OMWeb

Open Media Web

Deliverable N° D3.3

Standardisation Workshop report 3

December 2011

Standardisation Workshop Report 3

Name, title and organisation of the scientific representative of the project's coordinator[1]:

Dr Philipp Hoschka Tel: +33-4-92385077 Fax: +33-4-92385011 E-mail: ph@w3.org

Project website[2] address: http://openmediaweb.eu/

Project	
Grant Agreement number	*248687*
Project acronym:	*OMWeb*
Project title:	*Open Media Web*
Funding Scheme:	*Coordination & Support Action*
Date of latest version of Annex I against which the assessment will be made:	*August 15, 2009*
Deliverable number:	D3.3
Deliverable title	Standardisation Workshop Report 3
Contractual Date of Delivery:	M24
Actual Date of Delivery:	December 5, 2011
Editor (s):	François Daoust
Author (s):	François Daoust
Reviewer (s):	Dr. Philipp Hoschka
Participant(s):	ERCIM/W3C
Work package no.:	3
Work package title:	Standardisation
Work package leader:	François Daoust
Work package participants:	ERCIM/W3C
Distribution:	PU
Version/Revision (Draft/Final):	Version 1
Total N° of pages (including cover):	71
Keywords:	HTML5, Games, Standardisation, W3C

[1] Usually the contact person of the coordinator as specified in Art. 8.1. of the grant agreement

[2] The home page of the website should contain the generic European flag and the FP7 logo which are available in electronic format at the Europa website (logo of the European flag: http://europa.eu/abc/symbols/emblem/index_en.htm ; logo of the 7th FP: http://ec.europa.eu/research/fp7/index_en.cfm?pg=logos). The area of activity of the project should also be mentioned.

DISCLAIMER

This document contains description of the OMWeb project work and findings.

The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated in the creation and publication of this document hold any responsibility for actions that might occur as a result of using its content.

This publication has been produced with the assistance of the European Union. The content of this publication is the sole responsibility of the OMWeb consortium and can in no way be taken to reflect the views of the European Union.

The European Union is established in accordance with the Treaty on European Union (Maastricht). There are currently 27 Member States of the Union. It is based on the European Communities and the member states cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice and the Court of Auditors. (http://europa.eu.int/)

OMWeb is a project partly funded by the European Union.

TABLE OF CONTENTS

1 SUMMARY

This document reports on the Workshop on HTML.next for Games organized by Open Media Web in Warsaw in September 2011, following right after onGameStart, the first HTML5 Game conference.

HTML5 enables the development of games that run across devices, and are both easy to deploy and easy to maintain. Several features that are not yet part of the Web platform would be directly useful for games development, though. The workshop was the occasion to engage with the games community and to start listing features of interest. Workshop participants (including people from Bocoup, Google, Mozilla, RIM, Tecnalia, Wooga) were passionate about games and Web technologies. During the workshop, more than 20 features that would enable the development of better games using regular Web technologies were reviewed, refined and classified:

- 12 new features were identified, such as the need for a **Joystick API**, a **mouse lock** mechanism, an **orientation lock** mechanism, or **high performance timers**
- Standardization has already started for 5 features such as **accurate sound triggering** or **real-time peer-to-peer communications**.
- A few other features mentioned require more discussion, or were seen as out of scope for standardization in W3C.

This report describes the main use case for each of the features of interest and includes a short gap analysis of today's (end of November 2011) Web platform from the point of view of game development. Where applicable, the W3C working group and links to possible draft proposals are mentioned.

To ensure that games community needs are known and properly addressed in W3C, a **Games Community Group** was proposed and created at the end of the workshop. This group is dedicated to tracking the implementation of Open Web Platform features directly relevant for games development, and communicating how to build games on the Open Web Platform to the general public. A community group is a W3C discussion forum open to anyone, without fees, particularly well suited to serve as coordination point for a particular community within W3C.To engage more people in the activities of the group, a W3C Games Community Group Summit was organized by Bocoup next to the New Game Conference in San Francisco, on Thursday 3 November from 10AM to 1PM, hosted by Zynga.

2 INTRODUCTION

HTML5 enables the development of richer and ever more interactive applications. As far as gaming is concerned, the situation evolved from a world of hacks and plug-ins to a situation where one can envision the production of a multi-player first-person-shooter game using W3C standards such as canvas, Web Workers, Web Sockets, on-going standardization efforts on audio, real-time communications and various other Web and device APIs.

However rich this new Web platform may become, games require fine-grained control over:

- **Devices used for interaction**: keyboard, mouse, joystick, touch screens, gestures, cameras, etc.
- **Rendering surfaces**: ability to go fullscreen, to take screenshots, 3D APIs, audio synthesis, etc.
- **Processing power**: multi-threading, memory management, use of hardware acceleration, etc.

Fine-grained control is difficult to achieve today, either because the right API is missing, or because the specification does not address the exact needs, or simply because the specification is not clear enough to prevent interoperability issues. The Web as a games development platform is still in its infancy. The goal of this workshop was to meet and engage in discussions with developers of the games community as early as possible, taking advantage of onGameStart, the first HTML5 Game conference, and gather inputs for HTML.next, the next version of HTML, that would enable the development of fully immersive games using regular Web technologies.

Please see the call for participation[3] for details.

This document reports on the proceedings of the workshop[4]. Moreover, it takes into account further discussions at the W3C Games Community Group Summit organized by Bocoup in Zynga's office in San Francisco early November 2011[5]. The list of participants is provided in Appendix 1. Slides presented at the workshop are listed in Appendix 2.

[3] http://openmediaweb.eu/2011/09/14/workshop-on-html-next-for-gaming/
[4] http://www.w3.org/2011/09/games/
[5] https://docs.google.com/document/pub?id=1fs1hpZvP05ViEWtaLSmNQUV_PW2jCWS5Oe2GAdBKgl0

3 WEB GAMES

3.1 Existing Web Games

Today, support for 3D rendering using WebGL and other advanced features in Web browsers is still in development. Web games that take advantage of such functionalities (see Figure 1) are prototypes or proof-of-concept for the most part.

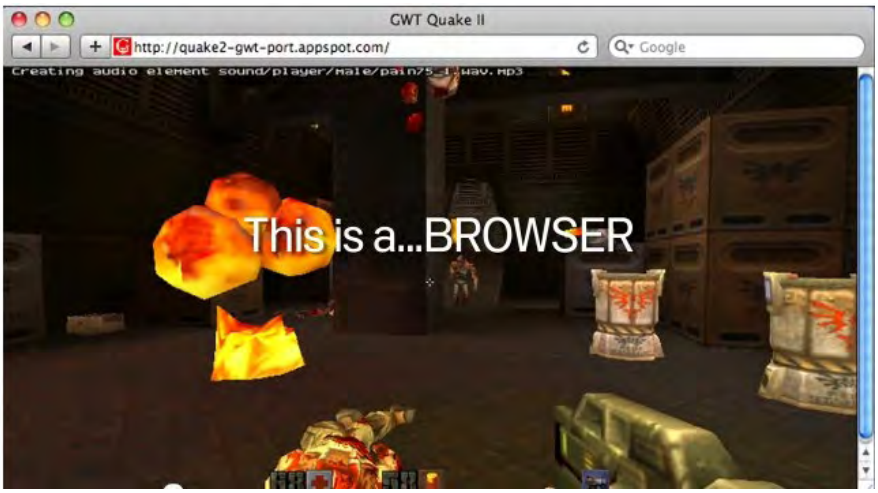


Figure 1 – HTML5 prototype of Quake 2 by Google (uses WebGL, WebSockets, Web Storage)

Today, for technical reasons, **casual games** are the main focus for commercial-grade games developed with Web technologies. This category includes puzzle games (e.g. Memory, Minesweeper), card games (e.g. Poker games), as well as 2D arcade games such as Angry Birds (ported to HTML5 by Rovio, illustrated in Figure 2).

Figure 2 – Screenshot of the HTML5 version of the popular Angry Birds by Rovio.

As their names suggest, casual games are typically played for short periods of time, often on the go. Thus, content aggregators that are specialized in Web games often focus on mobile platforms, building on the ubiquitous presence of the Web on terminals (see Figure 3).

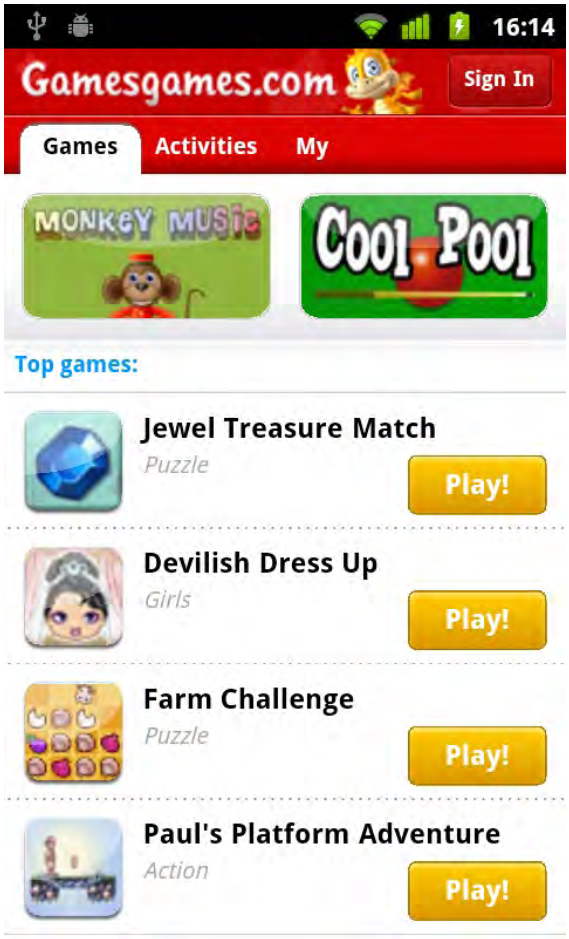


Figure 3 – Screenshot of the "GamesGames" games portal by Spilgames on an Android device

3.2 Web Games Development

On computers, traditional applications are **event-driven**: they respond to user input. In the absence of any user input, such applications are idle. For instance, a word processor reacts to key presses and mouse movements.

Games are usually different because they continue to be active in the absence of user interactions: enemies keep on moving, collisions still need to be checked, sounds may have to be played and graphics updated. To account for this need, games are typically designed around a **core "endless" loop** that embeds the game's logic and is repeated up until the game is over.

As opposed to what happens in an event-driven approach, the core loop in a game must synchronize the internal state of the game with events that may have occurred between two runs of the loop. Some of the features discussed during the workshop (e.g. Keyboard Capture, or Joystick API) include proposals to optimize that synchronization process, moving away from the traditional event-driven approach in today's Web browsers.

In Web browsers, the core loop may be implemented in JavaScript using repeated calls to the DOM function *window.requestAnimationFrame* when supported, or *setTimeout* as a fallback. As opposed to *setTimeout*, *requestAnimationFrame* is optimized by Web browsers for animations, giving the control back to the Web application when the browser is ready to update the screen. It is the preferred way to implement games core loops in HTML5.

Apart from the core loop, games follow the same design patterns as most other applications. To *immerse* users in the game, games often need fine-grained control over the available input methods (mouse, joystick, keyboard, etc.). Games also need to propose as good a visual experience as possible, using hardware accelerated rendering (through the Graphical Processor Unit or GPU) when available.

It is no surprise that most of the gaps identified during the workshop affect either user interaction or graphical user interfaces and associated computation needs.

4 REPORT ON THE PROCEEDINGS

4.1 Presentations

Several workshop participants presented their thoughts during the workshop. This section presents a summary of points raised during these talks.

Figure 4 – Workshop participants

4.1.1 François Daoust (W3C)

François gave an introduction to standardization at W3C and ways for the games community to engage in the development of the Open Web Platform, e.g. through the creation of a Community Group dedicated to games.

He further highlighted the list of groups in W3C that are of particular interest from a games perspective:

- **HTML WG**
  HTML5 spec, then HTML.next spec
  http://www.w3.org/html/wg/
  public-html@w3.org
- **CSS WG**
  CSS Level 3 (~40 specs), incl. CSS animations, CSS transformations, etc.
  http://www.w3.org/Style/CSS/
  www-style@w3.org
- **Web Applications WG (WebApps)**
  CORS, DOM Level 3 Events, File API, Indexed Database, Selectors API, XMLHttpRequest, Web Storage, Web Sockets, Widgets.

http://www.w3.org/2008/webapps/
public-webapps@w3.org

- **Device APIs WG (DAP)**
  Battery, Calendar, Contacts, Device Discovery, Gallery, Media Capture, System Info, etc.
  http://www.w3.org/2009/dap/
  public-device-apis@w3.org
- **Web Performance WG (WebPerf)**
  Efficient Script Yielding, Navigation Timing API, Page Visibility, requestAnimationFrame, setImmediate
  http://www.w3.org/2010/webperf/
  public-web-perf@w3.org
- **Web Real-Time Communications WG (WebRTC)**
  Peer-to-peer connection, audio/video/data (Joint work with IETF)
  http://www.w3.org/2011/4/webrtc/
  public-webrtc@w3.org
- **Audio WG**
  Audio processing and synthesis API
  http://www.w3.org/2011/audio/
  public-audio@w3.org
- **Geolocation WG**
  Geolocation API, DeviceOrientation Event Spec
  http://www.w3.org/2008/geolocation/
  public-geolocation@w3.org
- **Points of Interest WG (POI)**
  Points of Interest for Augmented Reality
  http://www.w3.org/2010/POI/
  public-poiwg@w3.org
- **Web Events WG**
  Touch Events
  Planned: Mouse Lock API Joystick API
  http://www.w3.org/2010/webevents/
  public-webevents@w3.org
- **SVG WG**
  Scalable Vector Graphics
  http://www.w3.org/Graphics/SVG/
  www-svg@w3.org
- **WebFonts WG**
  Good looking fonts on the Web
  http://www.w3.org/Fonts/WG/
  public-webfonts-wg@w3.org
- **Planned: Web Application Security WG**
  Security and policy mechanisms, secure cross-site communication
  http://www.w3.org/2011/webappsec/
  public-webappsec

Other groups, not listed here, may of course be of interest for games developers depending on the game under consideration. Groups home pages link to the current status of the deliverables produced by the group as well as to the list of participants, on-going issues and other news.

4.1.2 Seth Ladd (Google)

Seth reviewed priority topics for games development. Topics highlighted include:

- Mouse Lock API
- Keyboard capture to enable polling for keys pressed
- Joystick API, event-based vs. poll-based to read analog inputs
- Occlusion queries and texture compression format for WebGL
- Web application packaging, ways to store huge number of assets locally
- Monetization is a key topic, although out of scope for W3C. Business models will emerge
- Audio support
- Fullscreen API
- Surround sound support
- « Pass by reference » in Web Workers
- Vector support in JavaScript

4.1.3 Darius Kazemi (Bocoup)

See What game devs need from HTML5 set of slides in Appendix 2.

Darius said that the number one problem for Web games development is accurate sound (re)triggering, noting that the Web Audio API[6], under discussion within the W3C Web Audio Working Group, is close to perfect for games developers. According to him, the main need is to play a sound right now, synchronized with an animation. Darius cited the FMOD library as a good set of functionalities that covers 99% of games developers needs.

Darius raised other features needed such as a Mouse Lock API, asset loading techniques (the game “Fieldrunners” has about 150MB of content, of which 30MB need to be loaded at a time) and feature detection for auto-configuration purpose.

He added that there are a number of features and areas that are already good. For instance, networking issues are not restricted to Web technologies. They are more a technique problem than a technology problem. Web game developers could learn a lot from traditional game developers in that area. Web Workers are also great for games development. Finally, Darius asserted that code protection is a false problem: obfuscation techniques greatly help, but there is no way to protect data in the end. Business models will evolve to cope with it, he concluded.

[6] https://dvcs.w3.org/hg/audio/raw-file/tip/webaudio/specification.html

4.1.4 Laurent Hasson (RIM)

Laurent said that, while screen resolution is easy to extract, there is no way to tell the pixel density of a given screen, an increasing problem on mobile devices.

According to him, another area for discussion is how to map keys used to control games on devices that do not have physical keyboard. It is impossible to tell the keyboard layout from a Web page, and there may not be any keyboard in the end. Laurent proposed an API to register the keys used within a game so that the Web browser may propose to arrange these keys on a touch screen for instance. This would perhaps be better solved with best practices to handle keyboard events.

Figure 5 – Laurent Hasson (background) exchanges with Rob Hawkes (foreground)

Laurent wondered about the status of the Web on game consoles, where Web browsers are not yet primary tools that users use. Laurent explained that it is important that the Web experience moves to game consoles for the convergence to succeed.

Finally, Laurent proposed different new CSS properties (randomization parameters, higher-level semantics).

4.1.5 Rob Hawkes (Mozilla)

Rob mentioned Project Paladin in Mozilla, which is set to create the best gaming technology available for the open Web. The project includes Mouse Lock and Joystick APIs. Another project named Electrolysis aims at separating processes that display the browser UI, Web content and plugins.

Mozilla Research is heavily looking at games from various angles, including monetization and Web application packaging. Documentation is updated all the way. Contributions are welcome, Rob concluded.

4.2 The Web as a Games Development Platform

Discussions at the workshop covered a number of technical topics directly relevant to Web games development and standardization. Topics were sorted according to their current standardization status:

- New feature?
- Already being addressed?
- Out of scope for W3C?
- No standardization needed?

The list of topics is not meant to be exhaustive or authoritative. It is rather meant to serve as an initial set of features that the games community may want to track and support within W3C. The order of topics within each section has no particular significance.

Figure 6 – Live editing the list of topics raised by workshop participants

4.2.1 New features

Features in this category are missing from the Web platform as of November 2011. They are directly relevant to games development and do not have a home in W3C yet. The group that seems most relevant is mentioned whenever possible.

Some of these features may not need to be standardized in the end based on feedback, priorities, or e.g. because they cannot be enabled securely in a Web environment.

Out of the twelve gaps identified:

- Five are around **user inputs** and **user interaction**: Mouse lock, Keyboard lock, Keyboard capture, Joystick API, and Orientation lock.

- Four affect **outputs**, i.e. **graphics and audio rendering**: Higher-level semantics for CSS animations, CSS extensions, Surround sound support and WebGL in Web workers).
- Two target **device capabilities**: Hardware feature detection, Access to screen pixel density
- The last gap is around **precise time measurement** within the core loop of a game: High performance timers

4.2.1.1 User Input and Interaction

Mouse Lock

On desktop computers, a user navigates in a first person shooter (FPS) game with the keyboard to walk around and uses the mouse to aim and shoot.

In the context of a Web browser, the mouse cursor is always on and mouse events that get fired on a DOM element stop as soon as the mouse cursor leaves the element. In particular, from the point of view of a Web application, the mouse "disappears" as soon as it reaches the borders of the Web browser window. To use the mouse to control movements that should not be constrained in space, there should be a way to "lock" the mouse cursor to a single DOM element so that all mouse events are sent to that element (e.g. a sprite representing a "weapon") rather than to whatever DOM element the mouse cursor is currently "hovering over".

The need for a mouse locking mechanism extends beyond games. For instance, this feature is also useful to lock the mouse wheel to zooming in/out on a map rendered on the screen.

The Mouse Lock specification draft[7], proposed by Vincent Scheib (Google), describes a possible solution. Locking the mouse has privacy and security implications since it enables a Web application to take over one of the main interaction methods available to the user to control his computer.

The Web Events Working Group proposed to revise its charter to include Mouse Lock in its list of deliverables. The re-chartering process is on-going as of November 2011.

Keyboard Lock

Many keyboard shortcuts are used to control window behavior in practice, making it easy to accidentally close a browser window that runs a Web game, e.g. trying to crouch and walk forward in the game at the same time. There is currently no way for a Web application to receive notifications for keyboard combinations usually assigned to the underlying system.

An API similar to the Mouse Lock draft to lock the keyboard and ensure the Web application is in control of all possible keyboard combinations. As for mouse lock, locking the keyboard has privacy and security implications and needs to remain under the control of the user.

[7] http://dvcs.w3.org/hg/webevents/raw-file/default/mouse-lock.html

The Web Events Working Group would provide the appropriate forum to discuss this feature.

Keyboard Capture

In a shoot 'em up game, the user may press keys continuously to move the ship around and fire weapons non stop. During each execution of the core loop of the game, the game engine needs to track which keys are pressed to perform the right action.

On the Web today, key presses are event-based. Web applications receive one event when a key is pressed and one event when it is released. Games are usually built around a core loop repeated over and over again. From one loop to the other, games need to maintain a mapping of keys currently pressed and handle events separately. For efficiency and convenience, the current set of pressed keys should rather be exposed as a property of the window object so that game engines can easily poll the status of the keyboard during each loop.

The HTML Working Group would provide the appropriate forum for this feature.

Joystick API

All game consoles ship with a game controller that features buttons, one or more digital sticks and/or one or more analog sticks. In a flight simulator game, an analog stick gives users precise control over the movements of the aircraft ailerons and elevators, thus replicating the behavior of real aircraft commands. In a first person shooter game, the analog stick is used to control the movements of the character's head and/or weapon.

The DOM Level 3 Events Specification[8] introduces the `DOM_KEY_LOCATION_JOYSTICK` constant that makes it possible to detect whether buttons and digital keys got pressed on a game controller. However, analog sticks positions and movements cannot be adequately covered by existing events.

Buttons in game controllers may also include pressure sensors that are currently not exposed by the DOM.

[8] http://www.w3.org/TR/DOM-Level-3-Events/

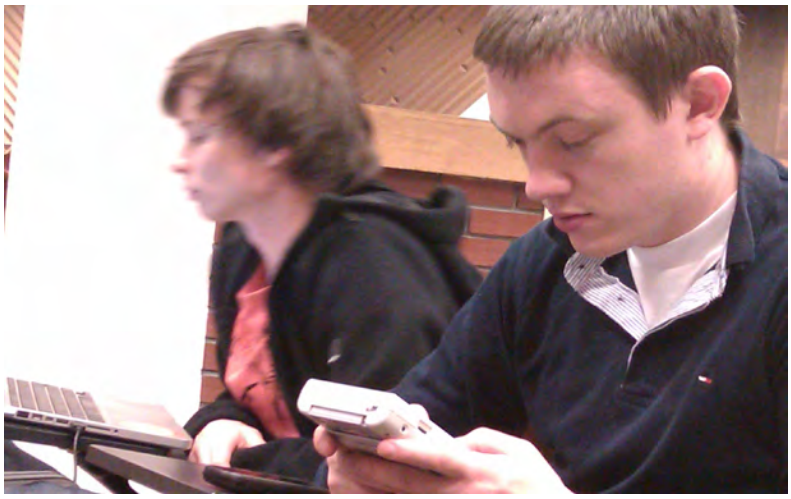


Figure 7 – Michal Budzynski (onGameStart organizer) plays with the gamepad of an authentic Gameboy

The Joystick draft specification[9] proposed by Scott Graham (Google) proposes to switch from an event-based mechanism to a poll-based mechanism that is more natural for games and analog measures. Games typically iterate over a main loop (through calls to requestAnimationFrame) to update game components based on user actions, and thus based on user input readings. An event-based mechanism attached to an analog stick would either fire more events than the game engine may handle or fewer events which would negatively affect the precision of the control.

The JoystickAPI draft specification[10] proposed by Ted Mielczarek, Jason Orendorff, David Humphrey, Kevin Gadd (Mozilla) takes a similar approach but also exposes joystick events.

The GamePad draft specification[11] attempts to merge both proposals.

The Web Events Working Group proposed to revise its charter to include the Gamepad API in its list of deliverables. The re-chartering process is on-going as of November 2011.

Orientation lock

While it is good practice to provide a flexible user interface that adapts itself to the size and orientation of the screen, it is not always doable or even a good idea. For instance, video content playing on many smartphones requires the user to switch to landscape mode. Many games have similar needs. On mobile devices, for action games for instance, it is easy to accidentally switch the orientation without actually meaning to, leading to a broken user experience.

While it is easy to detect the orientation of a screen in CSS and scripts, there is no way to constrain the user to a given mode (portrait or landscape).

[9] http://www.h4ck3r.net/joystick/joystick.html
[10] https://wiki.mozilla.org/JoystickAPI
[11] http://dvcs.w3.org/hg/webevents/raw-file/default/gamepad.html

There are no known draft proposals as of today. The basic approach would be to define something like a `window.lockOrientation()` function to lock the viewport orientation and `window.unlockOrientation()` to unlock it. The user would probably need to be made aware that the orientation is locked since that is not an expected behavior in a Web browser.

The Web Applications Working Group would probably be the right group to discuss this functionality.

4.2.1.2 Graphics and Audio Rendering

Higher-level semantics for CSS animations

Menu elements and/or user notifications in games are very often animated with effects such as bouncing, wobbling, glowing/fading, or rotating. These effects catch the eye and highly contribute to improving the user experience. Such animations are but a first step in the right direction. Since the human eye is very good at spotting variations in movements, adding some level of randomness to animations helps ensure that the player does not feel that the game is too repetitive.

The need for easy-to-achieve high-end visual effects extends well beyond games to include any Web application that wants to improve the visual user experience.

Game menus and user interfaces may be achieved using regular HTML and CSS to some extent. CSS specifications such as CSS Transitions, CSS Animations, CSS 2D Transforms and CSS 3D Transforms greatly enrich the tool set of effects game developers may rely on to define rich user interfaces.

However, seemingly simple effects such as the ones mentioned above (e.g. wobbling or glowing) cannot easily be achieved through CSS. More complex animations require scripting. Scripting is not a good long-term solution for purely visual effects as the Web browser is in a much better position to optimize the use of resources and CPU to perfom such visual effects.

Scripting is also required to introduce any kind of randomization parameter to animations. The open source Alice.js micro JavaScript library[12] (A Lightweight Independent CSS Engine), created by Laurent Hasson and co-developed with Jim Ing (RIM), explores the use of hardware-accelerated capabilities to generate high-end visual effects. The syntax used to describe these effects remains as close as possible to CSS on purpose. The library could typically serve to explore new effects and highlight potential candidates for new CSS properties.

A randomization parameter can be added to most effects in Alice.js to achieve natural randomness. Discussions on these additions should be brought to the CSS Working Group.

[12] http://blackberry.github.com/Alice/

CSS Extensions: Constants, Parameters, Mix-ins, Inheritance

Games interfaces are highly sensitive to the screen dimensions and resolution. Games often adapt sizes of the various visual game assets and adjust CSS property values to take advantage of each pixel available. Object oriented techniques also greatly help reuse, ensuring the resulting CSS remains as minimal as possible.

CSS does not allow to declare constants, parameters, mix-ins, or simple inheritance. The LESS JavaScript library[13] extends CSS with dynamic behavior such as variables, mix-ins, operations, functions and nested rules. A good solution needs to retain the simplicity of CSS.

Discussions on these additions should be brought to the CSS Working Group.

Surround sound support

As 5.1 channel loudspeaker arrangements become common in homes, games that feature a virtual world use surround sound effects to increase the realism of the action taking place in the scene.

Developments within the Audio Working Group should help create a basis for audio synthesizing using JavaScript which is not restricted to any particular speaker setting. Higher-level functionality to place a sound in a surround environment may be needed.

The Web Audio API draft proposal[14] by Google provides the basis mentioned above. The Audio Data API[15] by Mozilla is more basic but compatible with other uses of media streams, e.g. to enable real-time communications. The Web Audio API could be added on top of the Audio Data API. Further higher-level functionalities might help generate surround sounds in real-time from JavaScript.

Such discussions are in scope of the Audio Working Group.

WebGL in Web Workers

Games that use WebGL often need to run CPU-intensive complex transformations. Such computations block the main context, leading to potentially unresponsive user interfaces. Running the computation in a Web Workers would allow not to block the main context.

In addition to transferable objects, using immutable arrays in postMessage call sent to Web Workers would de facto provide a means to expose the WebGL context to a Web Worker.

The Web Applications Working Group, responsible for the Web Workers specification, is the appropriate forum to bring this feedback.

[13] http://lesscss.org/
[14] https://dvcs.w3.org/hg/audio/raw-file/tip/webaudio/specification.html
[15] https://wiki.mozilla.org/Audio_Data_API

4.2.1.3 Device Capabilities

Hardware feature detection

Games need to balance high quality graphics and sounds against the need to maintain smooth animations for an acceptable game play. This trade off requires adjusting settings based on the performances of the machine (enabling/disabling anti aliasing, switching to higher/lower texture quality, increasing/reducing the depth of the horizon in 3D space rendering, etc.). These settings can be switched on/off manually through a settings page. However, most users do not understand performance implications of switching a particular setting on/off. More importantly, for most users, games should just work. As much as possible, settings adjustments should thus be done on an automated basis.

A Web application cannot easily and automatically tell the performances of the CPU/GPU of the machine it runs on. It cannot access the hardware capabilities of the graphics card either. Such information should be exposed to Web applications so that they can take decisions on behalf of the user. At a minimum, a hardware performance score should be exposed so that Web applications can take decisions on settings. The score would be representative of the CPU and GPU performances and features. Such scores already exist, e.g. 3DMark[16].

Exposing a score (or more) creates a privacy issue though since it enables fingerprinting: a Web site would be able to generate a unique client identifier partly based on the information exposed by the mark. The `Do-not-track` HTTP header currently discussed in the Tracking Protection Working Group could perhaps be extended to protect users against such possibilities. More discussion is required.

Such functionality seems to fall in scope of the Web Performances Working Group.

Access to screen pixel density

The visual quality of the user interface is crucial in games. Whenever possible, higher-quality graphics are used to improve the visual experience of a game on screens that have a higher pixel density.

CSS defines CSS pixels. Viewports are typically set to match the definition of CSS pixels. Modern smartphones ship with higher density screens (sometimes coined as retina displays). For a Web application, the screen dimensions in CSS pixels are easy to tell. The actual pixel density is not, however. Browser implementations tend to report incorrect values for `window.devicePixelRatio`.

To figure out the density of pixels of the screen, the User-Agent HTTP header may be used in conjunction with a device description repository. Support for the resolution property of the CSS Media Queries specification also allows to narrow down the resolution of the screen automatically with a script. Both methods are indirect and inefficient, though.

[16] http://www.3dmark.com/

There are no known draft proposals as of today. Exposing the pixel density may pose a privacy concern since it exposes another identifier that could be used for fingerprinting purpose. However, to some extent, this identifier is already exposed through the methods mentioned above.

Since the goal is to expose a device feature, the Device APIs Working Group seems to be the right target for further discussions.

4.2.1.4 Precise Time Measurement

High performance timers

In platform games, accurate movements are important to ensure a good gameplay. This requires timer accuracy in the below-millisecond range. Today, events that are triggered in reaction to the user input expose a `timeStamp` property which specifies the time at which the event was created in milliseconds relative to `1970-01-01T00:00:00Z`. There is no way to attach a more precise high-resolution timer to an event.

The Web Performances Working Group would probably provide the right forum for further discussions.

4.2.2 On-going Standardisation: Features

Features in this category are missing from the Web platform as of September 2011 but are being addressed by a W3C Working Group. The games community should keep an eye on progress in these fields, in particular because solutions proposed by these working groups may not address all of the requirements identified for games development.

4.2.2.1 Improvements to Web Workers: « Pass by reference »

To retain a responsive UI and ensure the gameplay is not negatively impacted by delays, game engines delegate I/O intensive tasks and heavy computing to dedicated threads, while the core game engine remains available to react to user inputs. Delegated tasks may involve passing large chunks of memory between threads, for instance, such as an internal representation of the virtual world depicted in the game.

Web Workers allow Web application authors to spawn background processes running scripts in parallel to the main Web applications. Data that flows between Web Workers and the main thread can only be passed by copy, though. This approach is inefficient for large amounts of memory.

Web Workers would need to be extended to allow parameters to be passed by reference. While the feature may seem easy to add, it opens the door to race conditions and other non trivial synchronization issues. An alternative approach is to use a *transferable object*: the object gets passed from the Web Worker to the main thread and cannot be used in the Web Worker anymore once transferred.

The Web Applications Working Group discusses support for transferable objects in the context of Web Workers.

4.2.2.2 Accurate sound triggering

In a shoot 'em up video game, enemies “explode” when they are hit by a weapon. The explosion sound needs to be synchronized with the animation on the screen. Multiple explosions may occur at once.

The <audio> element in HTML5 was introduced to play audio. It is a good step in the right direction but the initial use cases did not include multi-channel low-latency and high-accuracy audio.

The Audio Working Group was created to address such use cases. The Web Audio API[17] and the Audio Data API[18] serve as starting points for the group.

4.2.2.3 Asset loading and smart caching solution

A racing game contains megabytes of textures and other assets to render a realistic 3D world. On the Web, downloading megabytes of data is not easily doable, e.g. if the user is on a 3G connection. To ensure that the game starts to run as fast as possible, such game assets should be stored locally and available offline.

Permissions required by the game also need to be declared in some format.

Finally, when many assets need to be retrieved, the game needs to know when it can move offline (i.e. when all the assets are locally available)

The File System API[19] is a step in the right direction. The HTML5 application cache mechanism[20] enables offline Web applications as well. W3C widgets[21] provide a useful packaging format. There is still fragmentation in this space.

The HTML Working Group and the Web Applications Working Group are discussing next steps. The W3C Workshop on the Future of Off-line Web Applications[22] held early November 2011 discussed related problems.

[17] https://dvcs.w3.org/hg/audio/raw-file/tip/webaudio/specification.html
[18] https://wiki.mozilla.org/Audio_Data_API
[19] http://www.w3.org/TR/file-system-api/
[20] http://www.w3.org/TR/html5/offline.html#offline
[21] http://www.w3.org/TR/widgets/
[22] http://www.w3.org/2011/web-apps-ws/

4.2.2.4 Real-time peer-to-peer communications

A multi-player game requires exchanges of messages among peers, in particular to exchange players coordinates and actions. In cooperative mode, players in a first person shooter game set up an audio communication channel to synchronize their actions.

The WebSocket API[23] (and protocol) enable bi-directional communication between a client and a server. The server needs to centralize all messages sent to players in a multi-player game. This is suboptimal as the number of players increases. Direct peer-to-peer connections would reduce latency and improve scalability.

The Web Real-Time Communications Working Group develops the WebRTC API while the RTCWEB group in IETF discusses the underlying protocols. The solution will enable audio/video calls as well as data exchanges between peers.

4.2.2.5 Fullscreen API

Most games run fullscreen to take advantage of the space available and to ensure players are not distracted by other elements on the screen that would not match the game's color scheme and ambiance.

While most Web browsers allow the user to switch to fullscreen mode (typically through the F11 key on desktop browsers), a Web application cannot automatically switch to fullscreen by itself for obvious security reasons.

The CSS Working Group is to develop the fullscreen API as well as add fullscreen detection to CSS Media Queries.

4.2.3 Features that require more discussion

Features in this category may or may not be seen as directly relevant features or high priority features for games. More discussions are needed to refine the scope of the desired functionality.

4.2.3.1 Screenshot API

At the end of a race, a car simulation game proposes to replay the heroic moments of the race. The user can stop the replay at anytime to take a screenshot and share it with other players or friends.

To ensure that players are not cheating e.g. using additional script libraries that could boost the performances of their character in a game, a multi-player game may take and upload regular screenshots to analyze the user's screen on the server and detect cheaters.

[23] http://www.w3.org/TR/websockets/

Today, a Web application cannot take a screenshot of a browser viewport.

The feature is already highlighted in the HTML.next input page of the HTML Working Group. The feature would obviously trigger privacy issues that need to be addressed.

4.2.3.2 Efficient JSON compression mechanism

A multi-player game needs to send regular messages between players to synchronize players' views. These messages often take the form of simple JSON structures. Keeping the messages terse greatly improves the latency required to keep the game in sync.

While content may be zipped upon transfer, typical compression/decompression protocols on the Web (gzip, deflate) are not efficient for small low-latency messages. It should be possible to define more efficient compression techniques for such use cases, for instance building on the Efficient XML Interchange (EXI) Format[24] for XML-based documents.

Better quantification of the actual need and envisioned gain would be a good idea to assess whether such an efficient compression mechanism would be useful in the context of games. This may also be a transfer protocol issue, which is not directly in scope for W3C.

4.2.3.3 Advanced canvas rendering capabilities

Sprite animations in a shoot 'em up game involves re-drawing parts of the underlying <canvas> element in a short amount of time. Pulling state information (e.g. pixel bounds) from canvas graphic entities without having to memorize it in JS on each draw command would improve efficiency.

Available canvas drawing methods are not highly efficient for the animation of sprites. A few JavaScript libraries make use of the “dirty rectangle” rendering technique, whereby only the portions of the canvas that have changed are rendered on a new rendering pass. Native support for this technique in a browser could improve efficiency.

It is unclear at this stage what functionality needs to be added. The efficiency it would bring also needs to be quantified. The Games Community Group will refine the need and review possible solutions, to be submitted to the HTML Working Group.

4.2.4 Discarded features

Other features were raised during the workshop and discarded, either because they are not in scope for W3C or because they simply do not seem worth standardizing.

[24] http://www.w3.org/TR/exi/

4.2.4.1 Monetization

Selling Web applications on the Web is challenging. Ads may not be particularly well suited in games and/or on mobile devices either. Workshop participants believe that new business models will emerge as Web games get developed. While micro-payments might be a topic of interest for standardization, the question of monetization is seen as largely out of scope for W3C.

4.2.4.2 Collision detection

There are roughly as many collision detection mechanisms as there are games that need to detect collisions between objects. There are no clear solution that would work in most cases and could be included in a standard.

4.2.4.3 Vector support in JavaScript

Computations in 3D worlds require manipulating vectors. Adding support for matrices operations in JavaScript and parallel computation on all available cores would ease developments. This is a Javascript language feature though, so more in scope for ECMA.

4.2.4.4 WebGL APIs

Similarly, new APIs for WebGL could ease games development and improve rendering quality, such as enabling occlusion queries (to tell which objects are behind or on top of others), developing a texture compression format, or enabling WebGL to run computation on the Graphics Processing Unit (GPU). Such considerations are in scope for the Khronos Group.

4.3 Next Steps

From workshop discussions, there is little doubt that HTML5 and upcoming Web standards enable, or will enable, the development of great games using regular Web standards technologies. However, there are also a number of features that the games community would like to see added to the Web platform and/or addressed as high priority items within W3C.

The use cases highlighted in this report should be brought back to the relevant working groups in W3C to ensure that the solution developed by these working groups address the needs expressed:

- For new features, the games community should assess the interest among key actors of the community, draft initial proposals and push for standardization within W3C (typically by submitting the draft proposal to an existing working group, or by proposing the creation of a new working group if needed).

- For features that are currently being standardized, the games community should send use cases and requirements that address games needs to the relevant working group. It should also track progress in the working group, help find resources to push the work forward, and send comments on technical solutions as early as possible in the process to ensure that the needs are properly addressed.
- The games community should refine needs for other features.

Within W3C, Community Groups were created to provide the right setting for such goals. They provide an open forum, without fees, where Web developers and other stakeholders develop specifications, hold discussions and connect with W3C working groups. A quick show of hands during the wrap-up session revealed that most workshop participants were in favor of creating a games community group. The **W3C Games Community Group**, submitted by Boaz Sender (Bocoup), is a direct outcome of the workshop.

The Games Community Group is to serve as an entry point for the games community within W3C, tracking standardization efforts that are relevant to games development and communicating how to build games on the Open Web Platform to the general public. The group will also help coordinate discussions on a proposed feature among games stakeholders and build concrete proposals that may then be fed back into working groups.

The creation of the community group is but a first step towards ensuring that the Web platform becomes a real games development platform. Next steps identified include:

- Refine the scope of the group, set rules of operation, identify chairs and people willing to take an active role
- Document goals and features under discussion on the group's Wiki
- Invite games developers to join the new community group
- Start reaching out to the groups identified above

5 WORKSHOP IMPACT

The **W3C Games Community Group**[25] was created as a direct outcome of the Warsaw workshop. As of November 2011, the group has already 50 participants. Companies represented in the group include Bocoup, Google, Ideateca (EU), Mozilla, Nokia (EU), Opera (EU), Spilgames (EU), Tecnalia (EU), Walt Disney, Wooga (EU), and Zynga.

This group serves as a central hub where developers from the games community who are not yet familiar with W3C can share their needs, check on standardization progress, learn more about W3C and engage in standardization activities.

The group held a first summit next to the New Game conference in San Francisco on 3 November 2011 where it clarified its scope and reviewed the technical topics that had been discussed in Warsaw.

Figure 8 – W3C Games Community Summit participants

The group has already identified a number of areas where it can contribute use cases to other W3C working groups. It will track specifications and vendor implementations and recommend new specifications to be produced as necessary.

[25] http://www.w3.org/community/games/

6 CONCLUSION

With the advent of HTML5 and its companion APIs, game developers envision the production of immersive games based on Web technologies that run on a wide range of devices. From a standardization perspective, feedback from the games developers community is extremely valuable: games raise **use cases and needs** that may not yet be addressed by W3C working groups and help highlight **performance and interoperability issues**.

The Open Media Web project organized a workshop on games next to the first HTML5 Games conference in Warsaw in September 2011 to invite the games community to engage in standardization activities at W3C. The workshop led to the launch of the **W3C Games Community Group**, proposed at the end of the workshop. The creation of the group was met with enthusiasm beyond participants of the workshop, attracting about 50 participants from various companies in less than 2 months. European companies are well represented in this group through Ideateca, Nokia, Opera, Spilgames, Tecnalia, Wooga

The W3C Games Community held its first summit in San Francisco, hosted by Zynga, on 3 November 2011 to refine its scope and review technical topics that had been raised during the workshop. The group will refine these topics in the upcoming months and engage in discussions with W3C Working Groups responsible for them or propose the creation of new W3C Working Groups if needed.

APPENDIX 1: LIST OF PARTICIPANTS

28 people attended the workshop, representing 14 companies, including Google, Mozilla, Tecnalia, and Wooga. Here is the list of participants with affiliations, sorted by last name:

- Garret Alfert, 5 Apps
- F. Javier Almeida, Tecnalia
- Vlado Banda, Interactive1
- Michal Budzynski, Virtual Design
- Piotr Cichosz, Britenet
- Alexander Conceiro, Tecnalia
- François Daoust, W3C
- Jan Filipowski, GameBoxed.com
- Marie-Claire Forgue, W3C
- Laurent Hasson, RIM
- Rob Hawkes, Mozilla
- Eduardo Ibanez, Tecnalia
- Krystian Jarmicki
- Darius Kazemi, Bocoup
- Sebastian Kippe, 5 Apps
- Petteri Koivumaki
- Seth Ladd, Google
- Sinisa Robert Loncaric, Kaliani Studio
- Lukasz Mankowski, SII
- Andrzej Mazur, Britenet
- Raul Otaolea, Tecnalia
- Maciej Partyka
- Rudy Pons
- Joceba Rodriguez, Tecnalia
- Boaz Sender, Bocoup
- Jakub Siemiatkowski, Virtual Design
- Kamil Trebunia, Wooga
- Jill-Jenn Vie, student at ENS Cachan

APPENDIX 2: WORKSHOP PROCEEDINGS

Slides presented at the workshop are included as appendix below:

Workshop on HTML.next for Games

Speaker
François Daoust <fd@w3.org>
World Wide Web Consortium (W3C)

This presentation
http://www.w3.org/2011/Talks/0924-html5-games-fd/

Location
onGameStart - First HTML5 Game Conference
Warsaw, Poland

Date
24 September 2011

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n°248687 - Open Media Web (OMWeb)

Outline

Photo by Elaine Vallet

- **W3C, HTML5 and Games**
  Why we are here…
- **HTML5? HTML.next?**
  What we're talking about. W3C groups to keep an eye on.
- **What *you* can do about it**
  Send comments, submit tests, raise new scenarios
  … and/or take the lead e.g. through a Community Group
- **Today's discussions**
  Structure, scope, outcome

Part 1:
W3C, HTML5 and Games

W3C: Shaping the Web of the future

- **Web Standards**
  (X)HTML, CSS, XML, SVG, PNG, XSLT, WCAG, RDF, ...
- **Consortium**
  330 members, from industry and research
- **World-wide**
  Offices in many countries, including Brazil, China, India, Morocco, South Africa, ...
- **One Web!**
  Founded and directed by inventor of the Web, Tim Berners-Lee
- **Global participation**
  32,000 people subscribed to mailing lists, 1,500+ participants in 60+ Groups

Er, right, W3C is not exactly into games development...

HTML5?

It's one big specification developed by the W3C HTML Working Group:

- Audio and Video on the Web at last!
- Rich Web applications
- Canvas
- new APIs
- …

But it's more than that. What we usually mean is:

- A long list of specs (incl. CSS and various APIs), more than 100!
- The next Open Web Platform
- A cool development platform for all sorts of crazy innovative stuff

Hmm, now we're getting closer to games!

Games

- Games push the limits of the platform
- Games mix specs together (e.g. visual effects made of video, CSS, JavaScript)
- Helps to nail down tricky interoperability issues and underspecified functionalities
- Games have specific needs not fully taken into account yet
- Games raise new issues, e.g. around performance and portability
- Passionate and people, lots of innovations
- Very important to hear from games community!
- *One Web* must include games

And, of course, games are fun!

Part 2:
HTML5? HTML.next?

The Next Open Web Platform

- HTML5
- CSS 2.1
- CSS 3 Selectors
- CSS 3 Media Queries
- CSS 3 Text
- CSS 3 Backgrounds and Borders
- CSS 3 Colors
- CSS 3 2D Transformations
- CSSOM View Module
- CSS 3 Transitions
- CSS 3 Animations
- CSS 3 Multi-Columns
- CSS Namespaces
- SVG 1.1
- WAI-ARIA 1.0
- MathML 2.0
- ECMAScript 5
- 2D Context
- WebGL
- Web Storage
- Indexed Database
- Web Workers
- Web Sockets Protocol/API
- Web Real-Time Communications Protocols/API
- Timing Control for Script-Based Animations
- Geolocation
- Navigation Timing
- Progress Events
- Element Traversal
- DOM Level 3 Events
- Media Fragments
- XMLHttpRequest
- Selectors API
- CSSOM View Module
- File API
- RDFa
- Microdata
- Widget Packaging and XML Configuration
- WOFF
- HTML Media Capture
- Contacts
- HTTP 1.1 part 1 to part 7
- TLS 1.2 (updated)
- IRI (updated)
- …

Legend:
spec: developed within W3C
spec: developed outside of W3C (IETF, Khronos Group, ECMA)

The <video> tag in HTML5

A regular HTML tag

```
<video src="myMovie" id="vid" />
```

A standard API

```
var vid = document.getElementById("vid");
vid.play();
vid.pause();
vid.currentTime = 0;
```

Interaction with CSS

```
video {
 border-radius: 2em;
 translate(100px, -100px) skewY(30deg) scale(0.5,0.5);
 opacity: 0.5;
}
```

Video and Canvas

The HTML5 <canvas> tag allows for direct manipulation of video data:

- Dump video frame to a <canvas> tag
- Analyse video frame as an array of pixels with JavaScript
- React consequently

Muppet says: **(nothing)**

Video and Vector Graphics

- W3C Standard released in 2001 (SVG 1.2 released in 2009)
- SVG is included in HTML5
- Complete freedom to redesign user interfaces
- Possibility to clip and filter video at will

CSS Transitions

CSS Transitions describe how CSS properties change smoothly from one value to another over a given duration.

Example (CSS transitions must be supported):

Home
About
Feedback
Help
Contact

CSS Media Queries

```
<link rel="stylesheet" type="text/css" href="mobile.css" />
<link rel="stylesheet" type="text/css" href="screen.css"
      media="only all and (min-width: 600px)" />
```

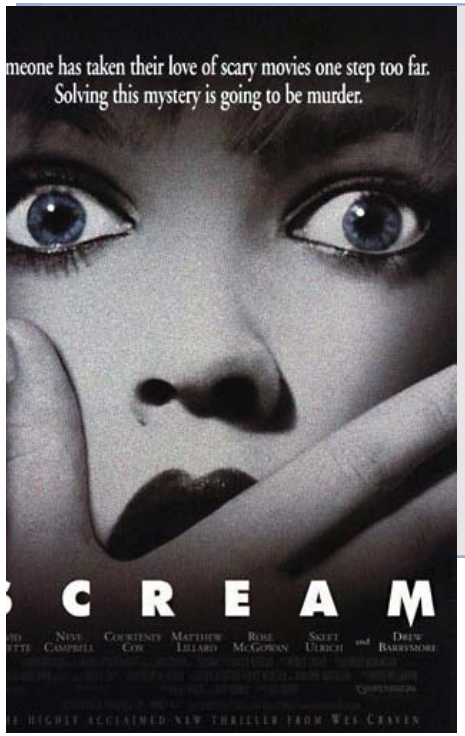
Web Sockets

```
var toruk = new WebSocket('ws://example.com/avatar/toruk');

toruk.onmessage = function (event) {
  if (event.data !== "grmpf") {
    alert("Argh, not good.");
  }
};

toruk.send("Attack!");
toruk.send("Calm down!");
```

Web Real-Time Communications




```
var local = new PeerConnection(
 'TURN sidney.scream.example.net',
 sendSignalingChannel);

// Start sending audio/video
local.addStream(aLocalStream);

// Send signaling message to the other side
function sendSignalingChannel (message) {
 …
}

// Signaling message received from other side
function receiveSignalingChannel (message) {
 local.signalingChannel(message);
}
```

The WebRTC API will include a data channel to send data in real-time (unreliable and/or reliable?)

Web Workers for multi-threading

```
var agentSmith = new Worker('clone.js');

agentSmith.onmessage = function (event) {
  alert("One more!");
};
```

requestAnimationFrame

```
function animate(time) {
 … // Do some animation
 window.requestAnimationFrame(animate);
}

window.requestAnimationFrame(animate);
```

Offline Web Applications

```
<html manifest="kungfu">
 <head>
  <link rel="widget" href="kungfu.wgt">
```

W3C Workshop on the future of Offline Web Applications
5 November 2011, Redwood City, CA, USA
http://www.w3.org/2011/web-app-ws/

Device APIs

HTML.next

- Too late to put new features in HTML5 spec
- Recent work started may already be seen as HTML.next
- Lots of things done in parallel. Modularity is good.
- Priorities?

Basic question, is: ***What could the Web do for you?***
Its corollary is: ***What can you do to ensure that happens?***

Part 3:
What *you* can do about it

Keep an eye on Working Groups of interest for games

- **HTML WG**
  HTML5 spec, then HTML.next spec
  http://www.w3.org/html/wg/
  public-html@w3.org (archives)

- **CSS WG**
  CSS Level 3 (~40 specs), incl. CSS animations, CSS transformations, etc.
  http://www.w3.org/Style/CSS/
  www-style@w3.org (archives)

- **Web Applications WG (WebApps)**
  CORS, DOM Level 3 Events, File API, Indexed Database, Selectors API, `XMLHttpRequest`, Web Storage, Web Sockets, Widgets, Mouse Lock API?
  http://www.w3.org/2008/webapps/
  public-webapps@w3.org (archives)

- **Device APIs WG (DAP)**
  Battery, Calendar, Contacts, Device Discovery, Gallery, Media Capture, System Info, etc.
  http://www.w3.org/2009/dap/
  public-device-apis@w3.org (archives)

- **Web Performance WG (WebPerf)**
  Efficient Script Yielding, Navigation Timing API, Page Visibility, `requestAnimationFrame`, `setImmediate`
  http://www.w3.org/2010/webperf/
  public-web-perf@w3.org (archives)

- **Web Real-Time Communications WG (WebRTC)**
  Peer-to-peer connection, audio/video/data (Joint work with IETF)
  http://www.w3.org/2011/4/webrtc/
  public-webrtc@w3.org (archives)

- **Audio WG**
  Audio processing and synthesis API
  http://www.w3.org/2011/audio/
  public-audio@w3.org (archives)

- **Geolocation WG**
  Geolocation API, DeviceOrientation Event Spec
  http://www.w3.org/2008/geolocation/
  public-geolocation@w3.org (archives)

- **Points of Interest WG (POI)**
  Points of Interest for Augmented Reality
  http://www.w3.org/2010/POI/
  public-poiwg@w3.org (archives)

- **Web Events WG**
  Touch Events, Joystick API?
  http://www.w3.org/2010/webevents/
  public-webevents@w3.org (archives)

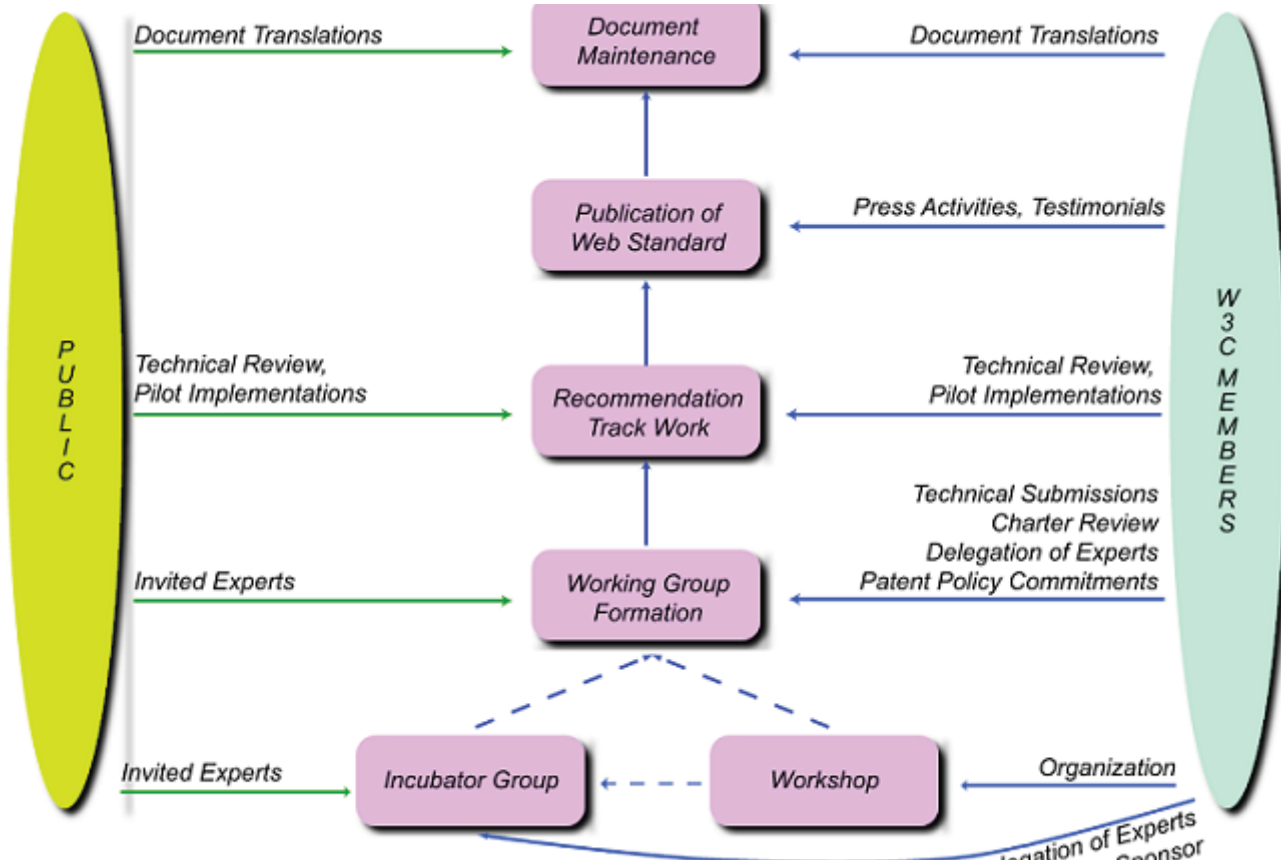
- **SVG WG**
  Scalable Vector Graphics
  http://www.w3.org/Graphics/SVG/
  www-svg@w3.org (archives)

- **WebFonts WG**
  Good looking fonts on the Web
  http://www.w3.org/Fonts/WG/
  public-webfonts-wg@w3.org (archives)

- **Web Application Security WG?**
  security and policy mechanisms, secure cross-site communication
  http://www.w3.org/2011/webappsec/
  public-webappsec (archives)

- **Other groups?**
  Any other group with a direct link to games?

Contribute to W3C



Other ways to contribute

- Send comments on spec
- Submit test cases
- Raise new scenarios
- Push for things through blog posts, talks, media, etc.
- Translate standards
- Wear HTML5 t-shirts
- …

Community Groups

http://www.w3.org/community/

- Open public forum
- Without fees
- Self-determined (no process constraints)
- Without time limit
- IPR balanced (lightweight protection during development)
- Tuned for transition to standards-track, although not mandatory.

Community groups are perfect to build communities (surprising, isn't it?) around a topic

Part 4:
Today's discussions

What we're looking for today

- Game inputs on standards (new or existing features):
  - **Use case**
    - A scenario that cannot be done at all
    - A scenario that cannot be done *efficiently enough*
    - A scenario that can be done provided things don't change
  - **Requirements**
    - What is missing?
    - Gap analysis
  - **Possible solution**
    - Simple piece of code that could work?
    - Draft specification somewhere?
  - **Possible hiccups**
    - Things that probably need to be investigated
    - (portability, accessibility, complexity, privacy, security, integration with other specs)
  - **Related Working Group**
    - The group that is responsible for this functionality, if any
- Interoperability problems?
- Need to work on best practices for games development?
- What next?

Possible next steps

What should we do next?

- Nothing?
- Provide feedback to identified groups?
- Continue discussions in New Game conference to engage more actors?
- *Gaming Community Group?*

If Community Group, what scope?

- Entry point for games community within W3C
- Track progress of games-related features in groups
- Identify missing features
- Write periodic *The Web as a games development platform* reports
- Develop initial technical specs
- Develop best practices?
- Develop tools?
- Develop test cases?

Thanks

François Daoust <fd@w3.org>
World Wide Web Consortium

http://www.w3.org/2011/Talks/0924-html5-games-fd/

24 September 2011, Warsaw, Poland
onGameStart

Follow the **Open Media Web** project:

- Web site: http://openmediaweb.eu/
- RSS feed: http://openmediaweb.eu/feed/
- Twitter: **@w3c_omweb**

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n°248687 - Open Media Web (OMWeb)

What game devs need from HTML5

Darius Kazemi

Bocoup

Bocoup, LLC

Where I’m coming from

- Unique position
  - traditional game developer for about 7 years
    - Lord of the Rings Online, Dungeons & Dragons Online, LEGO Universe, Carmen Sandiego Facebook, Oregon Trail Facebook; many many many canceled titles
  - Understand the culture of traditional game dev and what prevents them from doing work
  - Board of Directors, International Game Developers Association (http://www.igda.org)

Bocoup, LLC

Audio

- Accurate sound (re)triggering!
- We all know that <audio> tag can't guarantee that a sound effect will play at the same time as an animation, especially when retriggering the sound
- Web Audio API is pretty close to perfect for game dev
  - Store all audio in memory buffers
  - Not limited to audio defined in the DOM

Bocoup, LLC

Audio

- Look at fmod
- Basically industry standard middleware
- If you 100% ***copy fmod functionality***, 99% of game devs will be happy!

Bocoup, LLC

Mouse Lock

- Mouse lock for 3D games.
  - This is unacceptable
- open proposals exist: http://bit.ly/MouseLock
- Make this standard, or certain types of important 3D cameras can never be implemented in HTML5, forcing us to use plugins

Bocoup, LLC

Asset Loading

- We need asset loading and smart caching solution
- For example, Fieldrunners is ~150MB of content loaded ~30MB at a time
- File System API is a step in the right direction

Bocoup, LLC 

Feature Detection

- Absolutely necessary for auto-configuration
  - (if I called it “responsive games” would that be more “webby”?)
- Already have basic feature detection
  - Would like to know GPU, CPU, hardware info; not just browser feature detection
  - Sigh, fingerprinting, I know
    - Consent-based: “Click here to give us information about your computer so we can give you the best experience”

Bocoup, LLC

Stuff that's okay

- Networking
  - Big secret: networking tech sucks EVERYWHERE, including Xbox/PS3/Wii
  - Less a technology problem, more a technique problem (web devs could learn a lot from traditional game devs here)
- Web Workers
- Joystick/Gamepad events
  - (not critical, but coming down the line!)

Bocoup, LLC

A problem even though it shouldn’t be

- Most common Q: “How do I protect my game code/assets?”
- Most common A: “Minification, authentication with servers, blah blah blah”

Bocoup, LLC

A problem even though it shouldn't be

- Most common Q: "How do I protect my game code/assets?"
- Most common A: "Minification, authentication with servers, blah blah blah"
- Actual A: "Wake up! You've **never** been able to protect your game code/assets! If I can run it on a machine I administer, I can hack it. That's life."

Bocoup, LLC

Not W3C’s problem

- We need better tools!
- Game discovery
- Monetization
- Distribution

Bocoup, LLC