Video and Audio

In this section, focusing on the video and audio elements.

Overview of Web Media

HTML allows us to create standards-based video and audio players that don't require the use of plugins. Adding video and audio to a webpage is almost as easy as adding an image or formatting some text. We will look at a simple project.

HTML has a tag, the <video> tag for adding video to a web page.

The Video Tag

After you have your video compressed in the right format, it's really easy to embed it onto your website.

There are two tags we need to embed video on our website:

<video> — the wrapper, with attributes to adjust functionality.

<source> — used to present a video format.

A simple example of embedding a video on a website looks like this:

```
<video src="video/myvideo.mp4"></video>
```

Fallback for video

In between the open and close <video> tags, you should put some fallback content, kind of like the alt attribute on images.

```
<video src="video/myvideo.mp4">
  Some content
</video>
```

Multiple formats

If you wanted to support more older browsers with different formats you can use multiple `<source>` tags:

```
<video>
  <!-- Notice the `type` attribute to help the browser determine
what video to play -->
  <source src="video/myvideo.mp4" type="video/mp4">
  <source src="video/myvideo.webm" type="video/webm">
  <source src="video/myvideo.ogv" type="video/ogg">
</video>
```

Controls

By default, there are no playback controls on a video, we need to tell the browser to add those with the controls attribute.

```
<video src="video/myvideo.mp4" controls>
  Contents….
</video>
```

The browser will then present its default set of player controls—that you cannot style.

Poster

The poster attribute allows you to specify an image that will be displayed to the user before they start playing the video. It can be in any format the browser supports, but likely you want to use JPG or PNG.

```
<video src="video/myvideo.mp4" controls poster="videoImage.jpg">
  Contents...
</video>
```

Auto play

There is an `autoplay` attribute you can use on the video tag, it will automatically download and play the video.

Try to avoid auto playing videos—especially videos with sound. Some devices don't even allow auto play, they require a user interaction to start a video.

If you must auto play your video make sure to mute the sound with the muted attribute.

```
<!-- If you must auto play, make sure to add the `muted` attribute to avoid auto playing
sounds. -->
<video autoplay muted>

</video>
```

Specifying playback range

When writing in the `src` attribute of a video you can choose where the video should start and end.

We just need to add a hash onto the URL for the browser to register the range.

```
<video src="video/myvideo.mp4#t=5,20">
  Contents...
</video>
```

The format of the hash follows this convention:

```
#t=[starttime],[endtime]
```

Some examples:

`#t=10,20` — start at 10 seconds, end at 20 seconds.

`#t=,40` — start at the beginning, play until 40 seconds.

`#t=60` — start at 60 seconds, play to the end.

You can even specify the time in `hours:minutes:seconds`:

`#t=,2:58:00` — start at the beginning, play until 2 hours, 58 minutes in.

Responsive video

You'll most likely want to make your video flexible and scale with the dimensions of your website.

If your video is in the standard aspect ratio for HD of 16 by 9, your HTML will look something like this.

```
<div class="embed embed-16-9">
  <video class="embed-item" src="video/myvideo.mp4">
    Contents...
  </video>
</div>
```

And the CSS will look something like this:

```
body {
  width: 90%;
  margin: auto;
}

.embed {
  margin-left: 0;
  margin-right: 0;
  position: relative;
  width: 100%;
}

.embed-16-9 {
  padding-top: 56.25%;
}
```

```
.embed-item {
  height: 100%;
  left: 0;
  position: absolute;
  top: 0;
  width: 100%;
}
```

Using the above HTML & CSS will allow your video to scale while still maintaining the proper aspect ratio.

Adding captions & subtitles to video

Tracks and the `<track>` tag are a standard way to provide subtitles, captions, screen reader descriptions, chapters and more for your video.

We just need to add a `<track>` element to the `<video>` tag that points to the appropriate `.vtt` file:

```
<video src="video/myvideo.mp4">
  <track default src="video/subtitles.vtt" kind="subtitles"
srclang="en" label="English">
  Contents...
</video>
```

There are a few important attributes to look at:

`src="…"` — is a URL that points the text file for this track.

`kind="…"` — what kind of track it is: `subtitles`, `captions`, `descriptions`, `chapters` or `metadata`.

`srclang="…"` — the language of the content in the track.

`label="…"` — what the user will see in the interface describing this track.

`default=` it is important to have default

Also, for subtitles to show, the page should run on the server.

Subtitle tracks

Subtitles are used to add transcriptions or translations to the video. They are time-based so that specific words show on the screen in time with the video.

All the tracks use the WebVTT format, an open format for describing this type of content.

WebVTT

A WebVTT file is just a plain text file that you can create in your text editor—make sure to use the `.vtt` file extension when you save it.

Here's a basic example for a subtitle track:

```
WEBVTT

1
00:00:01.000 --> 00:00:10.000
Hey, how's it going?

2
00:00:15.000 --> 00:00:20.000
Thing's are good.
How you doin'?
```

The first line in the file must always be `WEBVTT`.

You can add an optional heading to each entry, like `1` above.

Next is the time code: `start --> end`. It's in the format of `hours:minutes:seconds.milliseconds`, but the hours are optional.

After the time code is the text to display, it can be written on multiple lines, but often it's better to let the device break the lines where appropriate.

Styling tracks

It's possible to style the tracks, especially subtitles and captions. Try to avoid doing too much style changes from the defaults.

Some of the basic things you can change are the following:

`<b>` — to add bold text.

`<i>` — to add italic text.

`<u>` — to add underlined text.

`<c.my-class>` — to add a class.

Some examples:

```
WEBVTT

1
00:00:01.000 --> 00:00:10.000
Curse your <i>sudden</i> but <b>inevitable</b> betrayal.

2
00:00:15.000 --> 00:00:20.000
I've got a <c.badClass>bad</c> feeling about this.
```

You could then use the .badClass *class in CSS to style the word.*

The audio tag

The audio tag has many of the same attributes as the video tag, here's a list of what you can use:

`controls` — displays the browser's default playback controls.

`muted` — whether it should be audible or silent.

`loop` — whether the sounds should be looped or not.

`volume="…"` — to set the volume of the sound: `0.0` is silent, `1.0` is the loudest, the sounds natural volume.

Audio formats

Currently the MP3 audio format is the most browser compatible.

If you want to support older browser versions you could supply OGG also using the `<source>` tag:

```
<audio controls>
  <source src="audio/audio.mp3" type="audio/mpeg">
  <source src="audio/audio.ogg" type="audio/ogg">
  [notes ...]
</audio>
```

Auto play

Using the `autoplay` attribute it is possible to make sounds play automatically on your website. But…

Don't auto play sounds. Ever, as it is annoying when some random sound blurts out.