



17 March 2008
Michael Lagally
MUG chair

API differences between GEM 1.0.2 and GEM 1.0.3

1 Introduction

Several GEM based terminal specifications are currently based on GEM 1.0.2. Significant effort was invested by the DVB-MUG group to create the updated GEM 1.0.3, which contains clarifications, improved language, additional examples and a few minor API changes. The purpose of this document is to give a concise summary on these changes and allow interested parties to understand the motivation and to assess the risk of these changes.

2 Overview of all GEM 1.0.3 changes at implementation level

The following table lists all changes at API level between GEM 1.0.2 and GEM 1.0.3. The “Type of change” indicates, whether a change was made for editorial/documentation reasons or contains a semantically meaningful bugfix. A detailed discussion on the API changes at method signature level follows below.

Changed class	Type of change	Details
org.davic.media.MediaTimeEventControl	Semantically meaningful bugfix	Parameter clarification: Deregistering listeners See section 3 below
org.dvb.io.persistent.FileAttributes	Semantically meaningful bugfix	public FileAttributes(Date expiration_date, FileAccessPermissions p, int priority) {} See section 3 below
org.dvb.application.AppProxy.java	Semantically meaningful bugfix	Destroy behavior, restarting of an AppProxy is not possible See section 3 below
Org.havi.ui.HScene	Optional change	Change in MHP 1.0.3 Errata 3 is made optional in GEM A.1
org.dvb.media.VideoFormatControl	Optional change	Change in MHP 1.0.3 Errata 3 is made optional in GEM A.1
org.dvb.media.DripFeedPermission.java	Editorial/JavaDoc only	Improved documentation, no semantic change GEM A.1.7
org.davic.media.LanguageControl	Editorial/JavaDoc only	Improved documentation, no semantic change GEM A.1.5

org.davic.resources.ResourceClient	Editorial/JavaDoc only	Improved documentation, no semantic change GEM A.2.2
org.dvb.application.AppsDatabase	Editorial/JavaDoc only	Improved documentation, no semantic change GEM S.1
org.dvb.dsmcc.ServiceDomain	Editorial/JavaDoc only	Improved documentation, no semantic change GEM P.2.5.6
org.dvb.net.rc.RCInterface.java	Editorial/JavaDoc only	Type of return value GEM A.1.8
org.dvb.ui.DVBGraphics	Editorial/JavaDoc only	GEM U.2

3. Semantically meaningful bugfixes

The following sections give the background of the change as well as the corresponding GEM 1.0.3 sections for easier reading.

3.1 GEM1.0.3 : A.2 Errata to DAVIC

A.2.1 org.davic.media.MediaTimeEventControl - deregistering listeners

org.davic.media.MediaTimeEventControl	Semantically meaningful bugfix	Parameter clarification: Deregistering listeners
---------------------------------------	--------------------------------	--------------------------------------------------

Background: The original MHP API had no way for an xlet to deregister `MediaTimeEventListeners`. This, coupled with the requirement that an xlet deregister all listeners, represented a contradiction that effectively made the `MediaTimeEventControl` API unusable for a well-behaved xlet. This API was found to be necessary for content-bound xlets, e.g. in Blu-ray. The change described below fixes the problem by defining a de-registration mechanism, without introducing an API signature change. The change is optional for platforms to implement in order to accommodate devices produced prior to this bugfix; on such platforms, this API is potentially unusable.

Language in GEM 1.0.3:

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 the system property.

Calling this method with a value that does not match a previously registered positive media time shall have no effect.

NOTE: When this method is called with a `mediaTime` value of 0 the result is implementation dependent.

NOTE: 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: 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: Although this class isn't required by MHP or 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.

3.2 GEM1.0.3 : A.1 Errata to MHP

A.1.4 `org.dvb.io.persistent.FileAttributes` class

<code>org.dvb.io.persistent.FileAttributes</code>	Semantically meaningful bugfix	public <code>FileAttributes(Date expiration_date, FileAccessPermissions p, int priority) {}</code>
---------------------------------------------------	--------------------------------	-----------------------------------------------------------------------------------------------------------

Background: In the MHP spec maintenance process, it was discovered that a clerical error lead to the "public" keyword being omitted from this constructor. This fix has been known and reflected in the MHP errata for years. This bugfix was discussed and approved in extensive MUG discussions that were open to all, and well attended.

Language in GEM 1.0.3:

In `org.dvb.io.persistent.FileAttributes`, the following constructor shall be considered to be present;

```
/**
 * Constructor.
 *
 * @param expiration_date an expiration date or null
 * @param p the access permissions to use
 * @param priority the priority to use in persistent storage
 */
public FileAttributes(Date expiration_date, FileAccessPermissions p,
int priority) {}
```

3.3. GEM 1.0.3 Annex A.1

A.1.1 MHP Errata Document

org/dvb/application/AppProxy.java	Semantically meaningful bugfix	Destroy behavior, restarting of an AppProxy is not possible
-----------------------------------	--------------------------------	-------------------------------------------------------------

Background: In the MHP spec maintenance process, it was discovered that it's ambiguous what state an AppProxy goes into after the xlet it represents has finished being destroyed. Is the DESTROYED state terminal, thus requiring the caller to create a new AppProxy instance for the given xlet, or does the state get "recycled" to NOT_LOADED or another state? In MHP this was resolved in one particular way; in GEM, either behavior was allowed to accommodate legacy devices, and informative guidance for application authors was substituted. Xlet authors who follow this guidance will be guaranteed correct xlet behavior with either allowed implementation of the DESTROYED behavior.

Language in GEM 1.0.3:

The change in tm2971r2 clause 4.7.6 (issue 4258, "App Proxy") should be replaced with the following for GEM terminal specifications:

In the class description for AppProxy.DESTROYED, the following text;

The application is in the destroyed state. This state is transient and entry to this state shall be followed with a transition to the NOT_LOADED state almost immediately. It shall be possible to re-start the application after the transition to the NOT_LOADED state.

is replaced with;

An AppProxy for a terminated application is no longer usable, An AppProxy instance should be considered invalid, once it was in the AppProxy.DESTROYED state.

Note: If a terminated application needs to be restarted, an AppProxy for a new application instance may be obtained from the AppsDatabase.