Innovative R&D by NTT

Requirements for Embedded Browsers needed by Web-based Signage

IRC: #websigns

Kiyoshi Tanaka, Shigeru Fujimura

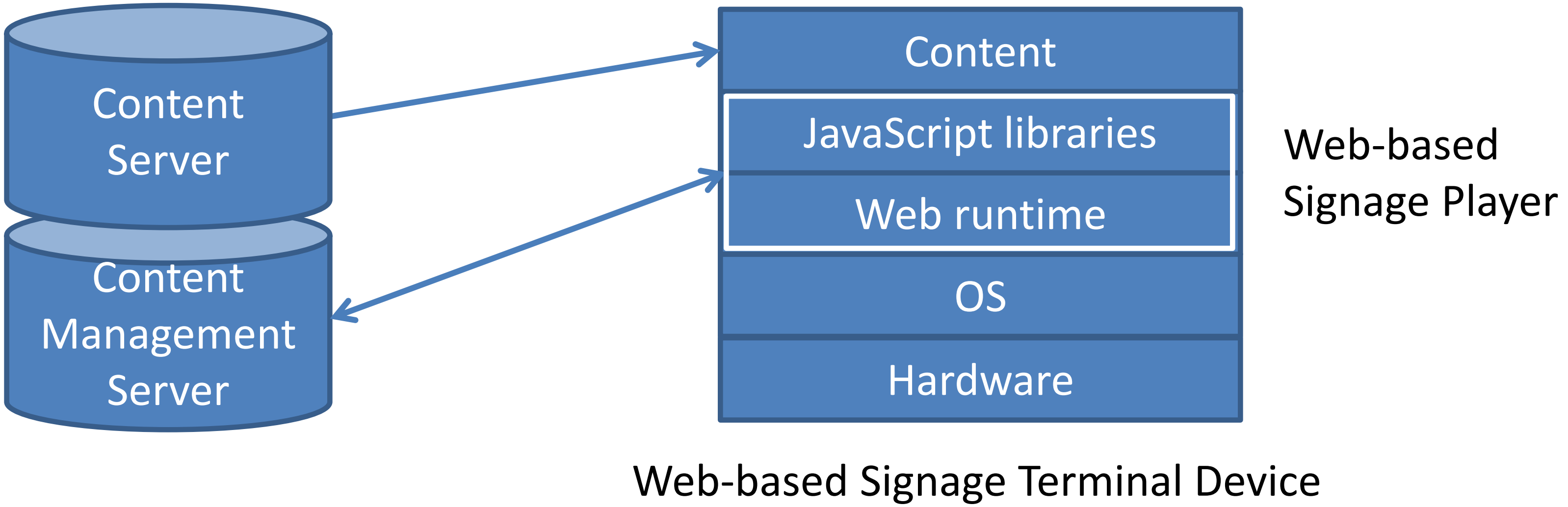
28 Oct 2015

Outline of this session

- Presentation
  - What is the web-based signage?
  - Embedded browser for the web-based signage and its requirements (required APIs)
- Discussion
  - Which is the proper WGs to discuss each API?
- Wrap-up
  - Identifying the newly required standardized items

What is the web-based signage?

- The digital signage ecosystem using Open Web Technologies (discussed in Web-based Signage BG).
- BG defines Web-based Signage Player as a set of Web runtime and JavaScript libraries.
- Currently, embedded browsers such as in TV sets can be used as Web-based Signage Player.



Embedded browser for web-based signage

	Embedded browser for web-based signage	Embedded browser for consumer devices	Traditional browser
Installation	Embedded in a factory		Allowed to install by a user
HW resource	Limited (CPU, Memory, Input device, …)		Easy to expand
Operation	Not operated by audiences but operated indirectly by a remote owner so far	Operated by a user who views contents directly	
Number of users at once	More than one audiences	Single user	

- APIs for remote operation or auto operation are needed at least for the embedded browsers (different environment model from traditional browser).
- Much more attractive for a variety of audiences (multi-screen, high quality video)

API idea for Signage Browser

- Auto Pilot API
  - Enable to control the underlying browser directly by a remote owner
- Rich Presentation API
  - Enrich the presentation of the web contents to the audiences

Where is the proper WGs to discuss standardization for each requirement?

API	Relevant WG
Auto Pilot API	
Power Management API	
Web NTP Client API	
System Context	System Applications WG?
System Events	System Applications WG?
Rich presentation API	
Multiple resources for video	HTML WG (Media TF)?
Multicast video playback	HTML WG (Media TF)?
Access control of external storage	
Double Buffering API	

Power Management API

- Enable web applications to control the browser behavior.
  - For some unknown trouble or emergency case
    - Restarting the underlying web runtime
    - Rebooting the underlying operating system
  - For the system operation
    - Shutting down the underlying operating system
    - Turning off the terminal device

Web NTP Client API

- Enable web application to get a precise time stamp interacting with the specified NTP server.
- The time stamp object provides the same functions as the Date object specified in the ECMAScript.

System Context

- Consider if we need system-wide background process
  - start working from the boot time, independent to loaded contents
  - workers which can interact with any windows (context/domains)
  - context with privilege to access any windows and origins
  - handle System Events (see next item)

Note: This is similar context/concept to Chrome process of Firefox or System Apps of Firefox OS.

System Events

- Consider to support system type events such as:
    - Boot - to run specific task or to load specific contents on boot
    - Window Closed (including Crash) - to reload, re-schedule, sync
    - Watchdog (busy more than xx ms) - to reload, close or change behavior

Note: Some of these can be defined as events of the System Context or additional events of Service Workers or within performance API.

Multiple resources for video

- Define container element or system which provides multiple video source and to control or gives hints to the user agent about which video resource to use, based on the screen pixel density, viewport size (especially for 2K/4K/8K), and other factors. i.e. <video> version of <picture> element.
- Multiple resource for iframe, basically same as above, <picture> like feature for <iframe>.

Multicast video playback

- Define a capability to playback multicast video streams in order to save network bandwidth to deliver video content to multiple signage terminals.
- For synchronization of the video playback between terminals, MMT technology might help synchronization(additional enhancement).

Access control of external storage

- Define an access control of external storage devices such as USB memory.

Double Buffering API

- Enable web application to make following resource (both pages and embedded media elements) requests send to the buffer
  - Rendering them on background
  - and then Switching them to foreground at the timing set by the application

Where is the proper WGs to discuss standardization for each requirement?

（Result of the breakout session）

API	Relevant WG
Auto Pilot API	
Power Management API	No WG
Web NTP Client API	(Decided after discussion jointly with Multi-device Timing CG)
System Context	(Decided after discussion in Web-based Signage BG)
System Events	SysApps WG for Service worker issue
Rich presentation API	
Multiple resources for video	Media TF (inputted after use-case discussion in Web-based Signage BG)
Multicast video playback	
Access control of external storage	(Decided after discussion in Web-based Signage BG)
Double Buffering API	SysApps WG for Service worker issue