Web Applications

HTML5
advanced features

Martin Caminada
Chris Jones

HTML5 versus XHTML

XHTML	HTML5
lower case element names	lower case or upper case element names
lower case attribute names	lower case or upper case attribute names
attribute values have to be in quotes	attribute values can be in quotes
each non-empty element needs closing tag	some non-empty elements don’t need closing tags
each empty element needs to be self-closed	empty elements don’t need to be self-closed

HTML5 versus XHTML

XHTML only accepts	HTML5 also accepts
<table>	<TABLE> or <Table>
	or
<meta charset="UTF-8">	<meta charset=UTF-8>
some text some more text	some text some more text
	<img src="dog.jpg" alt="dog">

HTML5 advanced features

- “semantic elements” for parts of a page
- form validation and new form input types
- video / audio
- MathML for displaying mathematical formulas
- JavaScript APIs: web storage, geolocation, drag and drop, graphics with “canvas”, …
- …

Semantic elements

<header>

Banner / header at beginning of page or beginning of a section of page

<footer>

Wraps a footer, i.e. content at end of page, such as contact details or copyright notices

<nav>

A section that contains navigational material - menu

Semantic elements

<section>

A logically related part of a web page

-e.g. “news”, “contact details”, “introduction” or a chapter of a document

<article>

A relatively independent piece of content, such as blog entry (article), or a particular news article

<aside>

Content separate from main content, - as found in a side bar (separate to right or left) e.g. adverts, notes on topics that the user might also be interested in.

Semantic elements

<time> To specify a date as *YYYY-MM-DD*

or date + time *YYYY-MM-DDThh:mm±hh:mm*

Where *±hh:mm is time zone*

Many variations, e.g.

```
<time datetime="2012-12-26"> 26th December 2012
</time>
```

```
<time datetime="2012-12-26T10.30+00:00"> 26th
December 2012 at 10.30 GMT</time>
```

Or

```
<time datetime="2012-12-26T10.30Z">
26th December 2012 at 10.30 GMT </time>
```

Video / audio

<video>

<audio>

- Replace need for third part plug-in (e.g. Adobe Flash, Apple QuickTime), which required <object> and <embed> elements
- E.g.

```
<video src="CuteCat.webm"
                controls type="video/webm">
</video>
```

<source>

Not all browsers recognise all formats, so need to provide the video in multiple formats, using the source element

```
<video  controls>
  <source src="CuteCat.mp4"
                type="video/mp4" >
  <source src="CuteCat.webm"
                type="video/webm">
  <source src="CuteCat.ogv"
                type="video/ogg">
</video>
```

Containers and codecs

- Video file formats such as

  MPEG4 (.mp4), WebM (.webm), OGG (.ogg), Flash (.flv) and AVI (.avi)

are containers for

- video track (or stream)
- audio track (or stream)
- metadata about the tracks (e.g. titles of the tracks, aspects ratios of videos)

Video players decode tracks in the container and output synchronized video and audio signals

CODECS

- Many ways to encode and decode video and audio (between analog and digital and *vice versa*)
  - Data compression methods: lossy vs lossless

A codec is an algorithm for coding and decoding e.g.

Video: H.264, Theora, VP8

Audio: MP3, Vorbis, AAC

Some Video codecs for HTML5

H.264 = MPEG-4 part 10 (MPEG-4 AVC)

Multiples *profiles* for vary levels of quality (e.g. baseline, main, high). (Blue ray uses high)

-Patented

Theora – developed by Xiph.org Foundation

evolved from VP3. No Patents

VP8 – from On2 – now Google

similar quality to H.264, but not patented

Some Audio codecs for HTML5

MP3 = MPEG 1 – Audio Layer 3 : Patented

1 or 2 channels,

various bit rates – can adapt to content

AAC (Advanced Audio Coding) : Patented

Up to 48 channels. Multiple profiles

(better quality than MP3)

Vorbis (or Ogg Vorbis) : Not Patented

fixed / variable bit rate. Up to 255 channels

Similar quality to AAC

Some containers supported on Web

MPEG-4 (.mp4)

H.264 + AAC

(Chrome, Firefox, IE, Opera, Safari)

OGG

Theora + Vorbis

(Chrome, Firefox, Opera)

WebM

VP8 + AAC

(Chrome, Firefox, Opera)

<video> attributes

```
<video src="CuteCat.webm"
   type="video/webm"
   width="500" height="480" controls >
</video>
```

width and **height** – of the video display

controls – to provide usual play, pause etc controls

autoplay – to download and start playing automatically

type – tells browser about the format

<audio>

Audio similar to video, but no size attributes, e.g.

```
<audio  controls>
<source src="DogBarking.mp3"
                              type="audio/mp3" >
<source src="DogBarking.ogv"
                              type="audio/ogg" >
</audio>
```

<canvas>

Graphics: lines (paths), rectangles, symbols, images, text

- colours, styles, fonts…

Specify canvas size in pixels

Drawing coordinates in pixel units

- origin in upper left corner
  - x coordinates rightwards
  - y coordinates downwards

Getting started

Insert a <canvas> element with specified id, height and width

Get a reference to it and create a graphics context object:

```
<canvas id="can1" width="300"
                  height="300"> </canvas>
<script type="text/javascript">
  canv = document.getElementById("can1");
  ctx = canv.getContext("2d");
…. </script>
```

Draw a rectangle

fillRect(xmin, ymin, xmax,ymax)

```
<canvas id="can1" width="300"
                        height="300"></canvas>
<script type="text/javascript">
  canv = document.getElementById("can1");
  ctx = canv.getContext("2d");
  ctx.fillStyle = 'rgb(255,0,0)';
  ctx.fillRect(5,5,200,100); //filled red
  ctx.strokeStyle = 'rgb(0,200,100)';
  ctx.strokeRect(5,5,200,100); //boundary
</script>
```

arc

arc(centreX,centreY, radius, startAng,
endAng, anti-clockwise[boolean])

```
<script type="text/javascript">
  canv = document.getElementById("can1");
  ctx = canv.getContext("2d");
  ctx.beginPath();/* start new path */
  ctx.arc(85,80, 20, 2*Math.PI, 1*Math.PI,
        false );  /*false = clockwise */
  ctx.lineWidth = 10;
  ctx.stroke();
</script>
```

lines

```
moveTo(x,y)  // go to start of line
lineTo(x,y)  // trace to end of line
stroke()     // draw the line
```

```
ctx.beginPath();
ctx.moveTo(70, 70);
ctx.lineTo(80, 55);
ctx.lineTo(90,70);
ctx.strokeStyle = 'rgb(50,50,200)';
ctx.lineWidth= "4"; //in pixels
ctx.stroke();
```

Favicons

- favicon (*favourite icon*)
  small icon that is displayed in the browser tab

```
<link rel="icon" type="image/png"
      href="HTML5logo.png" />
```

- Warning: older versions of IE expect the favicon to be called `favicon.ico` and located in the root directory.
  (see the favicon Wikipedia page for details)

HTTP Redirecting

- Not part of HTML5 but still useful: redirecting

```
<meta http-equiv="refresh"
content="5; url='http://www.cs.cf.ac.uk'" />
```

- This simulates a HTTP header (http-equiv)
- Technique is officially deprecated (accessibility) but can still be useful when having to redirect users to a new location of a web site.