</xsd:complexType>
```

This descriptor, if present, serves to state whether the application can be stored or cached in the receiver as defined in MHP [1], clause 10.8.2, "Application storage descriptor".

isStorable: if set to true, it means that the application can be stored in the receiver.

canCache: if set to true, it means that this application can be run entirely from cache.

The attributes launchableFromBroadcast, launchableCompletelyFromCache, launchableWithOlderVersion have exactly the same meaning as the bit flags as defined in MHP [1], clause 10.8.2, "Application storage descriptor".




# AR.3.11 StorageType

```
<xsd:simpleType name="StorageType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="BROADCAST-RELATED"/>
        <xsd:enumeration value="STANDALONE"/>
    </xsd:restriction>
</xsd:simpleType>
```

See MHP [1], table 113, "Semantics of storage property values".

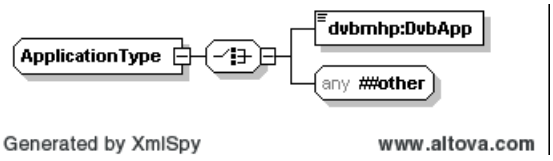
If the storage type is "BROADCAST-RELATED" then the application's life cycle is controlled by the broadcast service. Such applications are suitable for caching but not for running as a stand alone application.

If the storage type is "STANDALONE" then the application can usefully be launched by the user independently of a broadcast service.

# AR.3.12 ApplicationType

```
<xsd:complexType name="ApplicationType">
        <xsd:choice>
            <xsd:element name="DvbApp" type="dvb:mhp:DvbApplicationType"/>
            <xsd:any namespace="##other" processContents="lax"/>
        </xsd:choice>
    </xsd:complexType>
```

See MHP [1], table 75, "Application types".




# AR.3.13 DvbApplicationType

```
<xsd:simpleType name="DvbApplicationType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="DVB-J"/>
        <xsd:enumeration value="DVB-HTML"/>
    </xsd:restriction>
</xsd:simpleType>
```

This descriptor is a choice between the both application types as defined in MHP [1], table 75, "Application types".

## AR.3.14 ApplicationControlCode

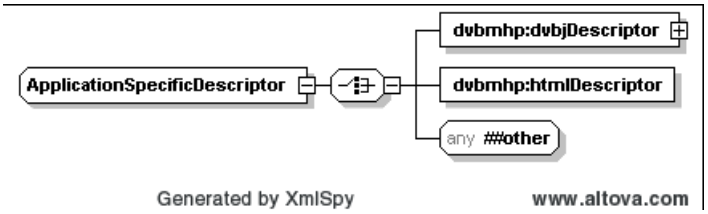
```
<xsd:simpleType name="ApplicationControlCode">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="AUTOSTART"/>
        <xsd:enumeration value="PRESENT"/>
        <xsd:enumeration value="DESTROY"/>
        <xsd:enumeration value="KILL"/>
        <xsd:enumeration value="REMOTE"/>
    </xsd:restriction>
</xsd:simpleType>
```

This descriptor serves to dynamically control application life cycle. The meaning of each one of the enumeration elements as well as the expected behaviour in the GEM receiver is fully defined in MHP [1], clause 10.7.3, "Application descriptor".

## AR.3.15 ApplicationSpecificDescriptor

```
<xsd:complexType name="ApplicationSpecificDescriptor">
    <xsd:choice>
        <xsd:element name="dvbjDescriptor" type="dvb:mhp:DVBJDescriptor"/>
        <xsd:element name="htmlDescriptor" type="dvb:mhp:DVBHtmlDescriptor"/>
        <xsd:any namespace="##other" processContents="lax"/>
    </xsd:choice>
</xsd:complexType>
```

This descriptor contains the specific descriptor depending upon the type of application. As well as descriptors defined in the present document, it may also include descriptors from outside, e.g. a descriptor for OCAP applications.




## AR.3.16 DVBJDescriptor

```
<xsd:complexType name="DVBJDescriptor">
    <xsd:sequence>
        <xsd:element name="location" type="dvb:DescriptionLocation"
        minOccurs="1" maxOccurs="unbounded"/>
        <xsd:element name="applicationStructure" type="dvb:mhp:ApplicationStructure"/>
    </xsd:sequence>
</xsd:complexType>
```

For executing DVB-J application, the receiver needs specifically where it is placed -its location- and how it is structured -in order to "rebuild" it-. The contents of this element correspond to the contents of MHP [1], table 97, "Syntax of selector bytes for interaction transport".

At least one location element must be provided in order to allow the receiver to find the ADF as well as the elements pointed by the ADF.

The location elements are of type dvb:DescriptionLocation which are defined in TS 102 034 [80] as URI. A set of URIs should be enough to find the elements. See example below:

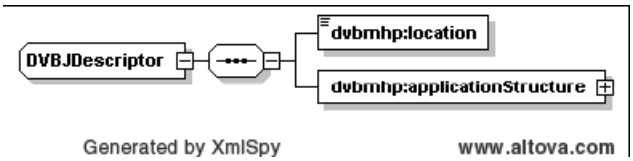
e.g.

```
< DVBJDescriptor>
    < location>http://www.dvb.org/applications/application1.zip</location>
    < location>http://www.dvb.org/graphics</location>
    < location>http://www.dvb.org/shared/utils.zip</location>
    < location>http://www.ebu.ch/general/misc.zip</location>
    < location>udp://www.dvb.org/byudp </location>
    <applicationStructure>
            <!—Not necessary for this example-- >
```

```
<   / applicationStructure>
</ DVBJDescriptor >
```

Then the search order of for the "dvb.fontindex" file will be the following:

- In the contents of http://www.dvb.org/applications/application1.zip.
- From the URL http://www.dvb.org/graphics.
- In the contents of http://www.dvb.org/shared/utils.zip.
- In the contents of http://www.dvb.org/shared/misc.zip.
- In the URL udp://www.dvb.org/byudp.

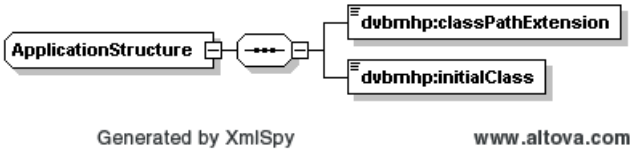



# AR.3.17 ApplicationStructure

```
<xsd:complexType name="ApplicationStructure">
    <xsd:sequence>
        <xsd:element name="classPathExtension" type="xsd:string"/>
        <xsd:element name="initialClass" type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
```

This complex type is an XML encoding of some of the information described in MHP [1], clause 10.9.2, "DVB-J application location descriptor".

In order to recreate the application, and execute it, only is required the set of files (class files, images, etc.) -ADF-, the classPathExtension, which can be "", and the initial class (i.e. the class which implements Xlet interface).

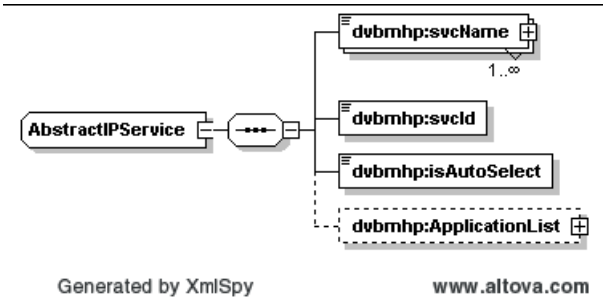



# AR.3.18 AbstractIPService

Most of the information in this complex type is the same as that carried in the abstract service descriptor defined in OCAP [5].

```
<xsd:complexType name="AbstractIPService">
    <xsd:sequence>
        <xsd:element name="svcName" type="dvb:MultilingualType" maxOccurs="unbounded"/>
        <xsd:element name="svcId" type="dvb:mhp:Hexadecimal24bit"/>
        <xsd:element name="isAutoSelect" type="xsd:boolean"/>
        <xsd:element name="ApplicationList" type="dvb:mhp:ApplicationList" minOccurs="0"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:simpleType name="Hexadecimal24bit">
    <xsd:restriction base="xsd:string">
        <xsd:pattern value="[0-9a-fA-F]{1}[1-9a-fA-F]{1}[0-9a-fA-F]{4}"/>
    </xsd:restriction>
</xsd:simpleType>
```

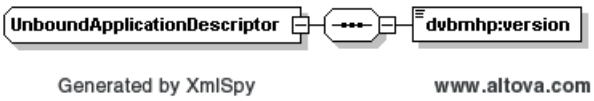



## AR.3.19 UnboundApplicationDescriptor

Most of the information carried in this complex type is the same as that carried in the unbound application descriptor defined in OCAP [5].

```
<!-- To signal an unbound application -->
    <xsd:complexType name="UnboundApplicationDescriptor">
        <xsd:sequence>
            <xsd:element name="version" type="dvb:mhp:Hexadecimal32bit"/>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:simpleType name="Hexadecimal32bit">
        <xsd:restriction base="xsd:string">
            <xsd:pattern value="[0-9a-fA-F]{1,8}"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:schema>
```



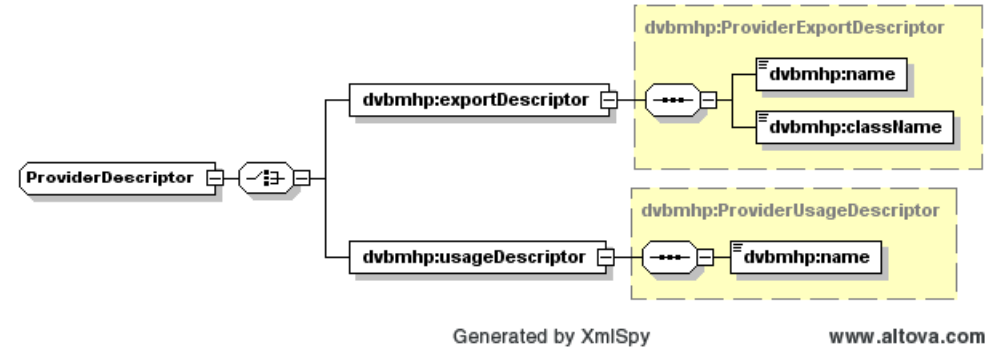

## AR.3.20 Provider descriptors

The information carried in these complex types is the same as that carried in the provider_export_descriptor and provider_usage_descriptor defined in MHP [1], clause 10.15, "Signalling for providers".

```
<xsd:complexType name="ProviderDescriptor">
    <xsd:choice>
        <xsd:element name="exportDescriptor" type="dvbmhp:ProviderExportDescriptor"/>
        <xsd:element name="usageDescriptor" type="dvbmhp:ProviderUsageDescriptor"/>
    </xsd:choice>
</xsd:complexType>

<xsd:complexType name="ProviderExportDescriptor">
    <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
        <xsd:element name="className" type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="ProviderUsageDescriptor">
    <xsd:sequence>
        <xsd:element name="name" type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
```


dvbmhp:ProviderExportDescriptor
dvbmhp:name
dvbmhp:className
ProviderDeecriptor
dvbmhp:exportDescriptor
dvbmhp:ProviderUsageDescriptor
dvbmhp:usageDescriptor
dvbmhp:name
Generated by XmlSpy
www.altova.com

# Annex AS (informative): IPTV Use-cases

## AS.1 Simple use-case

Table 90 breaks a simple use-case down into a series of steps. For each step, the means by which GEM supports that step is described and one or more references to the appropriate clauses provided.

This use case is for a GEM terminal supporting the privileged application option.

**Table 90: Simple Use-case**

| Step | How supported | References |
|---|---|---|
| Box is turned on and boots operator application | Signalling identifies operator applications. Loaded and started by existing carousel or HTTP mechanisms. | MHP [1], clause 10.16.2, "XAIT" |
| Application asks for list of IPTV services via SD&S | SIManager.filterServices(DeliverySystemTypeFilter) ServiceDetails.getName | 11.6.5.4, "Extensions" |
| Application presents list of services to user and asks to pick one | Existing GEM APIs | - |
| Application shows a preview of the content available on that service | Manager.createPlayer( new MediaLocator("rtsp:...") ) | 11.11.4.4, "Content referencing for IPTV" |
| Confirm purchase | Via https connection to web server using existing java.net | 11.6.4, "Conditional access API" |
| Selects service | Service selection API | 11.11.4.4, "Content referencing for IPTV" |
| Application removes its UI | Existing GEM (org.havi.ui or java.awt) | - |
| Content presented in TV mode | Existing GEM | - |
| Any service bound applications are now presented | Existing GEM | - |
| User uses pause / continue / fast-forwards / fast-rewind etc. | Either Player.setRate or TCP connection to server using java.net | 11.3.1.5, "java.net", 11.4.2.10, "Additional and modified semantics for IPTV" |
| In parallel, operator application(s) listens for modal (menu/home) keys | org.dvb.event with new VK_ codes defined in the present document | Table 86 "Minimum set of input events" |
| Operator application displays its UI or changes channel | Existing GEM (java.awt or service selection API) | - |
| Operator application stops content being presented | Existing GEM service selection API | - |

# Annex AT (normative): Application Management API

# Package org.dvb.application.privileged

Description

Provides application management facilities to privileged applications.

| Class Summary | |
|---|---|
| **Interfaces** | |
| ApplicationStorageHandler | Enables privileged applications to make decisions for some methods in the org.dvb.application.storage package. |
| **Classes** | |
| ApplicationStorageManager | Provides application management features and capabilities only appropriate for privileged applications. |

# org.dvb.application.privileged ApplicationStorageHandler

Declaration

```
public interface ApplicationStorageHandler
```

Description

Enables privileged applications to make decisions for some methods in the org.dvb.application.storage package. The methods in this interface correspond to the methods in ApplicationStorageController that throw a UserRejectedInstall-Exception. If an application storage handler does not permit a request then it should return false and the original method shall fail with a UserRejectedInstallException.

Implementations of the methods in this interface should complete quickly and not perform time-consuming operations such as blocking I/O calls.

Methods

removeAppRequested(AppAttributes)

```
public boolean removeAppRequested(org.dvb.application.AppAttributes app)
```

Called as part of the execution of ApplicationStorageController.remove(AppID).

Parameters:

app - the application to be removed.

Returns:

true if the remove request is permitted, otherwise false.

removeAppRequested(AppAttributes[])

```
public boolean removeAppRequested(org.dvb.application.AppAttributes[] app)
```

Called as part of the execution of ApplicationStorageController.remove(AppID[]).

Parameters:

app - the applications to be removed.

Returns:

true if the remove request is permitted, otherwise false.

### removeServiceRequested(ApplicationStorageController)

```
public boolean removeServiceRequested(org.dvb.application.storage.ApplicationStorageController
service)
```

Called as part of the execution of ApplicationStorageController.removeService.

Parameters:

service - the service for which removeService was called.

Returns:

true if the remove request is permitted, otherwise false.

### storeRequested(ApplicationStorageController, AppAttributes[], boolean[], String[][])

```
public boolean storeRequested(org.dvb.application.storage.ApplicationStorageController service,
org.dvb.application.AppAttributes[] app, boolean[] autoStart, java.lang.String[][] args)
```

Called as part of the execution of ApplicationStorageController.store(AppProxy[],...).

Parameters:

service - the service for which store was called.

app - the applications to be installed.

autoStart - the array of booleans as passed in the call to the store method.

args - an copy of the corresponding parameter passed to the store method where each entry is .equals to the corresponding entry passed in the call to the store method.

Returns:

true if the storage request is permitted, otherwise false.

### storeRequested(ApplicationStorageController, AppAttributes, boolean, String[])

```
public boolean storeRequested(org.dvb.application.storage.ApplicationStorageController service,
org.dvb.application.AppAttributes app, boolean autoStart, java.lang.String[] args)
```

Called as part of the execution of ApplicationStorageController.store(AppProxy, etc.).

Parameters:

service - the service for which store was called.

app - the application to be installed.

autoStart - the parameter as passed in the call to the store method.

args - an array of strings where each entry is .equals to the corresponding entry passed in the call to the store method.

Returns:

true if the storage request is permitted, otherwise false.

### storeRequested(Locator, ApplicationStorageController, AppAttributes[], boolean[], String[][])

```
public boolean storeRequested(org.davic.net.Locator fromService,
org.dvb.application.storage.ApplicationStorageController toService,
org.dvb.application.AppAttributes[] app, boolean[] autoStart, java.lang.String[][] args)
```

Called as part of the execution of ApplicationStorageController.store(Locator,AppID[], etc.).

#### Parameters:

`fromService` - the service containing the applications to be stored.

`toService` - the service into which the applications are to be stored.

`app` - the applications to be installed.

`autoStart` - the array of booleans as passed in the call to the store method.

`args` - an copy of the corresponding parameter passed to the store method where each entry is .equals to the corresponding entry passed in the call to the store method.

#### Returns:

true if the storage request is permitted, otherwise false.

# org.dvb.application.privileged ApplicationStorageManager

#### Declaration

```
public class ApplicationStorageManager extends org.ocap.application.AppManagerProxy
```

```
java.lang.Object
  |
  +--org.ocap.application.AppManagerProxy
        |
        +--org.dvb.application.privileged.ApplicationStorageManager
```

#### Description

Provides application management features and capabilities only appropriate for privileged applications. On systems where this class is supported, org.ocap.application.AppManagerProxy.getInstance shall return an instance of ApplicationStorageManager.

#### Constructors

### ApplicationStorageManager()

```
protected ApplicationStorageManager()
```

This constructor is not to be used by applications.

#### Methods

### registerUnboundApp(Element)

```
public void registerUnboundApp(org.dvb.xml.jdom.Element element)
```

Registers new unbound application entries.

#### Parameters:

`element` - an ApplicationList element containing the new or updated XAIT information.

Throws:

`java.lang.SecurityException` - if the caller does not have MonitorAppPermission("registrar").

`java.lang.IllegalArgumentException` - if the element is not a root element whose value is "Application-List".

### setApplicationStorageHandler(ApplicationStorageHandler)

```
public void setApplicationStorageHandler(org.dvb.application.privileged.ApplicationStorageHandler handler)
```

Registers an ApplicationStorageHandler to become part of decisions about permitting the storing and removing of applications and services using the facilities provided by the org.dvb.application.storage package. Only one handler can be registered at one time hence multiple calls to this method shall replace the handler set in the previous call. No handler being set shall be equivalent to a handler which always permits requests to succeed.

Parameters:

`handler` - either an ApplicationStorageHandler or null to remove the current handler if one is set.

Throws:

`java.lang.SecurityException` - if the calling application does not have MonitorAppPermission("registrar").

# Annex AU (normative):
# AU.1. IPTV content referencing API

# Package

# org.dvb.locator.ip

Description

Contains transport dependent locators for access to IP services.

| Class Summary | |
|---|---|
| **Classes** | |
| MulticastLocator | Represents a reference to an IP service accessed by an IGMP join request. |
| RTSPLocator | Represents a reference to an IP service accessed by RTSP. |
| UnicastLocator | Represents a reference to an IP service delivered by unicast UDP where the delivery of the service is not initiated by the implementation. |

# org.dvb.locator.ip

# MulticastLocator

Declaration

```
public class MulticastLocator extends org.davic.net.Locator
```

```
java.lang.Object
  |
  +--org.davic.net.Locator
        |
        +--org.dvb.locator.ip.MulticastLocator
```

All Implemented Interfaces:

```
javax.tv.locator.Locator
```

Description

Represents a reference to an IP service accessed by an IGMP join request.

Constructors

MulticastLocator(InetAddress, int)

```
public MulticastLocator(java.net.InetAddress address, int port)
throws MalformedURLException
```

Creates a `MulticastLocator` from the specified URL.

Parameters:

`address` - the multicast address to join.

`port` - the port to join.

Throws:

`java.net.MalformedURLException` - if the address is not a multicast address.

### MulticastLocator(InetAddress, int, int)

```
public MulticastLocator(java.net.InetAddress address, int port, int serviceID)
throws MalformedURLException, IllegalArgumentException
```

Creates a `MulticastLocator` from the specified URL.

Parameters:

`address` - the multicast address to join.

`port` - the port to join.

`serviceID` - a DVB-SI serviceID or MPEG program number to identify one service in a multi-program transport stream.

Throws:

`java.net.MalformedURLException` - if the address is not a multicast address.

`java.lang.IllegalArgumentException` - if the service ID is less than 0 or greater than 0xffff.

### Methods

### getServiceID()

```
public int getServiceID()
```

Returns the serviceID specified when this `MulticastLocator` was constructed.

Returns:

the serviceID specified when this `MulticastLocator` or -1 if no serviceID was specified.

# org.dvb.locator.ip

# RTSPLocator

### Declaration

```
public class RTSPLocator extends org.davic.net.Locator
```

```
java.lang.Object
  |
  +--org.davic.net.Locator
        |
        +--org.dvb.locator.ip.RTSPLocator
```

### All Implemented Interfaces:

```
javax.tv.locator.Locator
```

### Description

Represents a reference to an IP service accessed by RTSP.

### Constructors

### RTSPLocator(String)

```
public RTSPLocator(java.lang.String url)
throws MalformedURLException
```

Creates a `RTSPLocator` from the specified URL.

Parameters:

`url` - an "rtsp:" url.

Throws:

`java.net.MalformedURLException` - if the url is not a correct "rtsp:" url.

### RTSPLocator(String, int)

```
public RTSPLocator(java.lang.String url, int serviceID)
throws MalformedURLException, IllegalArgumentException
```

Creates a `RTSPLocator` from the specified URL.

Parameters:

`url` - an "rtsp:" url.

`serviceID` - a DVB-SI serviceID or MPEG program number to identify one service in a multi-program transport stream.

Throws:

`java.net.MalformedURLException` - if the url is not a correct "rtsp:" url.

`java.lang.IllegalArgumentException` - if the service ID is less than 0 or greater than 0xffff.

### Methods

### getServiceID()

```
public int getServiceID()
```

Returns the serviceID specified when this `RTSPLocator` was constructed.

Returns:

the serviceID specified when this `RTSPLocator` or -1 if no serviceID was specified.

# org.dvb.locator.ip

# UnicastLocator

### Declaration

```
public class UnicastLocator extends org.davic.net.Locator
```

```
java.lang.Object
  |
  +--org.davic.net.Locator
        |
        +--org.dvb.locator.ip.UnicastLocator
```

### All Implemented Interfaces:

```
javax.tv.locator.Locator
```

### Description

Represents a reference to an IP service delivered by unicast UDP where the delivery of the service is not initiated by the implementation. The only requirement on implementations when locators of this type are used is to listen on the specified port number for content to arrive.

The intended use case for this locator is as follows:

- the application picks a local port number;
- the application uses this locator to tell the implementation to listen for an IP service delivered on that local port number;

- the application obtains the IP address of the device on which it's running - e.g. via java.net.NetworkInterface;
- the application passes that port number and the IP address to a VoD server as part of a request for some content;
- the VoD server delivers the requested content to the specified IP address and port;
- the implementation detects the arrival of the content on the port and starts processing and presenting it.

The external form of this locator is not defined by GEM.

### Constructors

#### UnicastLocator(int)

```
public UnicastLocator(int port)
```

Creates a `UnicastLocator` from the specified local port number.

**Parameters:**

`port` - the local port number on which the content will be delivered.

#### UnicastLocator(int, int)

```
public UnicastLocator(int port, int serviceID)
throws MalformedURLException, IllegalArgumentException
```

Creates a `UnicastLocator` from the specified local port number and service ID.

**Parameters:**

`port` - the local port number on which the content will be delivered.

`serviceID` - a DVB-SI serviceID or MPEG program number to identify one service in a multi-program transport stream.

**Throws:**

`java.net.MalformedURLException` - if the port number is not valid.

`java.lang.IllegalArgumentException` - if the service ID is less than 0 or greater than 0xffff.

### Methods

#### getPort()

```
public int getPort()
```

Returns the port specified when this `UnicastLocator` was constructed.

**Returns:**

the port specified when this `UnicastLocator` or -1 if no port was specified.

#### getServiceID()

```
public int getServiceID()
```

Returns the serviceID specified when this `UnicastLocator` was constructed.

**Returns:**

the serviceID specified when this `UnicastLocator` or -1 if no serviceID was specified.

# AU.2. OTT content referencing API

# Package org.dvb.locator.ott

### Description

Contains transport dependent locators for access to OTT services.

| Class Summary | |
|---|---|
| **Classes** | |
| HTTPLocator | Represents a reference to an IP service accessed by HTTP. |

# org.dvb.locator.ott HTTPLocator

### Declaration

```
public class HTTPLocator extends org.davic.net.Locator
```

```
java.lang.Object
  |
  +--org.davic.net.Locator
        |
        +--org.dvb.locator.ott.HTTPLocator
```

### All Implemented Interfaces:

```
javax.tv.locator.Locator
```

### Description

Represents a reference to an IP service accessed by HTTP.

### Constructors

### HTTPLocator(String)

```
public HTTPLocator(java.lang.String url)
throws MalformedURLException
```

Creates a `HTTPLocator` from the specified URL.

### Parameters:

`url` - an "HTTP:" url.

### Throws:

`java.net.MalformedURLException` - if the url is not a correct "HTTP:" url.

### HTTPLocator(String, int)

```
public HTTPLocator(java.lang.String url, int serviceID)
throws MalformedURLException, IllegalArgumentException
```

Creates a `HTTPLocator` from the specified URL.

### Parameters:

`url` - an "HTTP:" url.

`serviceID` - a DVB-SI serviceID or MPEG program number to identify one service in a multi-program transport stream.

Throws:

`java.net.MalformedURLException` - if the url is not a correct "HTTP:" url.

`java.lang.IllegalArgumentException` - if the service ID is less than 0 or greater than 0xffff.

## Methods

### getServiceID()

```
public int getServiceID()
```

Returns the serviceID specified when this `HTTPLocator` was constructed.

Returns:

the serviceID specified when this `HTTPLocator` or -1 if no serviceID was specified.

# Annex AV (normative): Extended service list API

# Package

# org.dvb.service

### Description

Contains features which are logically extensions to those provided by the javax.tv.service package.

| Class Summary | |
|---|---|
| **Interfaces** | |
| `TransportDependentService` | Extensions to `Service` specific to transport dependent services. |
| `TransportIndependentService` | Extensions to `Service` specific to transport independent services. |
| `UnboundApplicationService` | Represents a service used to contain unbound applications. |
| **Classes** | |
| `DvbSIManager` | Extensions to `SIManager` for environments where a single device may support content delivery mechanisms with different types of delivery system. |
| `IPServiceInformationType` | IP extensions to the Java TV service information type |

# org.dvb.service

# DvbSIManager

### Declaration

```
public abstract class DvbSIManager extends javax.tv.service.SIManager implements
org.dvb.service.sds.SDSRecordAccess, org.dvb.tvanytime.javatv.CRIDAccess

java.lang.Object
  |
  +--javax.tv.service.SIManager
        |
        +--org.dvb.service.DvbSIManager
```

### All Implemented Interfaces:

```
org.dvb.tvanytime.javatv.CRIDAccess, org.dvb.service.sds.SDSRecordAccess
```

### Description

Extensions to `SIManager` for environments where a single device may support content delivery mechanisms with different types of delivery system.

### Constructors

### DvbSIManager()

```
protected DvbSIManager()
```

Constructs a DvbSIManager object

### Methods

#### getTransports(DeliverySystemType)

```
public abstract javax.tv.service.transport.Transport[]
getTransports(javax.tv.service.navigation.DeliverySystemType type)
```

Reports the various content delivery mechanisms currently available on this platform which are accessed via delivery systems of the specified type. If no such transports are available then an array of length zero is returned.

Parameters:

`type` - a delivery system type.

Returns:

an array of Transport objects representing the content delivery mechanisms currently available on this platform accessed via delivery systems of the specified type.

# org.dvb.service

# IPServiceInformationType

### Declaration

```
public class IPServiceInformationType extends javax.tv.service.ServiceInformationType
```

```
java.lang.Object
  |
  +--javax.tv.service.ServiceInformationType
        |
        +--org.dvb.service.IPServiceInformationType
```

### Description

IP extensions to the Java TV service information type.

### Fields

#### BCG

```
public static final org.dvb.service.IPServiceInformationType BCG
```

Service information type for DVB Broadband Content Guide.

#### SDandS

```
public static final org.dvb.service.IPServiceInformationType SDandS
```

Service information type for DVB SD&S.

### Constructors

#### IPServiceInformationType(String)

```
protected IPServiceInformationType(java.lang.String name)
```

Creates a service information type object.

Parameters:

`name` - the string name of this type.

# org.dvb.service TransportDependentService

### Declaration

```
public interface TransportDependentService extends javax.tv.service.Service
```

### All Superinterfaces:

```
javax.tv.service.Service
```

### Description

Extensions to `Service` specific to transport dependent services.

### Methods

#### getDeliverySystemType()

```
public javax.tv.service.navigation.DeliverySystemType getDeliverySystemType()
```

Return the delivery system type by which the service is accessed.

#### Returns:

a `DeliverySystemType`

#### retrieveDetails(SIRequestor)

```
public javax.tv.service.SIRequest retrieveDetails(javax.tv.service.SIRequestor requestor)
```

This method retrieves additional information about the Service. This information is retrieved from the service information in the network. This method returns data asynchronously.

#### Overrides:

`retrieveDetails` in interface `Service`

#### Parameters:

`requestor` - The SIRequestor to be notified when this retrieval operation completes.

#### Returns:

An SIRequest object identifying this asynchronous retrieval request.

# org.dvb.service TransportIndependentService

### Declaration

```
public interface TransportIndependentService extends javax.tv.service.Service
```

### All Superinterfaces:

```
javax.tv.service.Service
```

### Description

Extensions to `Service` specific to transport independent services.

### Methods

#### transformService()

```
public org.dvb.service.TransportDependentService[] transformService()
```

Returns all the `TransportDependentService`s to which this Service can be resolved.

**Returns:**

an array of `TransportDependentService` instances.

#### transformService(DeliverySystemType)

```
public org.dvb.service.TransportDependentService[]
transformService(javax.tv.service.navigation.DeliverySystemType type)
```

Return those `TransportIndependentService`s to which this Service can be resolved which are accessed via the specified `DeliverySystemType`. If there are no such Services then an array of length zero is returned.

**Parameters:**

`type` - a `DeliverySystemType` instance.

**Returns:**

an array of TransportDependentService instances.

# org.dvb.service

# UnboundApplicationService

### Declaration

```
public interface UnboundApplicationService extends org.ocap.service.AbstractService,
org.dvb.application.storage.ApplicationStorageController
```

### All Superinterfaces:

```
org.ocap.service.AbstractService, org.dvb.application.storage.ApplicationStorageController,
javax.tv.service.Service
```

### Description

Represents a service used to contain unbound applications.

**Since:**

MHP 1.2.

### Methods

#### getPriority()

```
public int getPriority()
```

Get the resource priority for this service between 0 and 255.

**Returns:**

the resource priority for this service.

#### setPriority(int)

```
public void setPriority(int priority)
```

Set the resource priority for this service between 0 and 255.

Parameters:

`priority` - the resource priority for this service.

Throws:

`java.lang.IllegalArgumentException` - if the priority is outside this specified range.

# Package org.dvb.service.navigation

Description

This package contains features which are logically extensions to those provided by the javax.tv.service.navigation package. These are mainly additional ways of filtering the service list. These are focused on more complex situations than found at the time JavaTV was developed, e.g. devices with multiple delivery system types or multiple service providers or multiple video and audio codecs.

| Class Summary | |
|---|---|
| **Classes** | |
| `CompatibleServiceFilter` | CompatibleServiceFilter represents a ServiceFilter based on the capabilities of the receiver. |
| `DeliverySystemTypeFilter` | DeliverySystemTypeFilter represents a ServiceFilter based on a particular DeliverySystemType. |
| `IPDeliverySystemType` | IP extensions to the Java TV delivery system type |
| `ServiceProviderFilter` | ServiceProviderFilter represents a ServiceFilter based on a particular ServiceProviderInformation. |

# org.dvb.service.navigation CompatibleServiceFilter

Declaration

```
public final class CompatibleServiceFilter extends javax.tv.service.navigation.ServiceFilter

java.lang.Object
  |
  +--javax.tv.service.navigation.ServiceFilter
        |
        +--org.dvb.service.navigation.CompatibleServiceFilter
```

Description

CompatibleServiceFilter represents a ServiceFilter based on the capabilities of the receiver. A ServiceList resulting from this filter will include only services which are compatible with this receiver.

- If the service is scrambled by a conditional access system or equivalent, the service is only compatible if the receiver would be able to descramble the service given appropriate rights or entitlements.
- If the service includes one or more streams of type video, it is only compatible if the receiver can decode at least one stream.
- If the service includes one or more streams of type audio, it is only compatible if the receiver can decode at least one stream.

### Constructors

#### CompatibleServiceFilter()

```
public CompatibleServiceFilter()
```

Constructs the filter.

### Methods

#### accept(Service)

```
public boolean accept(javax.tv.service.Service service)
```

Tests if the given service passes the filter.

Overrides:

`accept` in class `ServiceFilter`

Parameters:

`service` - An individual `Service` to be evaluated against the filtering algorithm.

Returns:

true if service is compatible with this receiver; false otherwise.

# org.dvb.service.navigation DeliverySystemTypeFilter

### Declaration

```
public final class DeliverySystemTypeFilter extends javax.tv.service.navigation.ServiceFilter
```

```
java.lang.Object
  |
  +--javax.tv.service.navigation.ServiceFilter
        |
        +--org.dvb.service.navigation.DeliverySystemTypeFilter
```

### Description

DeliverySystemTypeFilter represents a ServiceFilter based on a particular DeliverySystemType. A ServiceList resulting from this filter will include only Service objects that can be accessed via delivery systems of the specified `DeliverySystemType`.

### Constructors

#### DeliverySystemTypeFilter(DeliverySystemType)

```
public DeliverySystemTypeFilter(javax.tv.service.navigation.DeliverySystemType type)
```

Constructs the filter based on a particular DeliverySystemType.

Parameters:

`type` - A `DeliverySystemType` object indicating the delivery system type for the services to be included in a resulting service list.

### Methods

#### accept(Service)

```
public boolean accept(javax.tv.service.Service service)
```

Tests if the given service passes the filter.

Overrides:

accept in class ServiceFilter

Parameters:

service - An individual Service to be evaluated against the filtering algorithm.

Returns:

true if service can be accessed via a delivery system of the specified type; false otherwise.

### getFilterValue()

```
public javax.tv.service.navigation.DeliverySystemType getFilterValue()
```

Reports the DeliverySystemType used to create this filter.

Returns:

The DeliverySystemType used to create this filter.

# org.dvb.service.navigation

# IPDeliverySystemType

### Declaration

```
public class IPDeliverySystemType extends javax.tv.service.navigation.DeliverySystemType
```

```
java.lang.Object
  |
  +--javax.tv.service.navigation.DeliverySystemType
        |
        +--org.dvb.service.navigation.IPDeliverySystemType
```

### Description

IP extensions to the Java TV delivery system type.

### Fields

#### IP

```
public static final org.dvb.service.navigation.IPDeliverySystemType IP
```

Delivery system type for IP.

### Constructors

#### IPDeliverySystemType(String)

```
protected IPDeliverySystemType(java.lang.String name)
```

Creates a delivery system type object.

Parameters:

name - the string name of this type.

# org.dvb.service.navigation ServiceProviderFilter

### Declaration

```
public final class ServiceProviderFilter extends javax.tv.service.navigation.ServiceFilter

java.lang.Object
  |
  +--javax.tv.service.navigation.ServiceFilter
        |
        +--org.dvb.service.navigation.ServiceProviderFilter
```

### Description

ServiceProviderFilter represents a ServiceFilter based on a particular ServiceProviderInformation. A ServiceList resulting from this filter will include only Service objects that can be provided by the specified service provider.

### Constructors

#### ServiceProviderFilter(ServiceProviderInformation)

```
public ServiceProviderFilter(javax.tv.service.navigation.ServiceProviderInformation provider)
```

Constructs the filter based on a particular ServiceProviderInformation.

#### Parameters:

`provider` - A `ServiceProviderInformation` object indicating the service provider for the services to be included in a resulting service list.

### Methods

#### accept(Service)

```
public boolean accept(javax.tv.service.Service service)
```

Tests if the given service passes the filter.

#### Overrides:

`accept` in class `ServiceFilter`

#### Parameters:

`service` - An individual `Service` to be evaluated against the filtering algorithm.

#### Returns:

true if service can be provided by the specified service provider; false otherwise.

#### getFilterValue()

```
public javax.tv.service.navigation.ServiceProviderInformation getFilterValue()
```

Reports the `ServiceProviderInformation` used to create this filter.

#### Returns:

The `ServiceProviderInformation` used to create this filter.

# Annex AW (normative): API to DVB service discovery and selection

# Package

# org.dvb.service.sds

### Description

Extensions to the JavaTV SI API giving access to information signalled according to DVB's service discovery and selection (SD&S) specification.

| Class Summary | |
|---|---|
| **Interfaces** | |
| SDSRecordAccess | This interface provides access to Service Discovery and Selection Service Provider and Service Offering Records. |
| SDSRecordChangeListener | This interface is implemented by applications wishing to receive notification of changes to SD&S records |
| **Classes** | |
| SDSRecordChangeEvent | A SDSRecordEvent notifies an SDSRecordListener of changes detected in SD&S records. |

# org.dvb.service.sds

# SDSRecordAccess

### Declaration

```
public interface SDSRecordAccess
```

### All Known Implementing Classes:

```
org.dvb.service.DvbSIManager
```

### Description

This interface provides access to Service Discovery and Selection Service Provider and Service Offering Records. It is intended to be implemented by an instance of org.dvb.service.SIManager.

### Methods

#### addSDSRecordListener(SDSRecordChangeListener)

```
public void addSDSRecordListener(org.dvb.service.sds.SDSRecordChangeListener listener)
```

Register a listener for events based on changes to SD&S records.

#### Parameters:

`listener` - the SDS Record Listener to be registered.

#### getServiceOfferingRecords(String)

```
public org.dvb.xml.jdom.Element[] getServiceOfferingRecords(java.lang.String domainName)
```

Provides access to the SD&S Service Offering records for a particular Service Provider.

Parameters:

`domainName` - Domain name of the service provider for which offerings are requested.

Returns:

an array of SD&S Service Offering Records, including BCG records.

### getServiceProviderRecords()

```
public org.dvb.xml.jdom.Element[] getServiceProviderRecords()
```

Provides access to the SD&S Service Provider records.

Returns:

an array of SD&S Service Provider Records.

### removeSDSRecordListener(SDSRecordChangeListener)

```
public void removeSDSRecordListener(org.dvb.service.sds.SDSRecordChangeListener listener)
```

Removes a listener for events based on changes to SD&S records.

Parameters:

`listener` - the SDS Record Listener to be removed.

# org.dvb.service.sds

# SDSRecordChangeEvent

### Declaration

```
public abstract class SDSRecordChangeEvent extends java.util.EventObject
```

```
java.lang.Object
  |
  +--java.util.EventObject
        |
        +--org.dvb.service.sds.SDSRecordChangeEvent
```

### All Implemented Interfaces:

```
java.io.Serializable
```

### Description

A SDSRecordEvent notifies an SDSRecordListener of changes detected in SD&S records. Specifically, this event signals the addition, removal, or modification of an SD&S record.

### Constructors

### SDSRecordChangeEvent(DvbSIManager, SIChangeType, Element)

```
public SDSRecordChangeEvent(org.dvb.service.DvbSIManager siManager,
javax.tv.service.SIChangeType type, org.dvb.xml.jdom.Element sdsRecord)
```

Constructs an `SDSRecordChangeEvent` object.

Parameters:

`siManager` - the SIManager on which the Event initially occurred.

`type` - the type of change that occured.

`sdsRecord` - the SD&S record in which the change occurred.

### Methods

#### getChangeType()

```
public javax.tv.service.SIChangeType getChangeType()
```

Returns the type of change that occurred.

**Returns:**

SIChangeType the type of change that occurred.

#### getSDSRecord()

```
public org.dvb.xml.jdom.Element getSDSRecord()
```

Returns the SD&S record in which the change occured.

**Returns:**

Document XML document of the SD&S record in which the change occurred.

# org.dvb.service.sds

# SDSRecordChangeListener

### Declaration

```
public interface SDSRecordChangeListener extends java.util.EventListener
```

### All Superinterfaces:

```
java.util.EventListener
```

### Description

This interface is implemented by applications wishing to receive notification of changes to SD&S records.

### Methods

#### notifyChange(SDSRecordChangeEvent)

```
public void notifyChange(org.dvb.service.sds.SDSRecordChangeEvent event)
```

Notification of a change in either an SDS Service Provider or Service Offering record.

**Parameters:**

`event` - the event that caused the change.

# Annex AX (normative): API to DVB broadband content guide

In the present document, `DvbLocator` references are to be replaced with references to `org.davic.net.Locator` or an implementation-defined subclass.

# Package org.dvb.tvanytime.metadata.ip

### Description

Provides access to Broadband Content Guides delivered via IP.

| Class Summary | |
|---|---|
| **Classes** | |
| `BCGDiscovery` | Class providing access to Broadband Content Guides whose location is signalled by IP-based Service Discovery & Selection information. |
| `HTTPDatabase` | Class providing access to a Broadband Content Guide delivered using the unicast delivery method specified in TS 102 034 [80] (V1.2.1). |
| `MulticastDatabase` | Class providing access to a Broadband Content Guide delivered using the multicast delivery method specified in TS 102 034 [80] (V1.2.1). |

# org.dvb.tvanytime.metadata.ip BCGDiscovery

### Declaration

```
public abstract class BCGDiscovery

java.lang.Object
  |
  +--org.dvb.tvanytime.metadata.ip.BCGDiscovery
```

### Description

Class providing access to Broadband Content Guides whose location is signalled by IP-based Service Discovery & Selection information.

### Constructors

#### BCGDiscovery()

```
public BCGDiscovery()
```

This constructor is provided for implementations and should not be used by GEM applications.

### Methods

#### getAvailableDatabases(CRID)

```
public static org.dvb.tvanytime.metadata.Database[]
getAvailableDatabases(org.dvb.tvanytime.resolution.CRID crid)
throws DatabaseException
```

Get an array of Databases that can provide metadata for the specified CRID Authority.

Parameters:

crid - the authority field from the supplied CRID is used to decide which metadata servers to return. The CRID authority should be matched against Metadata Pointer Descriptors in Resolution Provider Notification Tables delivered by IP-based Service Discovery & Selection information or in a DVB service.

Returns:

an array of type org.dvb.tvanytime.metadata.Database . Each object in the returned array will be an instance of either MulticastDatabase , HTTPDatabase or org.dvb.tvanytime.metadata.IPDatabase.

Throws:

org.dvb.tvanytime.metadata.DatabaseException - if no metadata services can be found for the specified scope.

### getAvailableDatabases(Locator)

```
public static org.dvb.tvanytime.metadata.Database[] getAvailableDatabases(javax.tv.locator.Locator
locator)
throws DatabaseException
```

Get an array of Databases that can provide metadata for the specified service.

Parameters:

locator - a locator representing a service. This should be matched against entries in IP-based Service Discovery & Selection information to find Broadband Content Guides identified by ServiceDescriptionLocation and Services-DescriptionLocation fields.

Returns:

an array of type org.dvb.tvanytime.metadata.Database . Each object in the returned array will be an instance of either MulticastDatabase , HTTPDatabase or org.dvb.tvanytime.metadata.IPDatabase.

Throws:

org.dvb.tvanytime.metadata.DatabaseException - if no metadata services can be found for the specified scope.

### getAvailableDatabases(String)

```
public static org.dvb.tvanytime.metadata.Database[] getAvailableDatabases(java.lang.String
serviceProviderDomainName)
throws DatabaseException
```

Get an array of Databases that can provide metadata for the specified service provider.

Parameters:

serviceProviderDomainName - a String representing the domain name of a service provider. This should be matched against the TargetProvider fields in Broadband Content Guide Discovery Records delivered by IP-based Service Discovery & Selection information.

Returns:

an array of type org.dvb.tvanytime.metadata.Database . Each object in the returned array will be an instance of either MulticastDatabase , HTTPDatabase or org.dvb.tvanytime.metadata.IPDatabase.

Throws:

org.dvb.tvanytime.metadata.DatabaseException - if no metadata services can be found for the specified scope.

### getAvailableDatabasesByProvider(String)

```
public static org.dvb.tvanytime.metadata.Database[] getAvailableDatabasesByProvider(java.lang.String
BCGProviderName)
throws DatabaseException
```

Get an array of Databases from the specified BCG provider.

#### Parameters:

`BCGProviderName` - a String representing the domain name of the BCG provider. This should be matched against the BCGProviderName in Broadband Content Guide Discovery Records delivered by IP-based Service Discovery & Selection information.

#### Returns:

an array of type `org.dvb.tvanytime.metadata.Database` . Each object in the returned array will be an instance of either `MulticastDatabase` , `HTTPDatabase` or `org.dvb.tvanytime.metadata.IPDatabase`.

#### Throws:

`org.dvb.tvanytime.metadata.DatabaseException` - if no metadata services can be found for the specified scope.

### getDatabase()

```
public static org.dvb.tvanytime.metadata.Database getDatabase()
throws DatabaseException
```

Get an instance of a Broadband Content Guide where the platform chooses the host.

#### Returns:

a Database.

#### Throws:

`org.dvb.tvanytime.metadata.DatabaseException` - if a metadata service cannot be found.

# org.dvb.tvanytime.metadata.ip

# HTTPDatabase

#### Declaration

```
public abstract class HTTPDatabase implements org.dvb.tvanytime.metadata.Database

java.lang.Object
  |
  +--org.dvb.tvanytime.metadata.ip.HTTPDatabase
```

#### All Implemented Interfaces:

`org.dvb.tvanytime.metadata.Database`

#### Description

Class providing access to a Broadband Content Guide delivered using the unicast delivery method specified in TS 102 034 [80] (V1.2.1).

#### Constructors

### HTTPDatabase()

```
public HTTPDatabase()
```

This constructor is provided for implementations and should not be used by GEM applications.

### Methods

#### getURL()

```
public org.davic.net.Locator getURL()
```

Get the URL of the host providing this HTTPDatabase.

Returns:

the URL.

#### newInstance(String)

```
public static org.dvb.tvanytime.metadata.ip.HTTPDatabase newInstance(java.lang.String path)
throws DatabaseException
```

Get an instance of an HTTPDatabase where the caller chooses the server.

Parameters:

`path` - a URL that points to the metadata server.

Returns:

an HTTPDatabase.

Throws:

`org.dvb.tvanytime.metadata.DatabaseException` - if the platform is unable to connect to the remote server.

# org.dvb.tvanytime.metadata.ip MulticastDatabase

### Declaration

```
public abstract class MulticastDatabase implements org.dvb.tvanytime.metadata.MonitoredDatabase
```

```
java.lang.Object
  |
  +--org.dvb.tvanytime.metadata.ip.MulticastDatabase
```

### All Implemented Interfaces:

```
org.dvb.tvanytime.metadata.Database, org.dvb.tvanytime.metadata.MonitoredDatabase
```

### Description

Class providing access to a Broadband Content Guide delivered using the multicast delivery method specified in TS 102 034 [80] (V1.2.1).

### Constructors

#### MulticastDatabase()

```
public MulticastDatabase()
```

This constructor is provided for implementations and should not be used by GEM applications.

### Methods

#### getURL()

```
public org.davic.net.Locator getURL()
```

Get the URL of the host providing this MulticastDatabase.

Returns:

the URL.

newInstance(String)

```
public static org.dvb.tvanytime.metadata.ip.MulticastDatabase newInstance(java.lang.String path)
throws DatabaseException
```

Get an instance of a MulticastDatabase where the caller chooses the server.

Parameters:

`path` - a URL that points to the metadata server.

Returns:

a MulticastDatabase.

Throws:

`org.dvb.tvanytime.metadata.DatabaseException` - if the platform is unable to locate the metadata service.

# Annex AY (normative): TV-Anytime and Java TV Integration

# Package org.dvb.tvanytime.javatv

Description

Contains TV-Anytime specific extensions to javax.tv.*.

| Class Summary | |
|---|---|
| **Interfaces** | |
| CRIDAccess | This interface extends javax.tv.service.SIManager with overloaded methods that take CRID and InstanceMetadataId as parameters. |
| TVAProgramEvent | Extensions to ProgramEvent to enable access to extended TV-Anytime data associated with the event. |

# org.dvb.tvanytime.javatv CRIDAccess

Declaration

```
public interface CRIDAccess
```

All Known Implementing Classes:

```
org.dvb.service.DvbSIManager
```

Description

This interface extends javax.tv.service.SIManager with overloaded methods that take CRID and InstanceMetadataId as parameters. It is intended to be implemented by an instance of org.dvb.service.SIManager.

Methods

retrieveProgramEvent(CRID, String, SIRequestor)

```
public javax.tv.service.SIRequest retrieveProgramEvent(org.dvb.tvanytime.resolution.CRID crid,
java.lang.String instanceMetadataId, javax.tv.service.SIRequestor requestor)
throws SecurityException
```

Retrieves all ProgramEvent associated with the CRID and optionally the instance metadata Id.

Parameters:

crid - The TV-Anytime CRID associated with the ProgramEvent(s).

instanceMetadataId - The TV-Anytime Instance Metadata Id associated with the service or null.

requestor - the requestor object to be used for the asynchronous callback.

Returns:

the SIRequest object associated with the asynchronous request.

Throws:

java.lang.SecurityException.

### retrieveServiceDetails(CRID, String, SIRequestor)

```
public javax.tv.service.SIRequest retrieveServiceDetails(org.dvb.tvanytime.resolution.CRID crid,
java.lang.String instanceMetadataId, javax.tv.service.SIRequestor requestor)
throws SecurityException
```

Retrieves the `ServiceDetails` of all services associated with the CRID and optionally the instance metadata Id.

Parameters:

`crid` - The TV-Anytime CRID associated with the service(s).

`instanceMetadataId` - The TV-Anytime Instance Metadata Id associated with the service or null.

`requestor` - the requestor object to be used for the asynchronous callback.

Returns:

the SIRequest object associated with the asynchronous request.

Throws:

java.lang.SecurityException.

### retrieveSIElement(CRID, String, SIRequestor)

```
public javax.tv.service.SIRequest retrieveSIElement(org.dvb.tvanytime.resolution.CRID crid,
java.lang.String instanceMetadataId, javax.tv.service.SIRequestor requestor)
throws SecurityException
```

Retrieves all `SIElement` that are instances of `ProgramEvent` associated with the CRID and optionally the instance metadata Id.

Parameters:

`crid` - The TV-Anytime CRID associated with the SIElement(s).

`instanceMetadataId` - The TV-Anytime Instance Metadata Id associated with the service or null.

`requestor` - the requestor object to be used for the asynchronous callback.

Returns:

the SIRequest object associated with the asynchronous request.

Throws:

java.lang.SecurityException.

# org.dvb.tvanytime.javatv

# TVAProgramEvent

### Declaration

```
public interface TVAProgramEvent extends javax.tv.service.guide.ProgramEvent
```

### All Superinterfaces:

```
javax.tv.service.guide.ProgramEvent, javax.tv.service.SIElement, javax.tv.service.SIRetrievable
```

### Description

Extensions to `ProgramEvent` to enable access to extended TV-Anytime data associated with the event.

### Methods

#### getCRID()

```
public org.dvb.tvanytime.resolution.CRID getCRID()
```

Returns the `CRID` associated with this Program Event.

**Returns:**

the CRID associated with this Program Event.

#### getInstanceMetadataId()

```
public java.lang.String getInstanceMetadataId()
```

Returns the `InstanceMetadataId` associated with this Program Event (if specified) or null.

**Returns:**

the Instance Metadata Id associated with this Program Event.

#### getProgramInformationElement()

```
public org.dvb.xml.jdom.Element getProgramInformationElement()
```

Returns an XML Element from the relevant Program Information table entry.

**Returns:**

the TV-Anytime element that points to the start of the program information entry.

#### getProgramLocationElement()

```
public org.dvb.xml.jdom.Element getProgramLocationElement()
```

Returns an XML Element from the relevant Program Location table entry.

**Returns:**

the TV-Anytime element that points to the start of the program location entry.

# Annex AZ (normative): MHP terminal hardware API

# Package org.dvb.hardware

### Description

Contains features related to the GEM terminal itself.

| Class Summary | |
|---|---|
| **Classes** | |
| Terminal | Provides access to low level details of the GEM terminal. |

# org.dvb.hardware Terminal

### Declaration

```
public class Terminal

java.lang.Object
  |
  +--org.dvb.hardware.Terminal
```

### Description

Provides access to low level details of the GEM terminal.

### Constructors

#### Terminal()

```
protected Terminal()
```

This constructor is provided for the use of implementations and specifications which extend this specification. Applications shall not define sub-classes of this class. Implementations are not required to behave correctly if any such application defined sub-classes are used.

### Methods

#### getInstance()

```
public static org.dvb.hardware.Terminal getInstance()
```

Obtain an instance of the singleton Terminal object.

#### Returns:

the Terminal.

#### getRebootTimeout()

```
public int getRebootTimeout()
```

Get the time that the system shall wait for the SystemEventListener.notifyEvent(SystemEvent) to return before continuing regardless.

Returns:

the reboot timeout measured in seconds.

reinitialize()

```
public void reinitialize()
```

Re-initializes the GEM software environment including re-initializing all DVB-J virtual machine or machines and re-starting all GEM applications. The method SystemEventListener.notifyEvent(SystemEvent) shall be called before the re-initialization is performed by the GEM terminal. Once the notifyEvent method returns, the process shall continue. If the notify event method does not return within an implementation dependent time then the process may continue regardless.

Throws:

`java.lang.SecurityException` - if the caller does not have MonitorAppPermission("reboot").

reset()

```
public void reset()
```

Initiates a reset of the GEM terminal equivalent to a power cycle. The method SystemEventListener.notify-Event(SystemEvent) shall be called before the reboot is performed by the GEM terminal. Once the notifyEvent method returns, the reboot process shall continue. If the notify event method does not return within an implementation dependent time then the reboot process may continue regardless.

Throws:

`java.lang.SecurityException` - if the caller does not have MonitorAppPermission("reboot").

setRebootTimeout(int)

```
public void setRebootTimeout(int seconds)
```

Set the time that the system shall wait for the SystemEventListener.notifyEvent(SystemEvent) to return before it shall continue regardless. If not set then the default is implementation dependent.

Parameters:

`seconds` - the time to wait in seconds.

Throws:

`java.lang.SecurityException` - if the caller does not have MonitorAppPermission("reboot").

# Annex BA (normative): Content Download API

# Package

# org.oipf.download

Description

This package enables downloading of arbitrary content. It provides an extension of the shared DVR API [85].

| Class Summary | |
|---|---|
| **Classes** | |
| LocatorDownloadSpec | Specifies a recording request in terms of a Locator to a file. The identified file is asynchronously downloaded from the network. |
| ApplicationDownloadSpec | Represents a content download to be performed by an application |
| **Interfaces** | |
| ApplicationDownloadRequest | Represents a content download to be performed by an application. |
| **Exceptions** | |
| ApplicationDownloadException | Thrown when an application calls methods on ApplicationDownloadRequest which it is not permitted to call. |

## ApplicationDownloadRequest

```
package org.oipf.download
public interface ApplicationDownloadRequest
```

Represents a content download to be performed by an application. Requests of this type are handled totally by an application. They are created in the PENDING_NO_CONFLICT_STATE. The application handling the request is responsible for downloading the content and all resulting state changes.

Requests of this type shall always be visible to the application specified when the corresponding ApplicationDownloadSpec was created regardless of recording request specific security attributes. The methods defined in this class shall always fail with a ApplicationDownloadException if called by any application other than the one specified when the corresponding ApplicationDownloadSpec was created.

### getFile

```
java.io.RandomAccessFile getFile() throws ApplicationDownloadException
```

Return a file to which downloaded data can be written and from which downloaded data can be read.

Returns:

a File

Throws:

ApplicationDownloadException - if the caller is not permitted to call this method as defined above

### setFailedReason

```
void setFailedReason(int reason)
```

Set the reason to use for the exception returned by getFailedException. The reason must be one valid for the constructor of RecordingFailedException.

Parameters:

`reason` - the reason to use when constructing a RecordingFailedException

## setState

```
void setState(int state)
```

Set the state of the download.

Parameters:

`state` - the new state of the recording

Throws:

ApplicationDownloadException - if the caller is not permitted to call this method as defined above

# ApplicationDownloadSpec

```
java.lang.Object
  └ org.ocap.shared.dvr.RecordingSpec
      └ org.oipf.download.ApplicationDownloadSpec
```

```
package org.oipf.download
public class ApplicationDownloadSpec extends org.ocap.shared.dvr.RecordingSpec
```

Represents a content download to be performed by an application. The application to handle the request and the source of the content to download must be specified when the request is created. No other application is permitted to handle the request. The format of the source must be co-ordinated between the application requesting the download and the application handling the download request.

## ApplicationDownloadSpec

```
public ApplicationDownloadSpec(AppID app, java.lang.String source,  RecordingProperties properties)
```

Create a request for content download by application.

Parameters:

`app` - the application to perform the download

`source` - the source of the download

`properties` - the properties for the download

## getAppID

```
public AppID getAppID()
```

Return the application ID specified.

Returns:

an application ID

## getSource

```
public java.lang.String getSource()
```

Return the source specified.

Returns:

a String

# LocatorDownloadSpec

```
java.lang.Object
  └ org.ocap.shared.dvr.RecordingSpec
      └ org.oipf.download.LocatorDownloadSpec
```

```
package org.oipf.download
public class LocatorDownloadSpec extends org.ocap.shared.dvr.RecordingSpec
```

Specifies a recording request in terms of a Locator to a file. The identified file is asynchronously downloaded from the network.

When instances of this class are passed to RecordingManager.record(..), the following additional failure mode shall apply - if the locator does not reference a file then the record method shall throw an IllegalArgumentException.

No additional failure modes are defined for RecordingRequest.reschedule.

For recording requests resulting from a recording spec of this type, downloading shall start immediately. Such recording requests will never be in a pending or a IN_PROGRESS_INCOMPLETE_STATE state.

## LocatorDownloadSpec

```
public LocatorDownloadSpec(Locator source, RecordingProperties properties)
```

Constructor

Parameters:

`source` - Source of content to be downloaded

`properties` - the definition of how the recording is to be done

## getSource

```
public Locator getSource()
```

Returns the source of the content to be downloaded

Returns:

the source passed into the constructor

# ApplicationDownloadException

```
java.lang.Object
  └ java.lang.Throwable
```

```
└ java.lang.Exception
    └ org.oipf.download.ApplicationDownloadException
```

All Implemented Interfaces:

java.io.Serializable

```
package org.oipf.download
public class ApplicationDownloadException extends java.lang.Exception
```

Thrown when an application calls methods on ApplicationDownloadRequest which it is not permitted to call.

## ApplicationDownloadException

```
public ApplicationDownloadException()
```

Constructs a `ApplicationDownloadException` with `null` as its error detail message.

## ApplicationDownloadException

```
public ApplicationDownloadException(java.lang.String s)
```

Constructs a `ApplicationDownloadException` with the specified detail message. The error message string `s` can later be retrieved by the `Throwable.getMessage()` method of class `java.lang.Throwable`.

Parameters:

`s` - the detail message.

# History

| Document history | | |
|---|---|---|
| V1.0 | October 2010 | First draft after MUG F2F |
| V1.1 | November 2010 | Second draft |
| V1.2 | December 2010 | Third draft |
| V1.3 | January 2011 | Release candidate |
| | | |
| | | |