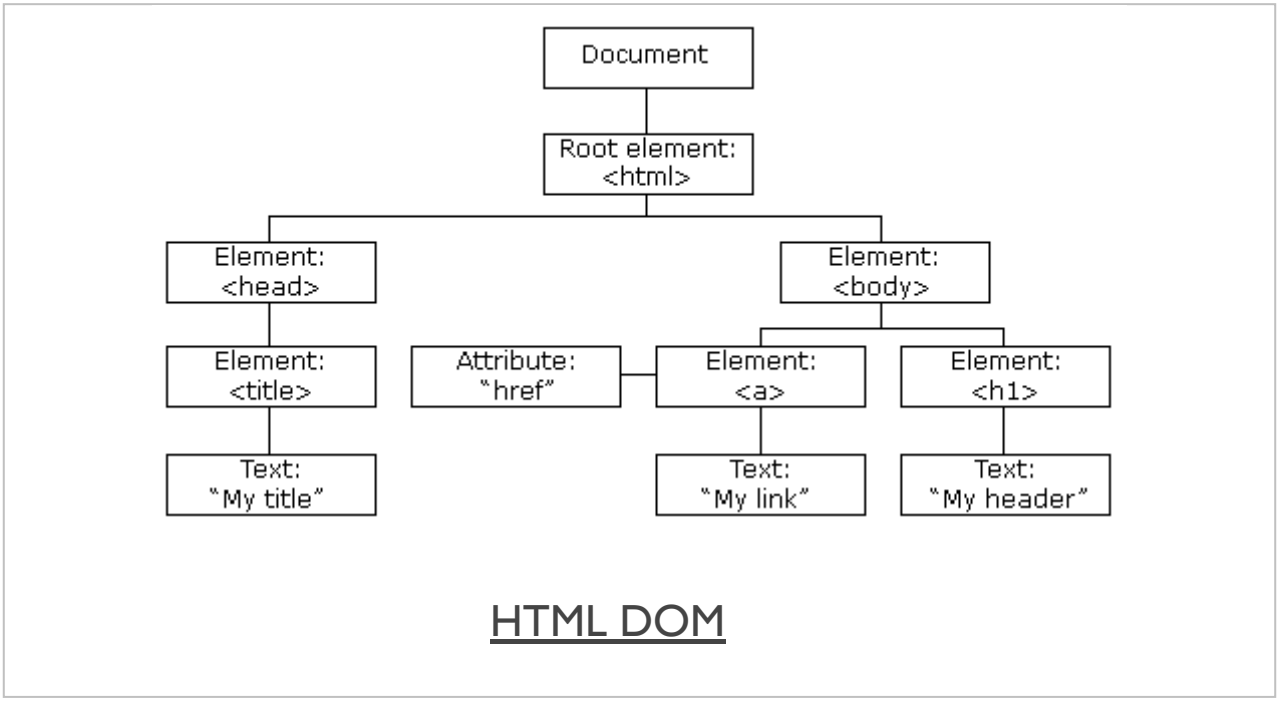
Comp 424 - Client-side Web Design

Fall Semester 2016 - Week 3

Dr Nick Hayward

DOM Basics - intro

A brief introduction to the document object model (DOM)



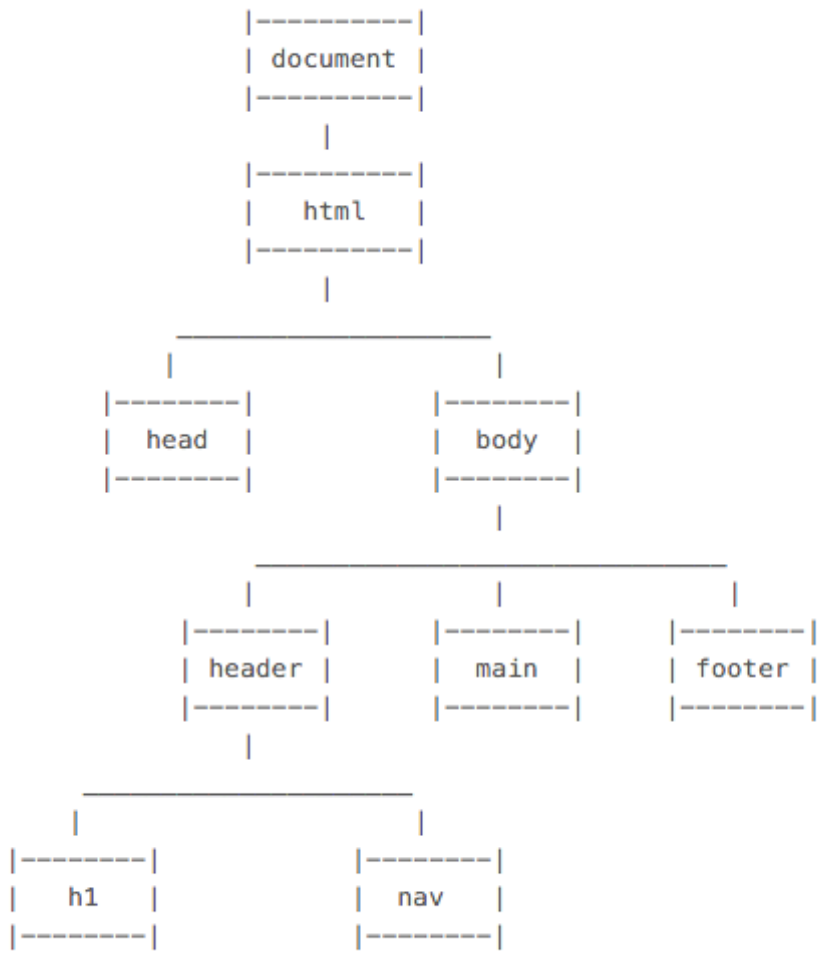
HTML DOM

- Source - W3Schools - JS HTML DOM

DOM Basics - what is DOM?

- **DOM** - platform and language independent way
  - *to access and manipulate underlying structure of HTML document*
- structured as a representation of a tree data structure
  - *its manipulation follows this same, standard principle*
- DOM tree is constructed using a set of nodes
  - *tree is designed as a hierarchical representation of the underlying document*
- each node on our tree is an element within our HTML document
- inherent hierarchical order originates with the **root** element
  - ***root** sits at the top of our **tree***
  - *descends down following lineage from node to node*
- each node is a child to its parent
  - *we can find many siblings per node as well*
- root at the top of the tree...

Image - HTML DOM



HTML DOM

DOM Basics - useful elements

element tag	usage & description
`<html>`	container element for a HTML document
`<head>`	contains metadata and document information
`<body>`	contains main content rendered as the HTML document
`<header>`	page header...
`<nav>`	navigation, stores and defines a set of links for internal or external navigation
`<main>`	defined primary content area of document
`<footer>`	page footer...
`<section>`	a section of a page or document
`<article>`	suitable for organising and containing independent content
`<aside>`	defines content aside from the content which contains this element
`<figure>`	logical grouping of image and caption
`<img>`	image - can be local or remote using url in `src` attribute
`<figcaption>`	image caption
`<h1>, <h2>...`	headings from 1 to 6 (1 = largest)
`<a>`	anchor - link to another anchor, document, site...
`<p>`	paragraph
`<ul>, <ol>, <dl>`	unordered, ordered, definition lists
`<li>`	list item, used with `<ul>, <ol>...`
`<dt>`	definition term, used with `<dl>`
`<dd>`	definition description, used with `<dl>`

`<table>`	standard table with rows, columns...
`<tr> >`	table row, used with `<table>`
`<th>`	table heading, used with `<table>` and child to `<tr>`
`<td>`	table cell, used with `<table>` and child to `<tr>`
`<div>`	non-semantic container for content, similar concept to `<section>`
`<span>`	group inline elements in a HTML document
`<canvas>`	HTML5 element for drawing on the HTML page
`<video>`	HTML5 element for embedding video playback
`<audio>`	HTML5 element for embedding audio playback

NB: *`<div>` and `<span>` can be used as identifiers when there is no other suitable element to define parts of a HTML5 document. e.g. if there is no defined or significant semantic meaning...*

DOM Basics - sample

```
<!DOCTYPE html>
<html>
<head>
<base href="media/images/">
<meta charset="UTF-8">
<!-- week 3 - demo 1 -->
<title>Week 3 - Demo 1</title>
</head>
<body>
  <header>
  <h1>Ancient Egypt</h1>
  </header>
  <nav>...</nav>
  <main>
    <section>
      <p>
        Welcome to the Ancient Egypt information site.
      </p>
      <figure>
        <img src="philae-demo2.jpg" alt="philae temple" width="333px"
        height="200px">
        <figcaption>Ptolemaic temple at Philae, Egypt</figcaption>
      </figure>
    </section>
    <aside>
      Temple at Philae in Egypt is Ptolemaic era of Egyptian history.
    </aside>
  </main>
  <footer>
    foot of the page...
  </footer>
</body>
</html>
```

- Demo - DOM Basics - Sample

DOM Basics - `index.html` page

`index.html` usage and structure

- basic `index.html` page for loading web apps
- app will start with the `index.html` document
  - *html pages saved as `.html` or `.htm`*
  - *`.html` more common...*
- `index.html` acts as a kickstart
  - *for loading and rendering the app*
  - *loads other app resources - CSS, JS...*
- consistent elements in the HTML DOM
  - *`<html>`, `<head>`, and `<body>`*
- HTML5 apps will add
  - *`<header>`, `<main>`, and `<footer>` (when required)*
  - *many other elements for building the app...*

HTML Basics - <head> element

- part of a HTML document's metadata
- allows us to set metadata for a HTML page
- customised just for that page or replicated as a site-wide implementation
- we can add numerous additional elements to <head>
- add similar links and code for JavaScript
  - *use the <script> element & attributes such as type and src*

```
<script type="text/javascript" src="script.js">
```

- add a <title> element with text added as the element content
- set a default base address for all relative URLs in links within our HTML

```
<base href="/media/images/" target="_blank">
```

- links now simply use the base URL or override with full URL

```
<img src="image.jpg">

<a href="http://www.flickr.com">Flickr</a>
```

HTML - <head> element example

```
<head>
  <meta charset="utf-8">

  <title>Sample...</title>
  <meta name="description" content="sample metadata">
  <meta name="author" content="COMP424">

  <link rel="stylesheet" type="text/css" href="style.css">
  <script type="text/javascript" src="script.js">

</head>
```

HTML Basics - <body> - part I

- to define the main body of the web page we use the <body> element
- headings can be created using variants of
  - *<h1>, <h2>.....<h6>*
- we can now add some simple text in a <p> element

```
<p>...</p>
```

- add a line break using the
 element
- <hr /> element adds a horizontal line
  - *implies rendering division*
  - *instead of defined structural divide...*
- comments can also be added through our HTML

```
<!-- comment... -->
```

HTML Basics - <body> - part 2

Linking in HTML

- linking is an inevitable part of web design and HTML usage
- can be considered within three different contexts
  - *linking to an external site*
  - *linking to another page within the same site*
  - *linking different parts of the same page*
- add links to text and images within the HTML
- <a> element for links plus required attributes, e.g.

```
<!-- external link -->
<a href="http://www.google.com/">Google</a>
<!-- email link -->
<a href="mailto:name@email.com">Email</a>
<!-- internal page link -->
<a href="/another_page.html">another page</a>
<!-- define internal anchor - using name attribute -->
<a name="anchor">Internal anchor</a>
<!-- define internal anchor - using ID attribute -->
<a id="anchor">Anchor</a>
<!-- internal anchor -->
<a href="#anchor">Visit internal anchor</a>
<!-- internal anchor on another page -->
<a href="/another_page.html#anchor">Visit internal anchor</a>
<!-- internal anchor on a page on an external site -->
<a href="https://www.test.com/test.html#anchor">Visit internal anchor on external site</a>
```

- Demo - HTML - Internal Anchor

HTML Basics - <body> - part 3

Linking in HTML - continued

- standard attributes supported by <a> element include
  - *class, id, lang, style, title...*
- optional attributes are available for <a> element including
  - *target, href, name...*
- target attribute specifies where the link will be opened relative to the current browser window
- possible attribute values include

```
_blank
_self
_parent
_top
```

HTML Basics - <body> - part 4

Working with images

- <img> allows us to embed an image within a web page
- <img> element requires a minimum *src* attribute

```
<img src="image.jpg" />
```

- other optional attributes include
  - *class, id, alt, title, width, height...*
- use images as links
- image maps

```
<map name="textmap">
  <area shape="rect" coords="..." alt="Quote 1" href="notes1.html" />
</map>
```

HTML Basics - <body> - part 5

Adding a table

- organise data within a table starting with the `<table>` element
- three primary child elements include
  - *table row, table header, table data*
  - *`<tr>, <th>, <td>`*

```
<table>
<caption>424 - basic test table</caption>
<tr>
<th>heading 1</th>
<th>heading 2</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

- also add a `<caption>`
- span multiple columns using the `colspan` attribute
- span multiple rows using the `rowspan` attribute

HTML Basics - <body> - part 6

Organising a list

- unordered list <ul>, ordered list <ol>, definition list <dl>
- and <ol> contains list items

```
<ul>
<li>...</li>
</ul>
```

```
<ol>
<li></li>
</ol>
```

- definition list uses <dt> for the item, and <dd> for the definition

```
<dl>
<dt>Game 1</dt>
<dd>our definition</dd>
</dl>
```

HTML Basics - <body> - part 7

Using forms

- used to capture data input by a user, which can then be processed by the server
- `<form>` element acts as the parent wrapper for a form
- `<input>` element for user input includes options using the *type* attribute
  - *text, password, radio, checkbox, submit*

```
<form>
Text field: <input type="text" name="textfield" />
</form>
```

- process forms using
  - *e.g. JavaScript...*

HTML5 - intro

- finally became a standard in October 2014
- introduces many new features to HTML standard
- additional features include, e.g.
  - *new canvas element for drawing*
  - *video and audio support*
  - *support for local offline storage*
  - *content specific elements*
  - *including article, footer, header, nav, section*
  - *form controls such as*
  - *calendar, date, time, email, url, search*
- new input type attribute values
  - *assigned to provide better input control*
- Check browser compatibility using HTML5 Test

HTML5 - basic template

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title></title>
</head>
<body>


</body>
</html>
```

HTML5 - Elements - part 1

- often known simply as **tags**
- elements allow us to add a form of metadata to our HTML page
- for example, we might add

```
<!-- a paragraph element -->
<p>add some paragraph content...</p>
<!-- a first heading element -->
<h1>our first heading</h1>
```

- this metadata used to apply structure to a page's content

HTML5 - Elements - part 2

- we can now add additional structure to our basic template

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<!-- title for the web page appears in the window, tab heading... -->
<title>Demo 1</title>
</head>
<body>
<h1>Our first web page</h1>
<p>
As we build our web apps, more elements and content will be added...
</p>
</body>
</html>
```

- Demo - Our first web page

HTML5 - Comments

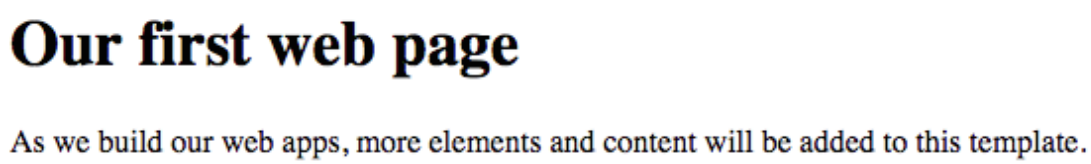
- comments are simple and easy to add to HTML
- add to HTML code as follows,

```
<!-- a comment in html -->
```

- comment not explicitly visible to the user in the rendered page
- comment appears in the code for reference...

image of HTML5 sample rendering 1

Image - rendering of demo 1



Source - Demo 1

HTML5 - semantic elements - part 1

- new semantic elements added for HTML5
- known as **block-level** elements
  - *includes the following elements,*

```
<article>
<aside>
<details>
<figure>
<figcaption>
<footer>
<header>
<main>
<nav>
<section>
```

- better structure underlying documents
  - *add clear semantic divisions*

HTML5 - semantic elements - part 2

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<!-- our second demo with lots of new elements -->
<title>Demo 2</title>
</head>
<body>
<header>
<!-- navigation elements, links... -->
<h1>Our first web page</h1>
</header>
<nav>Option 1</nav>
<main>
<section>
<p>
As we build our web apps, more elements and content will be added...
</p>
<figure>
<img src="media/images/philae-demo2.jpg" alt="temple of philae"
width="333px" height="200px">
</figure>
</section>
<aside>
Temple at Philae in Egypt is Ptolemaic era of Egyptian history...
</aside>
</main>
<footer>
foot of the page...
</footer>
</body>
</html>
```

- Demo - New elements added

image of html5 sample rendering 2

Image - rendering of demo 2

Our first web page

Option 1

As we build our web apps, more elements and content will be added to this template.

Temple at Philae in Egypt is Ptolemaic era of Egyptian history. Similar temples include Edfu... foot of the page...

Source - Demo - New elements added

HTML5 - semantic elements - part 3

- element tag `article` not used in previous demo
- `article` and `section` tag can both cause some confusion
- not as widely used as expected
- `div` element still widely seen in development
- HTML5 is supposed to have relegated `div`
  - *sectioning element of last resort...*
- `article` and `section`
  - *good analogy with a standard newspaper*
  - *different sections such as headlines, politics, health...*
  - *each section will also contain articles*
- HTML specification also states that an `article` element

> *represents a self-contained composition in a document, page, application, or site and that is, in principle, independently distributable or reusable, e.g. in syndication.*

HTML5 - semantic elements and structure - intro

- perceived issue or concern with HTML5 semantic elements
  - *how and when to add them to our document*
  - *where and when do we add them to our page?*
- non-semantic elements often considered simpler to apply
  - *generalised application and context for usage*

HTML - semantic elements and structure - header and nav

- `<header>`
  - *used to collect and contain introductory content*
  - *semantically appropriate for the head or top of a page*
  - *technically feasible and acceptable to include multiple `<header>` elements*
  - *e.g. `<header>` within main content, sidebar content, an article, a section...*
- `<nav>`
  - *short for navigation*
  - *stores and defines a set of links for internal or external navigation*
  - *not meant to define all page navigation links*
  - *often considered suitable for primary site links*
  - *additional links can be placed in*
  - *sidebar, footer, main content...*
  - *no need to consider a `<nav>` element for these links...*

HTML5 - Semantic elements and structure - `<main>`

- this element tag defines our **main** content
- traditionally the central content area of our page or document
- HTML4 often used a `<div>` element
  - *plus a `class` or `id` to define central content*
  - *e.g.*

```
<div id="main">
...
</div>
```

- HTML5 semantically defines and marks content as `<main>`
- `<main>` should not include any page features such as
  - *nav links, headers etc, that are repeated across multiple pages*
- cannot add multiple `<main>` elements to a single page
- must not be structured as a child element to
  - *`<article>`, `<aside>`, `<footer>`, `<header>`, or `<nav>`*

HTML5 - Semantic elements and structure - `<section>`, `<article>`, `<aside>` - part I

- `<section>`
  - *defines a section of a page or document*
  - *W3C Documentation defines as follows,*

> *a section is a thematic grouping of content. The theme of each section should be identified, typically by including a heading as a child of the section element.*

- a site can be sub-divided into multiple `<section>` groupings
  - *e.g. as we might consider a chapter or section break in a book...*
- `<article>`
  - *suitable for organising and containing independent content*
  - *include multiple `<article>` elements within a page*
  - *use to establish logical, individual groups of content*
  - *again, newspaper analogy is useful to remember*
  - *e.g. a blog post, story, news report...might be a useful article*
  - *key to using this element is often whether content can be used in isolation*
- `<aside>`
  - *used to define some content aside from containing parent content*
  - *normally used to help define or relate material to surrounding content*
  - *effectively acts as supporting, contextual material*

HTML5 - Semantic elements and structure - `<section>`, `<article>`, `<aside>` - part 2

- MDN Documentation suggests,

> *if it makes sense to separately syndicate the content of a `<section>` element, use an `<article>` element instead*

and

> *do not use the `<section>` element as a generic container; this is what `<div>` is for, especially when the sectioning is only for styling purposes. A rule of thumb is that a section should logically appear in the outline of a document.*

HTML5 - Semantic elements and structure - <figure>, <figcaption>

- <figure> & <figcaption>
  - *as with print media, we can logically group image and caption*
  - *<figure> acts as parent for image grouping*
  - *child elements include*
  - *<img> and <figcaption>*

```
<figure>
<img src="media/images/philae-demo2.jpg" alt="temple of philae"
width="333px" height="200px">
<figcaption>Ptolemaic temple at Philae, Egypt</figcaption>
</figure>
```

- updated demo with figure grouping - Demo - Semantic structuring

HTML5 - Semantic elements and structure - `<footer>`

- `<footer>`
  - *usually contains information about its containing element*
- example 1 - in a footer for an article
  - *might use this element to define and record*
  - *author or the article*
  - *publication date*
  - *suitable tags or metadata*
  - *associated documents...*
- example 2 - a footer simply placed at the **foot** of a page
  - *record copyright information*
  - *contextual links*
  - *contact information*
  - *small logos...*
- example 2 considered standard usage for `<footer>`
  - *continues from HTML4 and earlier usage...*

image of HTML5 page structure - part I

HTML5 Semantic elements - Part I

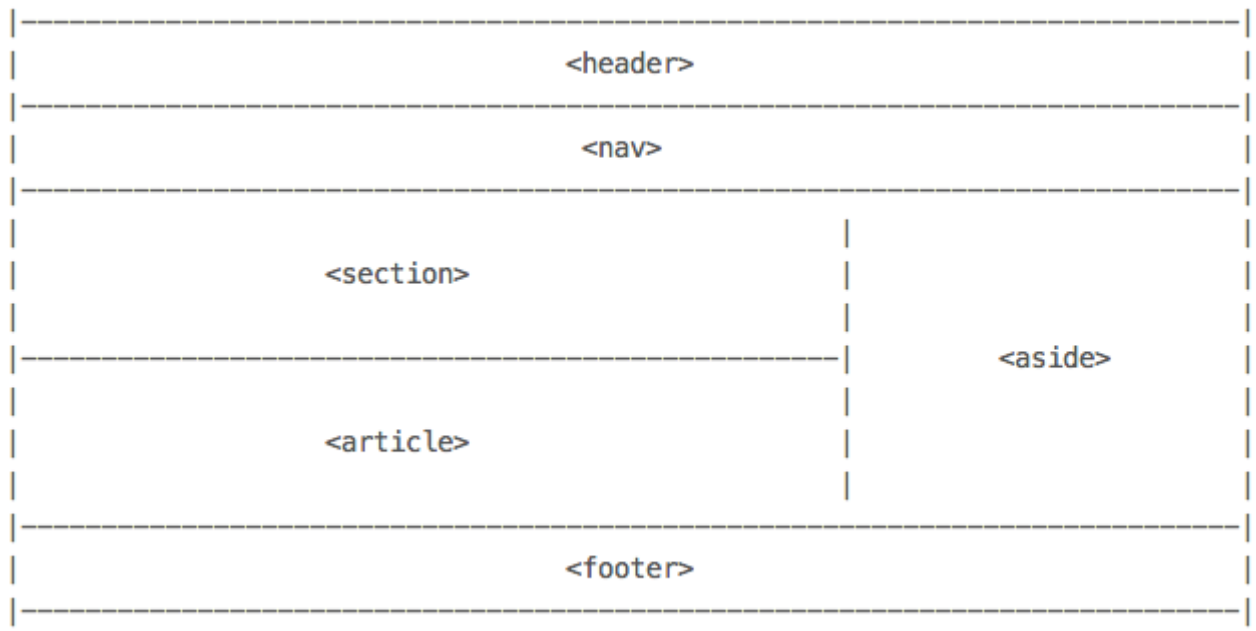
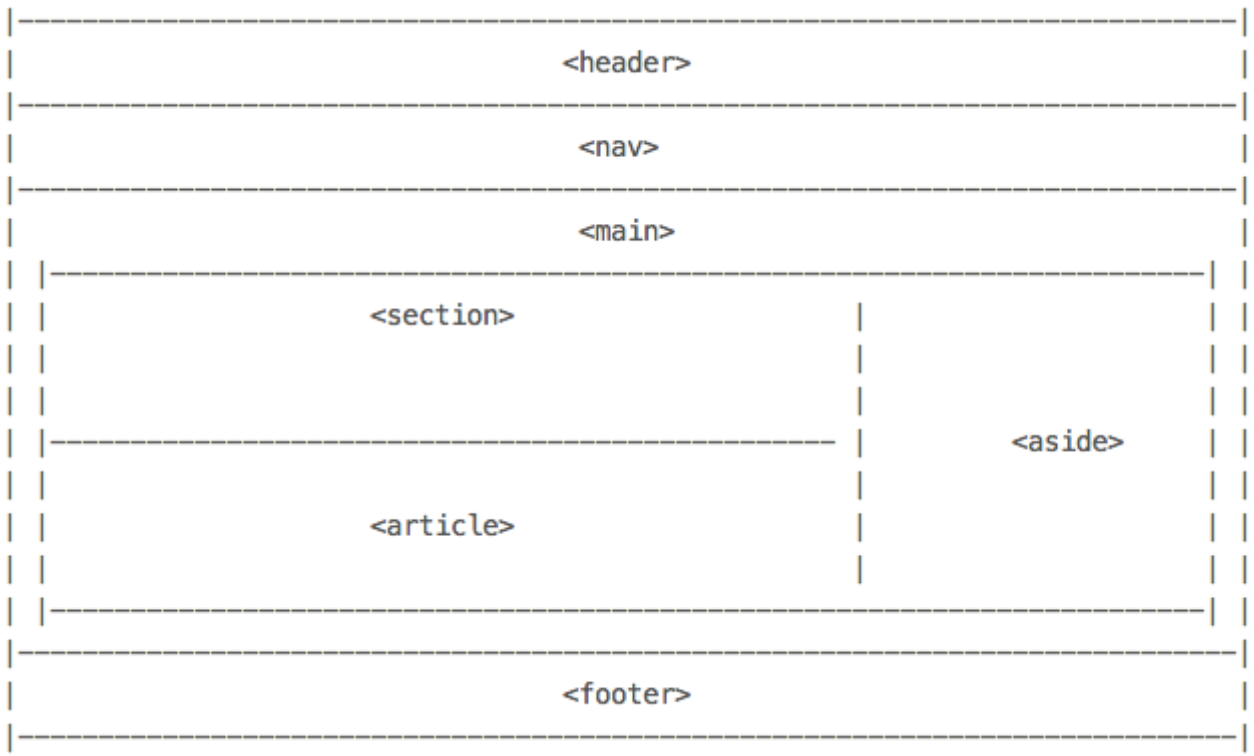


image of HTML5 page structure - part 2

HTML5 Semantic elements - Part 2



HTML5 page structure - part 3

- not included `<html>` and `<body>` tags in diagrams
  - *required for all HTML documents*
- divided the page into four logical, semantic divisions
  - *header*
  - *nav*
  - *main*
  - *footer*
- we could move `<nav>` into the `<header>` division at the top
  - *not always necessary and not compulsory*
- we could also add a sidebar etc for further division of content

HTML5 - extra elements - intro

- many other interesting and useful new HTML5 elements
  - *in addition to semantic elements*
- some struggle for browser compatibility
- useful new elements such as
  - *graphics and media*
- HTML5 APIs introduced as well, including
  - *App Cache*
  - *Drag/Drop*
  - *Geolocation*
  - *Local Storage*
  - ...
- again, check browser support and compatibility

Browser check

- Can I Use_____?
  - e.g. *Can I Use Drag and Drop?*

HTML5 - Extra elements - media - part 1

<video> element

- until HTML5, video playback reliant on plugins
  - *e.g. Adobe Flash*
- embed video using element tag <video>
- add attributes for
  - *height, width, controls...*
- not all web browsers support all video codecs
- option to specify multiple video sources
- best supported codecs include
  - *MP4 (or H.264), WebM, OGG...*
- good general support for <video> element
- check browser support for <video> element
  - *Can I use_____video?*

HTML5 - Extra elements - media - part 2

<video> - a quick example might be as follows,

```
<video width="300" height="240" controls>
  <source src="media/video/movie.mp4" type="video/mp4">
  <source src="media/video/movie.webm" type="video/webm">
  Your browser does not support the video tag.
</video>
```

- Demo - HTML5 Video playback

HTML5 - Extra elements - media - part 3

<audio> element

- HTML5 also supports standardised element for embedded audio
- supported codecs for <audio> playback include
  - *MP3 and mp4*
  - *WAV*
  - *OGG Vorbis*
  - *3GP*
  - *m4a*
- again, check browser support and compatibility
  - *Can I use_____audio?*
- fun test of codecs
  - *HTML5 Audio*

HTML5 - Extra elements - media - part 4

<audio> - a quick example might be as follows,

```
<audio controls>
  <source src="media/audio/audio.mp3" type="audio/mpeg">
  Your browser does not support the audio tag.
</audio>
```

- Demo - HTML5 Audio playback

HTML5 - Extra elements - graphics - part 1

- graphics elements are particularly fun to use
- use them to create interesting, useful graphics renderings
- in effect, we can draw on the page
- `<canvas>` element acts as a placeholder for graphics
  - *allows us to draw with JavaScript*
- draw lines, circles, text, add gradients...
  - *e.g. draw a rectangle on the canvas*

HTML5 - Extra elements - graphics - part 2

<canvas> will be created as follows,

```
<canvas id="canvas1" width="200" height="100">
  Your browser does not support the canvas element.
</canvas>
```

then use JavaScript to add a drawing to the canvas

```
<script type="text/javascript">
var can1 = document.getElementById("canvas1");
var context1 = can1.getContext("2d");
context1.fillStyle="#000000";
context1.fillRect(0,0,150,75);
</script>
```

Result is a rendered black rectangle on our web page.

- Demo - HTML5 Canvas - Rectangle

HTML5 - Extra elements - graphics - part 3

A cube can be created as follows,

```
<script type="text/javascript">
function draw() {
/*black cube*/
var can1 = document.getElementById("canvas1");
var context1 = can1.getContext("2d");
context1.fillStyle="#000000";
context1.fillRect(0,0,50,50);
}
</script>
```

Again, we end up with the following rendered shape on our canvas.

- Demo - HTML5 Canvas - Cube

HTML5 - Extra elements - graphics - part 4

- modify drawing for many different shapes and patterns
  - *simple lines, circles, gradients, images...*

1. shows different rendered shapes on a canvas.

- Demo - HTML5 Canvas - Assorted Shapes

2. little retro games

- Demo - HTML5 Canvas - Retro Breakout Game

Demos - DOM & HTML

- Demo - DOM Basics - Sample
- Demo - HTML - Internal Anchor
- Demo - Our first web page
- Demo - New elements added
- Demo - Semantic structuring
- Demo - HTML5 Video playback
- Demo - HTML5 Audio playback
- Demo - HTML5 Canvas - Rectangle
- Demo - HTML5 Canvas - Cube
- Demo - HTML5 Canvas - Assorted Shapes
- Demo - HTML5 Canvas - Retro Breakout Game

References - DOM & HTML

- HTML5 Audio formats
- HTML5 Test
- MDN - HTML developer guide
- Block-level elements
- Content categories
- Inline elements
- W3 Schools - HTML Block and Inline Elements
- W3C HTML5 Documentation
- W3Schools - HTML5 Semantic Elements