

DVB-HTML

— an optional declarative language within MHP 1.1

Philippe Perrot
Canal+ Technologies

This is the fourth and final article in our current series on the use of XML technologies in broadcasting. The other three articles were published in the [June 2001](#) issue of EBU Technical Review.

Here, we present an overview of the DVB Multimedia Home Platform, version 1.1, which provides additional functionality over the earlier version 1.0.1. In particular, the article provides a description of the optional DVB-HTML application format.

1. Introduction

On the 6th June 2001, the DVB Steering Board officially adopted version 1.1 of the Multimedia Home Platform (MHP) specification. It closely followed the adoption of version 1.0.1 (5th April 2001), which was the result of a corrigenda process on version 1.0.

MHP 1.0.1 [1] specifies an extensive application execution environment for digital interactive TV terminals, independent of the underlying vendor-specific hardware and software. It capitalises on the existing DVB open standards to provide enhancements to the incoming and outgoing signals, necessary to guarantee an open and interoperable interactive TV environment. This execution environment is based on the use of a Java virtual machine and the definition of generic APIs that provide access to the interactive digital TV terminal's typical resources and facilities. A Java application using those generic APIs is called a DVB-J application.

MHP 1.1 [2] provides additional functionality to the MHP 1.0.1 platform, such as the ability to:

- download interactive applications from the return channel;
- support the management of interoperable plug-ins (for the support of legacy application formats);
- store applications in persistent memory;
- access non-CA-related smart card readers.

In addition, MHP 1.1 specifies the Internet Access profile, in which applications can control the basic operations of Open Internet resident clients (Web browser, e-mail and news client). Eventually, in addition to the Java-based DVB-J application format, a new optional application type has been defined and named DVB-HTML.

The present article provides a description of this DVB-HTML application format and its integration within the MHP platform. It focuses on different considerations such as the rationale for having specified such a language, the requirements that led to its specification, the main technical aspects, and some conformance considerations. It assumes that the reader has some preliminary knowledge of the MHP platform and W3C technology.



In the following, and depending on the context, use of the term “DVB-HTML” will refer to either the XML language as defined by the DVB-HTML Document Type Definition (DTD) or, more generally, to the set of language structures that constitute the DVB-HTML application format, i.e. XML, CSS and DOM.

2. Rationale/context

In June 1999, the DVB Steering Board took the decision to launch the work – within the MHP technical group – on defining a declarative application format which would be introduced as an MHP plug-in in the Enhanced Broadcast profile and would be optional in the Interactive Broadcast profile (for which a return channel is available).

The decision to integrate a new application format, even optionally, was the result of a long process within DVB. Some main reasons for introducing a declarative application format along with the DVB-J one were:

- As content providers are diversifying their target audience (Internet, TV, PDA, mobiles), it would allow them to work on a common authoring basis for text services, addressing those different devices;
- It would provide faster authoring time for simple applications mainly devoted to presentation purposes such as magazines, where such static applications are easily handled through declarative formats;
- It would benefit from the WWW authoring community and capitalise on existing content of the Web.

At this stage, there could have been a temptation to select directly the existing HTML-related technologies used on the Web (HTML, Javascript, DOM, CSS). However, they have been primarily designed for a desktop-oriented environment and not for a TV environment: layout and interaction mechanisms are a lot different for example. Also, this new application format had to comply with all existing requirements defined by the DVB MHP commercial group that were already fulfilled by the DVB-J format. This underlined the need for a completely new declarative language, based upon W3C specifications and complying with the DVB requirements.

3. Requirements

DVB-HTML was designed following the classical DVB process, i.e. commercial requirements were first provided by the MHP Commercial subgroup, then translated into technical requirements before working on the specification.

Among the main commercial requirements, the following drove most of the technical work:

- interoperability;
- robustness;
- alignment with W3C specifications;
- rejection of non-conformant content;
- compatibility with the MHP security model;
- minimization of the redundancy with existing MHP technology.

More fine-grained requirements were also provided such as:

- precise content layout control mechanisms;
- support of different pixel aspect ratios;
- the display of DVB-HTML content within a DVB-J application;
- embedded animation, composition, synchronization and content manipulation.

Note that the last requirement was partly fulfilled. Only the external synchronization with audio-visual sources has been eventually addressed. The remaining part has been left for a future version of the MHP specification.

When it comes to interoperability, HTML doesn't own any good records. In this matter, the Internet is a good example. Current browsers do not comply with the W3C specifications and the same content will hardly be rendered the same on different user agents. As a consequence, web application authors need to tailor their content to the most famous browsers, and doing so, increase the overall cost of content production and contribute to the verticalization of the market. As DVB aims at a true horizontal market for the whole TV transmission chain, it considers this fact as established, and its commercial requirements are to mandate the rejection of non-conformant content.

Fig. 1 depicts in a simplistic way the Internet-related technology domains and the position of the DVB-HTML work. It shows the mismatch between the technologies available on the web and the corresponding specifications from the W3C. Although the diagram puts into evidence the gap between the WWW de facto technologies and DVB-HTML, it is important to note that, due to its XML nature, DVB-HTML can easily be processed and displayed on a WWW browser (although the result will not be predictable if the browser is not conformant to the DVB-HTML specification), but the converse is not true, since Web content will be rejected if it has not been transcoded into DVB-HTML prior to transmission.

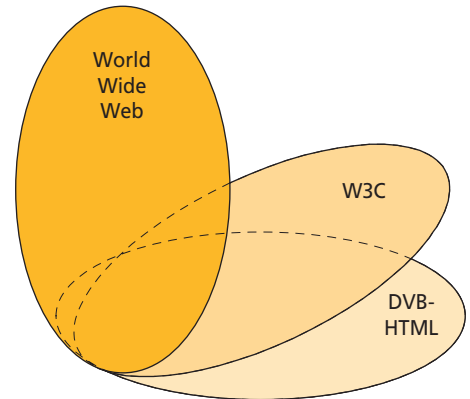


Figure 1
Function domain of the DVB-HTML specification.

4. DVB-HTML content formats

The W3C specifications (especially DOM and CSS) have not been conceived in a fine-grained modular way, which makes them difficult to subset. The effort of extracting a TV profile from them resulted in a situation where unnecessary features had to be incorporated for conformance reasons with W3C (for some examples, see *Section 4.4.4.*).

4.1. Basic Architecture

Fig. 2 shows the basic architecture of an MHP 1.1 implementation, and where the DVB-HTML option fits.

4.2. XML support

The DVB-HTML language is based on XML 1.0 [3]. XML is a meta-language used for the definition of mark-up based languages. It provides a platform-independent means for structuring data into textual form. The syntactic and grammatical rules of an XML language are described in a Document Type Definition (DTD). The DVB-HTML DTD is present in Annex AA of the MHP 1.1 specification [2].

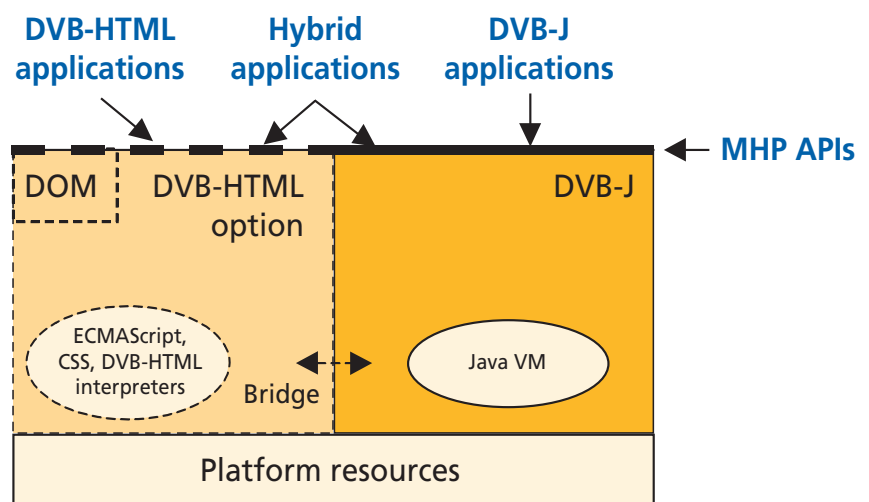


Figure 2
MHP 1.1 basic architecture.

XML defines the rules for checking that a document is “well-formed” (i.e. if it respects the basic XML language constraints) or “valid” (i.e. if it is well-formed and respects the constraints expressed in its associated DTD).

HTML

So as to obey the DVB commercial requirement on rejection of non-conformant content by MHP implementations, DVB-HTML requires the “validation” of any documents that are signalled (in the broadcast stream or in the file itself) as being of the DVB-HTML application type. A DVB-HTML document that violates a constraint expressed in the DVB-HTML DTD is not valid. As such, it will

not be processed and will be rejected by the MHP platform. Some flexibility has been proposed in DVB-HTML on this rule, so as to allow proprietary extensions to the DVB-HTML language but only in a private XML namespace [4] and to allow forward compatibility with future versions of the DVB-HTML language.

4.3. Markup language elements and attributes

The DVB-HTML language complies with the XHTML Modularisation W3C Recommendation [5].

This latter reformulates the W3C HTML 4.01 recommendation [6] into a collection of abstract modules with specific types of functionality. This modularisation work enables the creation of new HTML-based languages by just selecting those modules, offering a functionality pertinent to the target application area.

4.3.1. DVB-HTML selected modules

Table 1 lists all the modules from the XHTML Modularisation Recommendation [5] and defined by DVB, and specifies if they are supported or not in DVB-HTML.

Table 1
Required modules for DVB-HTML.

Module name	Supported
Structure	Yes
Text	Yes
Hypertext	Yes
List	Yes
Applet	No
Presentation	Yes
Edit	No
Bidirectional text	Yes
Basic forms	No
Forms	Yes
Basic tables	Yes
Tables	No
Image	Yes
Client-side image map	Yes
Server-side image map	No

Table 1
Required modules for DVB-HTML.

Object	Yes
Frames	Yes
Target	Yes
Iframe	Yes
Intrinsic events	No
Metainformation	Yes
Scripting	Yes
Style sheet	Yes
Style attribute	Yes
Link	Yes
Base	Yes
Name identification	No
Legacy	No
DVB Intrinsic Events (defined by DVB)	Yes

a) *DVB Intrinsic Events module*

Note that in addition to the W3C XHTML modules, the DVB-HTML language requires the support of a DVB-defined module – the DVB Intrinsic Events module – which provides convenience functionality to content authors for safe event registration.

This new module adds three event handlers on the body and frame elements: `ondvbdomstable`, `onload` and `onunload`. Those handlers catch events which are sent by the MHP implementation, based upon the pre-conditions listed in *Table 2*. The events enable us to refine the lifecycle of DVB-HTML documents and, in particular, the way those events are handled on page transitions.

Table 2
Pre-conditions for MHP implementation to trigger events.

Event	Pre-condition to trigger event
dvbdomstable	When the mark-up structure comprising the DVB-HTML document has been fully received and parsed (the resulting DOM structure is then stable). As a consequence, any DOM structure modifications can be performed “safely” by the calling application.
load	When all the associated resources (images, Xlets, content formats embedded in the object element) of the DVB-HTML document have been received.
unload	When the DVB-HTML document is removed from the window or the frame.

In order to respect namespace policies, this additional module has been scoped within an XML namespace defined by DVB [4]. The prefix necessary when referencing elements and attributes from this namespace is `dvbhtml`. Thus, one can avoid any potential (present or future) conflicts with elements and attributes from the default W3C namespace. An example of prefixing in use is provided below:

```

<!DOCTYPE html PUBLIC "-// DVB//DTD XHTML DVB-HTML 1.0//EN"
"http://www.dvb.org/mhp/dvb/dvbhtml-1-0.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
<script type="text/ecmascript" src="setupEventHandlers.js" />
</head>
<body xmlns:dvbhtml="http://www.dvb.org/mhp"
dvbhtml:ondvbdomstable="setupEventListeners()">
<!-- rest of document here -->
</body>
</html>

```

4.3.2. Semantics restrictions/extensions

The W3C elements and attributes semantics are as defined in HTML 4.01 [6], except in some places where they have been either restricted, modified or extended.

A major area of this work consisted of identifying the list of supported MIME media types in various places of a DVB-HTML document, and specifying the associated semantics.

As a concrete example, the `href` attribute on the anchor element `<a>` has only associated semantics within DVB-HTML when its value (a URI) points to resources with the following MIME media types:

- **text/xml**, **application/xml** (DVB-HTML document or application) or **application/dvbj** (DVB-J application): either DVB-HTML documents within the same application will be presented or new DVB (whether DVB-HTML or DVB-J) applications will be launched according to the MHP application launching mechanisms defined in [1].
- **audio/mpeg** (audio stream): to play an audio stream.
- **multipart/dvb.service** (DVB service): to select a service.

If the value of the URI points to other types of resources, then the behaviour is implementation-dependent.

Semantics have been also specified for some MIME media types referred to by `object` elements, such as MPEG media streams, DVB services, Xlets. This allows us to manage the display of DTV services within a DVB-HTML document.

4.4. Cascaded Style Sheets (CSS)

In order to associate style information with a document, DVB-HTML requires the support of, and is fully compliant with, the W3C CSS Level 2 specification [7]. It respects the conformance clauses as expressed in section 3 of [6]. This, however, does not mean that DVB-HTML supports all properties, data types and rules defined in CSS Level 2. A subset has been defined and extended to cope with the TV environment constraints, with full respect to the specification conformance constraints.

4.4.1. Selectors

Selectors are elements of the CSS syntax. They enable us to define the scope within the DVB-HTML document structure where the style settings will be applied. For conformance reasons with the CSS Level 2 specification,

DVB-HTML requires the support of all CSS2 selectors, whatever their induced complexity may imply. It is thus possible in DVB-HTML to set style to elements identified by their type, their ID value, a specific attribute value, their relative position in the DOM tree (child, sibling, descendant) ... See *Section 4.4.4*.

4.4.2. “dvb-tv” media type

In order for a content provider to target different platforms (PDAs, mobile phones, DTV sets, PCs) and be able to differentiate the TV environment, the “dvb-tv” media type has been created.

DVB-HTML

By decoupling style information from the document structure, content providers can indeed address – without any modifications to the DVB-HTML document – the display specificity of different platforms, by encapsulating in the associated stylesheet, the style settings within those media rules (“screen” for desktop viewing, “dvb-tv” for TV viewing). The “dvb-tv” media type is an extension of the “screen” media type which encompasses all style properties necessary for displaying on a desktop device.

It is up to the implementation to present itself as the most appropriate medium, depending on the context of use. As an example, a desktop device could present itself as a TV device and emulate TV experience and vice-versa.

4.4.3. CSS extensions

CSS extensions have been specified to address TV specificity such as:

- **Opacity management and composition rules:** the `opacity` and `compose-rule` properties have been added in order to perform translucency effects within the graphics plane.
- **Clip-video:** the `clip-video` property enables us to define an arbitrary rectangular-shaped portion of the video as a possible source to the `object` and `image` DVB-HTML elements.
- **Remote control navigation:** navigation properties enable us to manage 4-direction navigation through the remote control.
- **MHP graphics model integration:** the `viewport` rule has been created along with additional related properties in order for a DVB-HTML application to define its initial containing block (size, position), to interrogate the video processing being performed so that the presentation can be tailored to fit, and to be able to manage the sources of the video and background planes. New kinds of selectors are also available. They enable a DVB-HTML application to adapt itself to local conditions such as the current pixel aspect ratio, the resolution, the interlaced or progressive scan nature of the display. If content providers encapsulate their style property settings depending on the values of those selectors, MHP implementations can select the most appropriate ones.

4.4.4. Authoring guidelines

Because DVB-HTML respects the conformance clauses of CSS Level 2, it is supporting functionality which is not necessary to the TV environment. This has in particular some impacts on the overall complexity of the specification. As a concrete example, DVB-HTML supports four positioning schemes: “normal flow”, “floats”, “absolute positioning” and “fixed positioning”. This means that an MHP 1.1-compliant terminal, supporting the DVB-HTML option, is required to implement those four schemes. Using the fixed positioning scheme when translucency effects are used on a document, will require heavy computations that may be rendered only very slowly. As another example, CSS2 conformance requires the support of all CSS2 selectors. Among them, the contextual selectors [7] might affect the overall performance.

Authoring guidelines have thus been provided to warn application authors about the possible impact that the use of some constructs might have.

4.5. Document Object Model (DOM)

DOM is a platform and language neutral interface that provides programmatic access to the content, structure and style of XML or HTML documents, thus allowing us to add dynamicity to a DVB-HTML application.

Support for language bindings in the DOM, for both Java language and ECMAScript, is required in DVB-HTML.

4.5.1. DOM Level 2

Table 3 lists all the DOM modules from the DOM Level 2 set of recommendations [8][9][10][11][12], as well as the additional DOM modules defined by DVB, and specifies if they are supported or not in DVB-HTML.

Table 3
DOM modules supported by DVB-HTML.

DOM Module		Supported
Package	Feature String	
Level 2 Core	Core	Yes
	XML	No
Level 2 HTML	HTML	No
Level 2 Views	Views	Yes
Level 2 StyleSheets	StyleSheets	No
Level 2 CSS StyleSheets	CSS	No
	CSS2	Yes
Level 2 Events	Events	Yes
	UIEvents	Yes
	MutationEvents	Yes
	HTMLEvents	No
	MouseEvents	No
Level 2 Traversal and Range	Traversal	No
	Range	No
DVB-HTML	DVBHTML	Yes
DVB Events	DVBEvents	Yes
DVB Key Events	DVBKeyEvents	Yes
DVB CSS	DVB CSS	Yes
DVB Environment	DVBEnvironment	Yes

4.5.2. DVB-defined DOM extensions

Extensions to the W3C specifications are needed to offer sufficient flexibility to the ECMAScript or Java execution engines, in order to address DVB-HTML specificity such as the use of a particular DTD, the application model, access to the CSS extensions, and the A/V synchronization.

a) *DVB-HTML DOM module*

Since the DOM Level 2 HTML module (which provides access to the HTML structure), as well as the DOM Level 1 specification [13], assume support for the HTML 4.01 DTD, this module is not required in DVB-HTML. It has been replaced by the DVB-HTML DOM module which only exposes some of the DVB-HTML structure, and where the semantics are in line with the modified/extended semantics of the XHTML modules in DVB-HTML (see section 8.5.3 in [2]).

b) *DVB Events DOM module*

This module adds the DVB-defined event interfaces and event types that are necessary for the integration of DVB-HTML within the MHP platform. One new interface:

- **DVBLifecycleEvent**

and some new event types are indeed necessary to cope with the DVB-HTML application lifecycle definition, such as:

- **AppStarting**: application which was prefetched is displayed
- **AppActive**: application enters the active state (it is either displayed or waits for a starting event, if it is prefetched)
- **AppPause**: application is paused
- **AppResume**: application is resumed
- **AppDestroyed**: application is destroyed
- **AppKilled**: application is killed
- **AppTerminating**: application, contained within a sub-frame, is terminated

One new interface is also necessary to manage synchronization with external audio-visual sources:

- **trigger-event**

This new module also contains the additional event types which correspond to the respective event handlers specified in the DVB Intrinsic Events module (see paragraph (a) in *Section 4.3.1.*).

c) *DVB Key Events DOM module*

The DOM Level 2 specifications do not currently specify the event bindings for keyboard and remote control events. This will be provided only when the DOM Level 3 specification is available. Meanwhile, DVB provides its own binding which are defined in this module.

d) *DVB CSS DOM module*

This module has been added in order to gain programmatic access to all the new DVB-defined constructs which extend CSS to the TV environment (see *Section 4.4.3.*)

e) DVB Environment module

It is very convenient for a DVB-HTML application to be able to programmatically access some environment variables such as, for example, the containing window or frame, or some of the DVB-HTML application engine parameters. The W3C DOM set of specifications does not address this area, although current Internet browsers already expose this type of access. DVB-HTML supports this additional module which contains only three interfaces: Window, Navigator and Locator. A DVB-HTML application can thus query the environment through those interfaces.

On the Internet, content is frequently tailored to vendor-specific user agents through requests to the navigator interface. In order to avoid this “verticalised” situation, conformance clauses within DVB-HTML forces a DVB-HTML user agent to be vendor-neutral by returning immutable DVB-defined strings mentioning DVB-HTML and the version number, plus additional identifiers such as the profile number.

A typical string would be: “DVB-HTML/1.0.0 (ib 1.0.0)”, indicating a DVB-HTML 1.0 implementation in the interactive broadcast profile.

5. Integration within the MHP platform

5.1. Application model

One of the first areas of work for the DVB-HTML sub-group was to define what could be a “DVB-HTML application” and its associated lifecycle within the MHP platform. A DVB-HTML application is defined as a set of documents selected from the DVB-HTML family of elements and content formats as described in *Section 4*. The extent of the set is described by an application boundary which is provided within the application signalling information. As long as the end user navigates within the boundary, the DVB-HTML application remains active, and only document transitions occur.

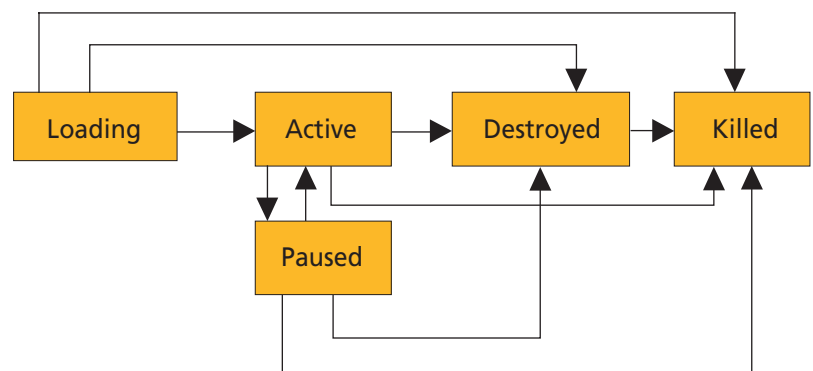


Figure 3
State diagram of the DVB-HTML application lifecycle.

Fig. 3 describes the state diagram of the DVB-HTML application lifecycle.

Those states have the following semantics:

- **Loading**

The DVB-HTML implementation is waiting for documents to be available or is loading them without any rendering actions.

- **Active**

The DVB-HTML implementation has sufficient information in order to render the application. It is either rendering it or is waiting for a trigger event for displaying it. When rendering it, it is implementation-dependent whether user interaction is allowed or not.

- **Paused**

The DVB-HTML implementation is minimizing its use of resources.

- **Destroyed**

This state indicates that the DVB-HTML implementation may no longer be able to access the content resources required to run the DVB-HTML application. The DVB-HTML application may continue to run in this state.

- **Killed**

The DVB-HTML application is terminated.

In addition to those application states, DVB-HTML defines states at the document level in order to refine event propagation semantics on page transition. There are indeed several events sent by the DVB-HTML implementation to the current document, which are either relevant to the application scope or to the document scope:

- **Document events** (dvbdomstable, load and unload)

They inform the current DVB-HTML document about the loading status of its related resources (see paragraph (a) in *Section 4.3.1.*).

- **Application lifecycle events** (AppStarting, AppActive, AppPause, AppResume, AppKilled, AppDestroyed, AppTerminating)

They inform the current DVB-HTML application about an application state transition.

- **Application trigger events** (content provider defined)

To target one document within an application.

The refined model enables us to define the cohabitation of those types of events. It is not described here but the reader is referred to section 9.4 of [2].

5.2. Graphic model

Integration within the MHP graphics model is managed through extensions of the CSS Level 2 structures as mentioned in *Section 4.4.3.*

5.3. Transport

Transport of DVB-HTML resources is ensured through the DVB Object Carousel protocol, as defined in [14] and amended in the MHP specification [1][2]. An additional descriptor has been provided within the File and Directory DSMCC messages which specifies the MIME media type of the described resource. This enables the receiver to operate a filtering operation on manageable resources listed at the directory level, and to avoid loading non necessary objects, thus optimizing the platform resources usage.

5.4. Application signalling

The MHP application signalling enables:

- A receiver to identify applications associated with a DVB service and where to retrieve them and their associated data.
- The broadcaster to manage the lifecycles of applications.

This information is provided through the Application Information Table (AIT) [1][2] which can be considered as an extension of the Program Map Table (PMT). In the case of DVB-HTML, new descriptors have been defined in the application (inner) descriptor loop of the AIT which indicate the physical location of the DVB-HTML application entry point, and the boundaries of the application.



The broadcaster gets the same control over the application as with DVB-J. The application can be autostarted or considered as being present (i.e. not active but launchable through another application), can be killed or destroyed, based on control codes inserted in the signalling. Since a DVB-HTML application can't control its time of presentation, unlike a DVB-J application, the PREFETCH control code has been added to indicate that the DVB-HTML implementation has to preload the DVB-HTML application and wait for an external trigger before presenting it.

5.5. Java relationships

Since DVB-HTML is a presentation format, it is a very good complement to the Java execution environment [15][16][17] which enables access to the TV receiver resources through the MHP Java APIs [1][2]: tuner, persistent storage, return channel, service information, conditional access, object carousel structures, MPEG sections, user settings and preferences, ...



Content authors have a choice between two access ways, either embed the Xlets within their DVB-HTML document (an Xlet is the interface that any DVB-J application needs to implement, in order to be executed on an MHP platform), or access directly the Java constructs from an ECMAScript-to-Java bridge.

5.5.1. Xlet embedding

Embedding an Xlet is roughly the equivalent of embedding an applet but enables us to stick to the Xlet lifecycle and all security aspects built around Xlets permissions. The Xlet and applet interfaces are similar, although not identical. In DVB-HTML, the object element is required for that operation. The semantics of its attribute values have been properly adjusted to fit the MHP model.

5.5.2. ECMAScript-Java bridge

Subject to the MHP security model, this bridge enables public MHP Java packages, classes, methods and fields to be visible from ECMAScript execution contexts, using a property of the ECMAScript `Packages` global object. For example, a DVB-HTML application that would like to access some classes from the `java.lang` package would use, for that purpose, the following ECMAScript construct: `Packages.java.lang`.

5.6. A/V synchronization

One area of commercial importance is certainly synchronized application, i.e. the ability to display an interactive application on top of the main audio-visual stream at a precise point in the media timeline. Advertisements would benefit a lot from that feature but the synchronization needs to be accurate (at the frame-level).

The synchronization for DVB-HTML relies on the same principle as that of a DVB-J application. It is enabled through the use of DSMCC StreamEvent Objects [18]. DSMCC StreamEvent Objects are present in the application file structure (as transported in the Object Carousel). They indicate the presence and location of synchronization events which are themselves transported on dedicated elementary streams and encapsulated in DSMCC stream event descriptors (also referred to by the term “triggers”; see section B.2.4 of [1][2]). Those events contain an ID and a time reference which is based on NPT, which is a frame-accurate media time defined by DSMCC and reconstructed from PCR information present in the MPEG-2 stream [19]. Events can be dissociated from a time reference and be fully asynchronous: in this case, they are referred to “do-it-now” events.

Applications need to register explicitly to those synchronization events. Extension mechanisms have been defined so that the time reference can be the UTC time and the limitation of the payload size of the stream event descriptor (256 bytes) be overcome.

5.7. Content referencing

DVB-HTML makes full use of the locators defined in MHP which reference DVB services, DVB components, DVB events, files and directories from the Object carousel. In addition, a “contextual” form of locator has been defined that enables us to reference:

- the service currently selected by the application;
- the originating service of the application;
- the audio, or the video, or the audio and video being presented on the background device;
- another application within the currently selected service;
- the root directory of the application;
- the icon of the application.

The term “contextual” means that those locators can’t be resolved without any a priori knowledge of contextual information (meaning here, the context of the application within a service). These locators indeed provide short-cut access to a DVB-HTML application which, in the case of DVB-J, are provided through other means.

5.8. Security

DVB-HTML makes full use of the MHP application authentication mechanisms that are independent of the application content formats. The reader is referred to the security section of the MHP specifications (section 12 of [1][2]).

Security was however a major concern when it came to introduce languages such as ECMAScript within MHP. For example, ECMAScript enables the interpretation of strings from arbitrary sources as executable code through

Abbreviations

AIT	Application Information Table	ID	Identification
API	Application Programming Interface	MHP	(DVB) Multimedia Home Platform
CA	Conditional Access	MIME	Multipurpose Internet Mail Extension
CSS	Cascaded Style Sheets	MPEG	Moving Picture Experts Group
DAVIC	Digital Audio-Visual Council	NPT	Normal Play Time
DOM	Document Object Model	PCR	Program Clock Reference
DSMCC	Digital Storage Media Command and Control	PDA	Personal Digital Assistant
DTD	(W3C) Document Type Definition	PMT	(MPEG) Programme Map Table
DTV	Digital Television	URI	Uniform Resource Identifier
DVB	Digital Video Broadcasting	UTC	Universal Time Co-ordinated
DVB-J	DVB - Java	VM	Virtual Machine
ECMA	European Computer Manufacturers Association	XHTML	Extensible Hypertext Markup Language
HTML	Hypertext Markup Language	XML	Extensible Markup Language
		W3C	World Wide Web Consortium
		WWW	World Wide Web

the `eval()` call. In order to overcome that, extensions to the ECMAScript specification [20] have been provided in order to be able to distinguish between internal (aka safe) strings and external (aka unsafe) strings.

6. Conformance considerations

Considering the history record of HTML-based languages with respect to interoperability and since MHP aims at ensuring the emergence of a true horizontal market, the introduction of this optional language can only be successful if a suitable conformance regime is defined. This is perfectly in the spirit of DVB which has already established the conditions for the release of an MHP 1.0.1 test suite. For that purpose, some conformance and interoperability requirements on the DVB-J APIs have been first issued and an overall process for the selection and the acceptance of the official MHP test suite has been put in place within DVB. This process is based on self-certification of implementations and the granting of a logo upon successful completion of the conformance testing based on the official MHP test suite.

A similar process needs to be defined for the DVB-HTML declarative API in order to complement the DVB-J test suite and to certify MHP 1.1 implementations with the DVB-HTML option. This work task is a challenge for DVB since DVB-HTML relies a lot on W3C specifications for which a lot of freedom has been left to implementations.

It is important to note that a DVB-HTML-only implementation can not claim any kind of DVB MHP conformance and that DVB-HTML compliance cannot be dissociated from the overall MHP compliance testing and interoperability rules and procedures. This avoids the threat of market fragmentation and ensures that DVB always offer MHP backwards-compatible profiles. An MHP 1.1 implementation that supports the DVB-HTML option will have to comply with the conformance and interoperability rules defined by DVB.

7. Conclusions

Despite its complexity, DVB-HTML is a powerful application format that will enable a common authoring base for content producers who want to target different embedded devices. Thanks to its adherence to the latest specifications from the W3C, it decouples content from presentation and enables the same content to be tailored to different environments. It also enables us to capitalise on existing web content, through necessary transcoding steps from web documents into DVB-HTML conformant documents.

As shown in this article, DVB-HTML is a language that has been optimized for the MHP platform. This enrichment provides the MHP platform with the necessary support to succeed in the migration towards the Internet. However, DVB is still confronted with the challenge of guaranteeing its basic requirements, such as openness and interoperability, by working on the provision of a robust conformance regime.



Philippe Perrot is currently responsible for the standardization activities within Canal+ Technologies. He holds an engineering degree from the Ecole Nationale Supérieure des Télécommunications in Paris. He joined both Canal+ Technologies and the DVB activities in mid-1999, where he has since been an active member of the MHP specification development activities.

Before joining Canal+ Technologies, Mr Perrot was respectively a visiting scientist at the University of Strathclyde (Glasgow, UK), in charge of signal processing algorithmic studies in the field of non-destructive testing, and a research scientist within the Laboratoire d'Electronique Philips (France). There, he was responsible for several projects around the provision of conversational services (mainly video-telephony) over low or non-guaranteed QoS networks (mobile, IP), and the prototyping of DAVIC-compliant interactive TV digital terminals in the context of the Jasmin project within the French Information SuperHighways program.

Bibliography

- [1] **MHP 1.0.1 specification**, as adopted by the DVB Steering Board, tam232r27, 5th April 2001
http://www.mhp.org/technical_essen/specification.html.
- [2] **MHP 1.1 specification**, as adopted by the DVB Steering Board, tam668r12, 6th June 2001
[http://www.mhp.org/technical_essen/pdf_and_other_files/34\(01\)23.zip](http://www.mhp.org/technical_essen/pdf_and_other_files/34(01)23.zip).
- [3] T. Bray, J. Paoli, C.M. Sperberg-McQueen (Eds): **Extensible Markup Language (XML) 1.0, 2nd Edition**
<http://www.w3.org/TR/2000/REC-xml-20001006>.
- [4] T. Bray and al.: **Namespaces in XML**
<http://www.w3.org/TR/1999/REC-xml-names-19990114/>.
- [5] M. Altheim et. al.: **Modularization of XHTML**
W3C Recommendation, 10th April 2001
<http://www.w3.org/MarkUp/>.
- [6] D. Raggett, A. Le Hors and I. Jacobs: **HTML 4.01 Specification**
W3C Recommendation, 24th April 1998
<http://www.w3.org/TR/html4/>.
- [7] H.W. Lie, B. Bos, C. Lilley and I. Jacobs: **Cascading Style Sheets, level 2**
W3C Recommendation, 12th May 1998
<http://www.w3.org/TR/REC-CSS2/>.
- [8] **Document Object Model DOM Level 2 Core Specification, version 1.0**
W3C Recommendation, 13th November 2000
<http://www.w3.org/DOM/DOMTR>.
- [9] **Document Object Model DOM Level 2 Events Specification, version 1.0**
W3C Recommendation, 13th November 2000
<http://www.w3.org/DOM/DOMTR>.
- [10] **Document Object Model DOM Level 2 Views Specification, version 1.0**
W3C Recommendation, 13th November 2000
<http://www.w3.org/DOM/DOMTR>.
- [11] **Document Object Model DOM Level 2 Style Specification, version 1.0**
W3C Recommendation, 13th November 2000
<http://www.w3.org/DOM/DOMTR>.
- [12] **Document Object Model DOM Level 2 Traversal and Range Specification, version 1.0**
W3C Recommendation, 13th November 2000
<http://www.w3.org/DOM/DOMTR>.
- [13] **Document Object Model DOM Level 1 Specification**
W3C Recommendation, 1st October 1998
<http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001/>.
- [14] EN 301 192 version 1.2.1: **Specifications for data broadcast**
ETSI, January 1999
http://pda.etsi.org/pda/home.asp?wki_id=7555.
- [15] T. Lindholm and F. Yellin: **The Java Virtual Machine Specification**
Addison-Wesley, 1997, ISBN: 0-201-63452-X.
- [16] **Java Platform 1.1 API Specification**, Part of ISBN:1-892488-25-6
<http://www.java.sun.com>.

- [17] J. Gosling, W. Joy and G. Steele: **The Java Language Specification**
ISBN 0-201-63451-1
<ftp://ftp.javasoft.com/docs/specs/langspec-1.0.pdf>.
Clarifications can be found at:
<http://java.sun.com/docs/books/jls/clarify.html>.
- [18] ISO/IEC 13818-6, 1998: **Information technology – Generic coding of moving picture and associated audio information: Extension for Digital Storage Media Command and Control**
<http://www.iso.ch/iso/en/prods-services/catalogue/intstandards/CatalogueListPage.CatalogueList>.
- [19] ISO/IEC 13818-1, 1996: **Information technology – Generic coding of moving picture and associated audio information: Systems**
<http://www.iso.ch/iso/en/prods-services/catalogue/intstandards/CatalogueListPage.CatalogueList>.
- [20] **ECMAScript Language Specification, 3rd edition**
ECMA, December 1999
<ftp://ftp.ecma.ch/ecma-st/Ecma-262.pdf>.
- [21] H.W. Lie and B. Bos: **Cascading Style Sheets, level 1**
W3C Recommendation, 17th December 1996
<http://www.w3.org/TR/REC-CSS1>.
-