CANVAS
CANVAS AND VIDEO BINDING

CUES
DIGGING DEEP INTO CUES

TRACK META
META DATA TRACK USAGE

OVERLAYS
PLACING CONTROLS OVER

TIMELINE
CONTROLLING TIME

SUBTITLES
LEVERAGING SUBTITLES
WHAT IS INTERACTION
FOR WHAT PURPOSE?

• Supplemental information on demand – hotspots, Ajax info.
• Enhanced navigation possibilities – menu, bookmarks, thumbnailing.
• Validating knowledge retention – inline quizzes.
• V-commerce – item in context viewing.
• Improved story telling opportunities.
• More engaging user experiences.
• Incorporation of social networking.
CROSS ORIGIN ERRORS

In the examples provided multiple files will be used including text track files. These files must be served from the same origin as the video media files to prevent browser cross origin errors.
HTML5 Video Element

- Added in HTML5 to mainstream media video playback with out plugins.
- Built in controls available.
- Multiple sources to provide various browser support.
- Fallback messaging allowed for unsupported browsers.

```html
<video controls>
  <source src="/video/big_buck_bunny_480p_surround.avi" type='video/mp4'>
  <source src="/video/big_buck_bunny_480p_stereo.ogg" type='video/ogg'>
  <p>HTML5 Video is not supported by this browser.</p>
</video>
```
DIGGING DEEP
GOING TO THE EDGE

HTML5 Video structures are still young.
As we dig deeper this will be more and more evident.
Present inconsistencies will be noted.
Browsers of course have various degrees of support as we go deeper.

VIDEO
TRACKS
CUES

WATCH THAT LAST STEP…
OVERLAYING ELEMENTS ON TOP OF VIDEO
OVERLAYING HTML ELEMENTS

BASIC ELEMENTS

• Leverage CSS position to place elements.
• Canvas an option as well.
• Provides for interactions
  – While video loading.
  – During playback or timeline navigation.
  – When play ends.
• Can overlay entire video element or partial.
EXAMPLE: HOTSPOTS
Overlaying hotspots on a video.
TIMELINE
SYNCHRONIZING MANUAL EVENTS
**VIDEO API EVENTS & PROPERTIES**

**SMALL BUT KEY SUBSET**

- **Key events**

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>play</td>
<td>Video is played.</td>
</tr>
<tr>
<td>pause</td>
<td>Video playback is paused.</td>
</tr>
<tr>
<td>ended</td>
<td>Video playback has ended.</td>
</tr>
<tr>
<td>timeupdate</td>
<td>Video currentTime is updated.</td>
</tr>
</tbody>
</table>

- **Properties Include**
  - `currentTime` – read/write of the playing pointer on the video media.
TImeLine Method / Events
A Sample Flow

- Methods and current time fired automatically or manually.
TIMEUPDATE EVENT
SYNCHRONIZING WITH THE TIMELINE

- `timeupdate` fired when `currentTime` updated.
- By definition;
  - Continually fired while media playing.
  - Current media position adjusted.
  - Frequency not specified or guaranteed.
- In practice, events fires several times per second.

```javascript
video.addEventListener('timeupdate', function() {
    console.log("currentTime = "+ video.currentTime);
}, false);
```
EXAMPLE: TIMELINE CONTROL
A DVD style content menu controlling current time and using overlay actions.
< TRACK >
TEXT TRACKS IN PARALLEL
TRACK ELEMENT OVERVIEW
SUBELEMENT TO MEDIA

- Track elements are added under the video element.
- Can have multiple tracks for a single video element.

```html
<video controls>
  <source id='mp4' src="/video/trailer.mp4" type='video/mp4'>
  <track kind="subtitles" label="English" src="subtitles_en.vtt" srclang="en" default>
  <track kind="subtitles" label="Italiano" src="subtitles_it.vtt" srclang="it">
    This browser does not support HTML5 video.
</video>
```

- Tracks accessible by element id attribute or textTracks array.

```javascript
var allTracks = document.getElementById("sampleVideo").textTracks;
```
TRACK ELEMENT
ELEMENT TAG

• Track element attributes
  – Kind           The kind of track included.
  – Src            The source file of the track.
  – Srclang        Two letter designation for language.
  – Label          User readable label for the text track.
  – Default        Default text track designation for use.

• Can be referenced with the id attribute
• Multiple tracks with different kind attributes allowed.
• [link](http://www.w3.org/wiki/HTML/Elements/track)
5 KINDS OF TEXT TRACKS
TYING INFORMATION TO VIDEO

**SUBTITLES**
transcription or translation of the dialogue, suitable for when the soundtrack is unavailable.

**CAPTIONS**
transcription or translation of the dialogue, sound effects, relevant musical cues, and other relevant audio information, suitable for when the soundtrack is unavailable.

**DESCRIPTIONS**
Textual descriptions of the video component of the media resource, intended for audio synthesis when the visual component is unavailable.

**CHAPTERS**
Chapter titles, intended to be used for navigating the media resource.

**METADATA**
Tracks intended for use from script.
**Track Files**

**Web Video Text Tracks (VTT)**

- Mark up layout for text track information.
- Each subtitle cue separated by a blank line.
- Closed captioning when on supersedes subtitles.

---

**WebVTT File Inclusion**

Since the browser is including the WebVTT for the track the WebVTT must be served from a web server the file cannot be read from a file:// directive.
**TRACK FILE FORMAT**

**VTT FILE STRUCTURE**

- Header statement
- Individual Cues
  - An identifier
  - Start to end time
  - Text for cue

**WEBVTT FILE**

```
Subtitle1
00:00:11.000 --> 00:00:14.000
Cosa ti porta nella terra dei guardiani?

Subtitle2
00:00:18.000 --> 00:00:20.000
Sto cercando qualcuno.

Subtitle3
00:00:36.000 --> 00:00:40.000
Una pericolosa missione per un cacciatore solitario.
```
SUBTITLE TRACKS
TIMELINE COORDINATION

• One or more cues handled automatically by html5 player.
• Cues contain an id, start time, end time, and text.

**timeline diagram**

- **cue1**
  - startTime: 00:01:250
  - endTime: 00:03:250
  - Cue1 text displayed

- **cue2**
  - startTime: 00:05:00
  - endTime: 00:08:00
  - Cue2 text displayed
EXAMPLE: SIMPLE SUBTITLES
VIDEO LOADEDMETADATA EVENT
WHEN ARE TRACKS AVAILABLE

- Fired when track sources all loaded.
- Track sources are loaded asynchronously.
- Not included in the window load event.
- Must set separate event listener for `loadedmetadata`

```javascript
video.addEventListener('loadedmetadata', function () {
    // Perform functions on video here.
}, false);
```
**TRACK MODE**

**SETTING TRACK VISIBILITY**

- The track visibility is set through the `mode` property.

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disabled</td>
<td>Text track is disabled for the video element.</td>
</tr>
<tr>
<td>Hidden</td>
<td>Text track is hidden from being shown.</td>
</tr>
<tr>
<td>Showing</td>
<td>Text track is currently being used to show.</td>
</tr>
</tbody>
</table>
EXAMPLE: CHANGING TRACKS
Allowing the user to select their language of subtitles. Leverages pulling all tracks, kind, and mode.
METADATA
USING METADATA FOR TRIGGERS
METADATA TRACKS
LEVERAGING JAVASCRIPT

- Metadata track cues not shown by video player.
- Many potential uses choreographing functionality.

```html
<video>
  <source ...
  ...
  <track kind="metadata" label="Quiz Cues" src="quiz-metadata-en.vtt" default>
</video>
```

METADATA TRACK LOADING
To have a metadata track loaded the default attribute must be used in the <track> element.
METADATA TRACKS

JSON INSIDE CUES

• Text field can include complex data.
• Read like normal text property and then parsed.

Bubble1
00:00:05.000 --> 00:00:09.000
{
"title":"Fingerless Gloves",
"text":"Madonna made it fashionable to wear fingerless gloves, also called Glovelettes, in the 1980s with the record cover photo for Like a Virgin.",
"xpos":"450px",
"ypos":"350px"
}

• But first need to be able to grab each cue.
CUES
ACCESSING CUES FROM TRACKS
CUES
CUE OBJECT STRUCTURE

- Cues accessed as array from `textTrack` object.
- Properties of each cue can be accessed or modified.

```javascript
for (var cueIdx = 0; cueIdx < textTrack.cues.length; cueIdx++) {
  console.log("Cue id:" + textTrack.cues[cueIdx].id);
  ...
}
```

<table>
<thead>
<tr>
<th>id</th>
<th>The id of the specific cue.</th>
</tr>
</thead>
<tbody>
<tr>
<td>startTime</td>
<td>Start time as defined by cue.</td>
</tr>
<tr>
<td>endTime</td>
<td>End time as defined by cue.</td>
</tr>
<tr>
<td>text</td>
<td>Text description of cue.</td>
</tr>
</tbody>
</table>
TRAPPING CUE EVENTS
TWO METHODS TO CATCH CUE CHANGES

• Cue events
  – enter (onenter)   fired when a cue is entered.
  – exit  (onexit)    fired when a cue is exited.

• Text track cuechange (oncuechange).
  – Fired whenever the list of active cues changes.

```javascript
  textTrack.oncuechange = function () {...};
```
  – oncuechange appears to be more reliable than cuechange
TRAPPING CUE EVENTS
TWO METHODS TO CATCH CUE CHANGES

- activeCues array of one or more active cues.
- If activeCues length is zero when cuechange fired then exiting cue.

```javascript
console.log(textTrack.activeCues[0].id);
```

CUE EVENT RELIABILITY
Cue events are known to be problematic with inconsistent behaviors (events not firing, events ceasing to fire).
EXAMPLE: BUBBLES

Using cue events and cuechange to effect the famous music video pop up bubbles.

Fraught with trouble. Don’t look down and step away from the edge.
EXAMPLE: BUBBLES NEW

Using what we know with timeline update events to create a bubble solution.
TRACKS ON THE FLY
CREATING TRACKS AND CUES

- In JS can also add a new text track.
- `addTextTrack` takes the kind, label and option language code.

```javascript
newTextTrack = myVideo.addTextTrack('chapters','bookmarks');
newTextTrack.default = true;
```

- New cues added with `addCue(cue)`.

```javascript
var newCue = new TextTrackCue(startTime, endTime, 'label');
newCue.id = 'bookmark1';
newCue.pauseOnExit = false;
newTextTrack.addCue(newCue);
```

- Cues can also be removed with `removeCue(cue)`
EXAMPLE: BOOKMARKS

Create cues dynamically and list out the cue list. Navigate to bookmarks created by user.
**COMPLEX INTERACTIONS**
SHOWING VIDEO ON CANVAS ELEMENTS

- Take frame images from video and place on canvas.
- Video element is hidden while canvas is shown.
- Leverages
  - canvas `getContext("2d")`
  - `drawImage`, `getImageData`, and `putImageData`

**Image Copied**

```
<video>
<canvas>
```

[Dynamic Content Injection example]
COMPLEX OVERLAYING
ADDING SPECIAL EFFECTS

- Timing challenge with direct processing.
- Frame buffer needed for holding frames from timer.
RESOURCES
VARIOUS HTML5 VIDEO RESOURCES

• Documentation
  – Whatwg.org video element -

• 3rd Party Libraries to aid development (not used here)
  – cuepoint.js    cuepoint.org    Jquery plugin for subtitles and creating cue points.
  – popcorn.js     popcornjs.org   JS Media framework for easier time-based interaction.
  – video.js       videojs.com     Open source html5 video player and library.
Q&A

Code samples (will be posted)
https://github.com/cahudson/html5video

Chuckahudson+smw@gmail.com
VIDEO CREDITS

- Thanks to the video providers
  - Blender Foundation Open Source Projects [http://www.blender.org/features/projects/]