HTML5 AS A PLATFORM FOR DELIVERING MOVIE EXTRAS AND INTERACTIVTY

A position paper for the third W3C Web and TV Workshop19-20 September, 2011, Hollywood, USA

Contributors: Jim Helman, MovieLabs

MovieLabs is a technology joint venture of the six major Hollywood studios. Our primary purpose is to foster the development of technologies related to digital distribution of TV and movie content. MovieLabs has been deeply involved in the technical specification of UltraViolet and recently launched EIDR, a unique identifier registry for audio-visual assets.

1 Background

Many years ago, the DVD was designed to give consumers a richer experience than broadcast TV or VHS tapes could offer. It supported menus, chaptering and additional content. Blu-ray discs took this much further offering a full, Java-based execution environment that also added Internet connectivity.

As content consumption moves away from physical media toward digital downloads and streaming, there is wide experimentation with new formats. Apple defined an HTML-based authoring format for iTunes Extras. Streaming movie services have provided bonus material via their HTML-based user interfaces for some titles. And studios have packaged movies with interactive material and extras into "app editions" for tablets and phones. As yet, no standards exist either for authoring or for delivering these experiences. iTunes Extras is to date the only significant attempt to specify a format that wraps up an interactive package with self-contained assets and navigation.

Internet connected platforms such as Blu-ray players, tablets, mobile devices, streaming services, and Facebook provide a rich toolbox for creating experiences that can blow past limitations of a closed DVD-like package, bringing in social networking, reviews and marketing. But the resources required to write custom applications and to author at a high quality for so many different platforms is a barrier to the wider deployment of digital extras, interactive packages and “apps editions” of movies. There is also the issue of longevity of enhanced experiences that go outside of the traditional DVD-like walled garden package, utilizing Internet resources that may not be permanent.

With its new capabilities around media, HTML5 is a strong candidate as a delivery platform for interactivity and digital extras, for both closed and more open connected experiences. But with a wide set of use cases and many complexities around the new HTML5 video features, some digging is in order.

2 Use Modalities

A content platform for delivering interactive digital extras needs to address several different usage modalities that encompass both the delivery mechanism and the nature of the user agent.

There are at least four delivery modes:

- downloaded in a single package,
- downloaded as late bound additions to a package,
- retrieved on a page and/or stream basis from a server over the Internet, or
- retrieved on a page and/or stream basis from another device in the home.

Additionally, the user agent performing the playback can take a number of different forms, including:

- a general browser user agent, such as Safari, Firefox or IE, with or without "plug-in" extensions,
- a media player application that embeds a browser engine, e.g. iTunes, or
- an application created specifically to play back a particular movie, e.g. a "app edition" tablet application.

3 Key Features

3.1 Basic Audio and Video Features

Interactive extras have a set of basic video requirements around video playback that are similar to those found for general video usage in HTML

1) Video start, stop, pause, and seek
2) Audio track selection
3) Subtitle track selection
4) Support for multiple A/V codecs
5) Triggers

At a high level, these seem to be well supported by HTML5's <video>, <audio>, and <track> elements. With regard to subtitling, the format(s) implemented in HTML5 browsers need to support the capabilities of the standards commonly used in TV and movie distribution. Any end-user delivery format needs to enable the easy and mostly lossless transcoding from other commonly used formats.

3.2 Graphics Support

CSS, HTML5 and SVG together provide a very rich basis for moving, layering and animating graphics and video. The graphics models of the DVD and Blu-ray specifications can serve as some reference for requirements here. A detailed mapping of Blu-ray and "app edition" interactive titles to HTML5 can both determine any gaps, as well as inform the generation of profiles for authoring and implementation.

3.3 Adaptive Bitrate

When the video for an interactive package is streamed over the Internet, the requirements are quite similar to those for any other web video application, and that includes adaptive bitrate. As with codecs, many adaptive bitrate solutions exist. If in HTML5 user agents, the video player and its networking support are fixed and built into the browser rather than as pluggable modules, it means that widely utilized adaptive systems need to be supported, or sufficient information and control exposed up to JavaScript to allow the application to directly control stream selection and switching. The movement towards standardizing adaptive bitrate mechanisms into a common framework that supports key existing protocols, e.g. MPEG (DASH), is a valuable step. But even if a harmonized specification were implemented in all user agents, some use cases may still require the HTML/JS application to override or limit the set of stream options provided by the server in the adaptive bitrate manifest file.

3.4 Content Protection

There are many classes of content and content owners. For premium content, many distribution agreements require encryption. Sometimes link-level protection, e.g. HTTPS, is deemed sufficient, at least for closed devices. But many agreements require DRM for premium downloads. When playback of the latter is done in a browser today, typically it uses an <object> or <embed> element, which is pluggable in most desktop user agents. The plug-in can provide both the DRM and APIs to the HTML/JS application.

As to how this might work with HTML5, the two key issues are how a DRM module gets built or loaded into user agents, for both desktop and device platforms, and how interaction between the HTML/JS application and the module is supported The former may be more of a question for browser developers and their partners, but the latter could benefit from the specification of standard APIs for tasks such as querying DRM-related capabilities, e.g. canPlayType(), and handling DRM-related errors.

3.5 Chaptering

Chaptering has been a key feature since the DVD. More broadly, it's mechanism for providing pre-indexed video segments or time positions that a viewer might want to quickly navigate to. This could be done entirely by the surrounding HTML/JS application with video reference points provided as external data. Typically, chaptering is implemented at the application level because the presentation of thumbnails, titles and descriptions is very much part of the look and feel of the experience. The inclusion of chapter names and time reference points in WebVTT tracks addresses some of the simple cases, but complex experiences probably require application code.

3.6 Dealing with Variations in Capabilities and Performance

Capabilities and performance are always increasing, which is wonderful. And content creators keep using whatever they are given, which is also wonderful. But creators get frustrated when authoring for a range of platforms. They don't want to dumb down content to the lowest common denominator, but they also don't want their users to have a bad (or no) experience. And their budgets don't allow them to author and QA a different version for each target device.

Authors of web multimedia content targeted at PCs have dealt with this through a combination of dumbing down content, authoring multiple versions up front or adapting at runtime, and sometimes just writing off some users.

The problem is worse with consumer and TV-connected devices. The range of basic computational performance is wider. Device designers often have to trade off flexibility (e.g., a honking host processor) for cost and power. Some functions are eliminated while others are moved into hardware, but without all of the bells and whistles. Graphics and video architectures are especially variable. And given the full range of HTML5 and SVG capabilities, some features are likely to be out of bounds for some platforms, e.g. multiple simultaneous video streams, real-time convolutions on the video buffer, etc.

There isn't a silver bullet for this.

Profiles, jointly agreed to by content and platform creators, can work in purpose-built devices. But in the case we're discussing, most target devices are unlikely to be built specifically for the purpose of HTML5 interactive video packages.

But a few things can be done to mitigate the problem and make content developers lives better.

- Ensure that user agents are well behaved when platform capabilities are exceeded, e.g. only one video stream supported. When possible, failure behavior and mechanisms for signaling errors should be defined.
- Ensure adequate mechanisms for an application to determine what it's running on. Authors can then at least keep track of what works where and adapt.
- Work to raise the lowest common denominator especially in the areas that matter the most to this set of content authors.

In any event, the first step is to identify the capabilities most needed by authors and which are likely to be an issue on some platforms so that the problem this poses to authors can be mitigated.

3.7 Home Networking

If stored locally on the home network, interactive extras should be accessible across devices in the home via DLNA or other mechanisms.

4 Conclusion

HTML5 meets many of the requirements to be an authoring and/or a delivery platform for interactive extras for movies. There are a number of areas in which necessary features and use cases generate additional work or requirements.

HTML5 Requirements

- Expose metrics to JavaScript to allow support for least some forms of adaptive bitrate at the application level.

User Agent Requirements

- Achieve a common set of adaptive bitrate implementations that supports key existing protocols (http://www.w3.org/Bugs/Public/show_bug.cgi?id=12399)
- Achieve a common set of DRM implementations or a mechanism by which they can be dynamically loaded, similarly to plug-ins.

Authoring Practices

- Define key features, practices and performance metrics to provide both authors and device builders with targets.

Open Questions

- For adaptive bitrate solutions that cannot be implemented in JS, can adaptive bitrate support queries and setup parameters be mapped onto existing HTML5 mechanisms?
- Can DRM support queries, setup parameters and error reporting be mapped onto existing HTML5 mechanisms or are new one required?
- Can DLNA or other home networking standards be used to support the access of interactive extras across the home network?

Two areas are of particular concern, namely adaptive bitrate and DRM support. If each user agent chooses its own adaptive bitrate technologies and its own DRMs to incorporate, each using different, proprietary HTML5/JS extensions to query availability, provide setup parameters and signal errors, it could cause proprietary fragmentation. That would be bad for HTML5 and would dilute its power as a delivery platform for interactive extras. And leaving these key commercial uses in a backwater using <embed> and <object> is completely non-viable as a long-term solution.