CI435 Introduction to Web Development

Lecture 4

Cascading Style Sheets (CSS)

Coursework progress check

- By this point you should have completed the first 3 lab tutorials –
  - Set up your website folders and files (lab tutorial 1)
  - Made a *Learning Journal* with lots of different HTML elements marking up text and images (lab tutorials 2-3)
  - Written – **in your own words** – 2 or 3 weekly posts about what and how you are learning
  - Included some references to sources you are using

Coursework progress check

- I will continue to use the *Learning Journal* as an example in the CSS lab tutorials
- But by now you should be keeping your own journal for real...
  - Writing weekly posts about what you are learning
  - Linking to online/offline resources you are using
  - Reflecting on your success, problems and solutions
- Web dev is about creating **interesting content** as well as the design and application of technology

This lecture will cover…

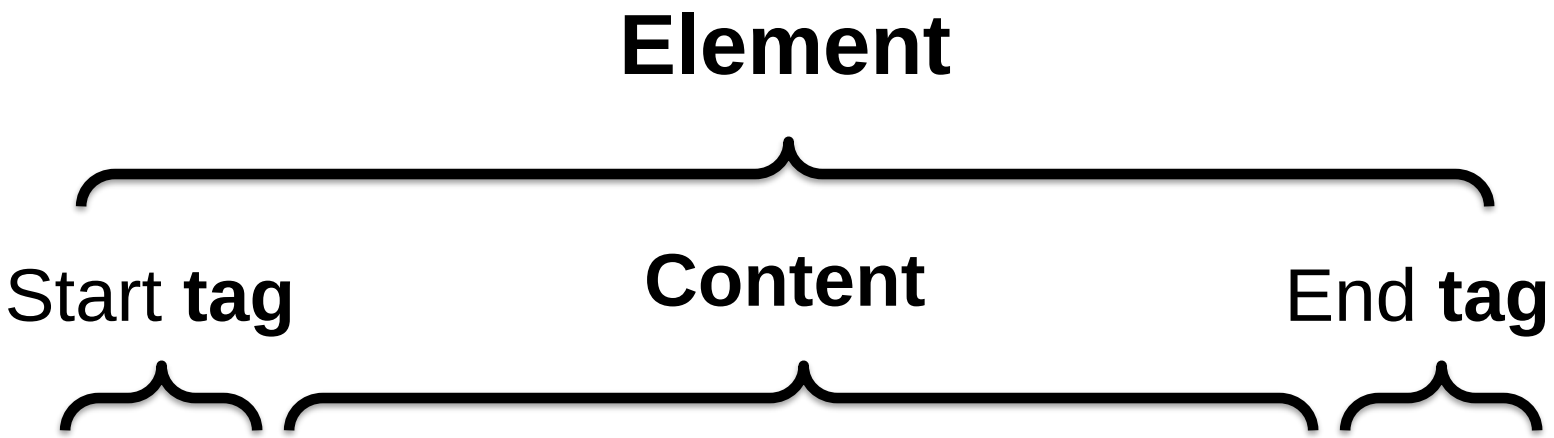
- Introduction to Cascading Style Sheets (CSS)
  - What is CSS? Background to CSS3
  - External style sheets, embedded and inline styles
  - CSS syntax – rules, properties and values
  - Selectors
  - Inheritance and specificity
  - `normalize.css`
  - This week's lab tutorial and basic style sheet
- This week's reading

CSS - Cascading Style Sheets

The following lectures cover CSS -

- **This week (4)** – CSS fundamentals; targetting HTML elements with CSS selectors
- **Next week (5)** – CSS box model and measurement
- **(Week 6) Responsive Web Design 1** – styling mobile first, CSS media queries, layout, fluid grids
- **(Week 7) Responsive Web Design 2** – Flexible media
- **(Week 8) Responsive Web Design 3** - CSS3 layout

Cascading stylesheets



** **A paragraph of text** **

- Elements are objects in the HTML document
- CSS rules target elements in order to instruct the browser how to present content

Cascading Style Sheets

- CSS is a **declarative language** for specifying the presentation of HTML elements by a browser
- CSS has been developed since 1998 by W3C
- **CSS Level 3 and Level 4 are** evolving rapidly: most level 3 **properties** are supported in the latest browsers

Reference –

- CSS properties -

  https://developer.mozilla.org/en-US/docs/Web/CSS/Reference
- Browser support - http://caniuse.com/#cats=CSS

CSS 3

- Using CSS 3 we can easily style HTML pages in ways that in the past could only be done with JavaScript, or digital images, *e.g.*
  - `border-radius` - http://jh1033.brighton.domains/ci435/tutorials/border-radius.html
  - Shadow on boxes and text
  - Opacity/transparency
  - Decorative borders
  - Importing fonts from the web
  - Gradients - http://lea.verou.me/css3patterns/
  - Animation - transitions and transforms - https://daneden.github.io/animate.css/ http://leaverou.github.io/animatable/ [be careful - less is more]

Progressive enhancement

- Old browsers don't support the most recent CSS 3 properties – but we can still use them
- If a browser cannot display a CSS 3 property the fallback will usually be acceptable
- *E.g.* CSS 3 `border-radius` and `box-shadow` – old browsers will display right angle corners and no shadow, but users can still access the HTML+content and the element looks OK
- One of the reasons why it's good practice to create well-formed, valid HTML and good content **FIRST**

Cascading Style Sheets

- CSS has led to economies in website development and maintenance -
  - Structure of content (HTML) and presentation (CSS) are **separate**: all presentation is controlled by a style sheet and can easily be changed
- Web page accessibility is improved -
  - Different style sheets can be attached to the same HTML document - *e.g.* a rich graphic style for GUI browsers; a print style sheet – optimised for the printed page with navigation elements removed

Attaching an external stylesheet

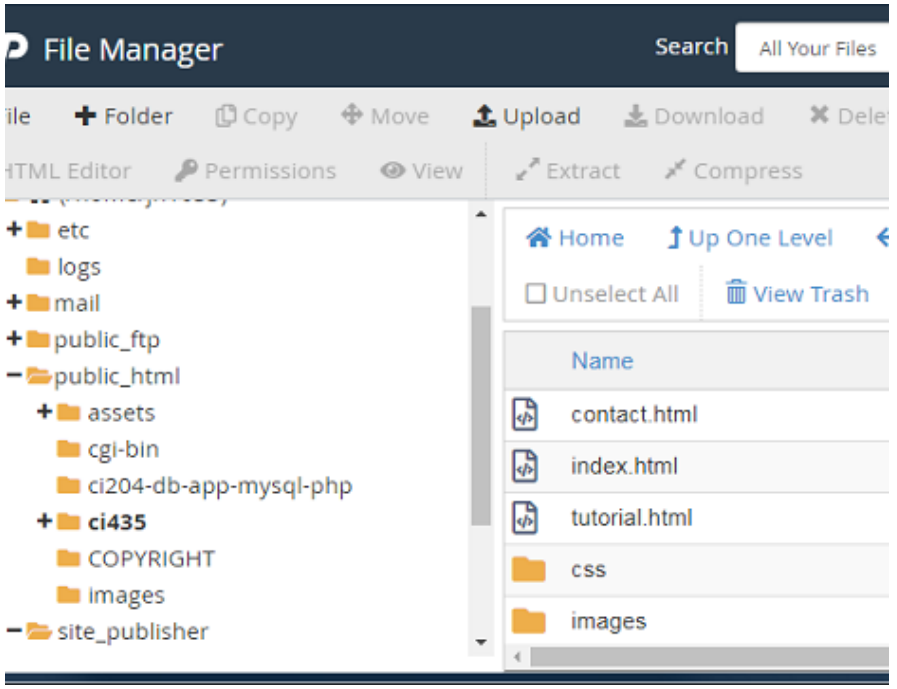
- Best practice is to write CSS in an **external style sheet**
- This file is attached to all HTML documents to which it applies by a **`<link>`** element in the document head -

```
<link href="stylesheet.css" rel="stylesheet" />
```

- The **`href`** attribute points to the style sheet file
- The **`rel` attribute** specifies the relationship between the web document and the file it is linked to – *i.e.* its style sheet

Attaching an external stylesheet

- Make a folder in your Brighton Domains webspace folder called 'css'
- Copy the basic style sheet `stylesheet.css` we have provided into this folder and add this line to ALL your HTML files, just after the `<title>` and `<meta>` elements (in the <head> section)




```
<title>Learning Journal</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link href="css/stylesheet.css" rel="stylesheet">
```

Embedded and inline CSS

- **Embedded (internal) style sheet** - the styles are written inside **<style>** tags in the **<head>** of the HTML document *e.g.*

```
<head>
 <style>
   h1 { color: blue; }
 </style>
</head>
```

  - Only do this if you are creating a single web page
- It's also possible to use the <style> attribute to create **inline styles** within the <body> of the HTML document –

```
<h1 style="color:blue;">Heading</h1>
```

  - This is the least maintainable – use only for testing

Embedded and inline CSS

- Embedded and inline CSS will over-ride an external style sheet – which makes pages hard to debug, change and maintain
- The **"cascade"** – lower level CSS over-rides higher level

1. The browser's default style sheet is over-ridden by…
2. **External style sheet** is over-ridden by…
3. Embedded CSS is over-ridden by…
4. Inline CSS

CSS syntax

- A style sheet consists of a number of **rules**
- These specify how the browser should present the elements in an HTML document
- A **rule** consists of two parts **-**
  - the **selector** which targets an HTML **element**
  - one or more **declarations**, inside curly braces, state how the element will be displayed
- A **declaration** also has two parts -
  - a **property**, *i.e.* the name of some aspect of presentation – *e.g.* `font-size`, `background-color`
  - a **value** for the property – *e.g.* `14px`, `#FFFFFF`

A CSS rule

Selector – targets all **<h1>** elements

Declaration – instructs browser how to present the element

```
h1 {
  font-family: Arial, Helvetica, sans-serif;
  color: #000000;
  font-size: 30px;
}
```

Note the punctuation – very important

Property – features of an element that can be styled

Value of the property

#000000 is the hexadecimal colour code for black https://htmlcolorcodes.com/

CSS properties

- CSS syntax is straightforward, but must be free of errors
- The scope of the language comes from 350+ properties that can be used to specify the presentation of an element
- When a number of properties are combined in a rule they can be used to create graphic effects like rollovers, patterns, gradients and layouts
- Learning CSS basics means learning properties and how to style them correctly
- You'll need to refer to an **index of CSS properties** - https://developer.mozilla.org/en-US/docs/Web/CSS/Reference

CSS selectors

- A selector is a **pattern** which matches the style rule to the HTML element *e.g.*

```
p {font-size: 18px;}
```

- The **p selector** matches all elements marked up with `<p>` tags and applies CSS style rules to them
- There are several different **types** of **selector**
- You will be learning how to use four selector types to target HTML elements in **lab tutorial 4**

Type (element) selector

- The **type selector** matches any element of that type
- *E.g.* **body** is the selector that matches the **<body>** element type: it's used to declare some base values for the whole document – such as the page background colour and font properties

```
body {
  background-color: #004080;
  font-family: Verdana, sans-serif;
  color: #FFFFFF;
  font-size: 16px;
}
```

Note U.S. spelling

Hexadecimal colour codes start with **#**

CSS – **inheritance** and **specificity**

- The `body` (or `html`) element selector is normally the **first rule** in the style sheet
- All HTML elements are 'children' of the **<body>** element, as they are nested inside it: they will **inherit** its CSS properties …
- …until a **specific** rule is written for an element with properties that over-ride the body properties
- The following rule - **`body { font-size: 16px; }`** - will determine the font-size of text, until another rule is created further down in the style sheet cascade -

```
footer { font-size: 14px; }
```

Type selector

- **`h1, h2...h6`** are the selectors for the **`<h1>, <h2>…<h6>`** elements. Declare properties that specify how the element will be styled *e.g.* `font-family, font-size, color, line-height`

```
h1      {
  color: #F00;
  font-size: 48px;
  line-height: 36px;
}
```

Use hex shorthand – when the numeric value is in 3 pairs

`#FF0000` = red

`#F00` = red

Type selector

- In the tutorial example *Learning Journal* the element **<header>** has been used several times, to markup all text content that is a heading – for the page, for an **<article>**, for an **<aside>**
- **header** is the selector that will target any element of the type **<header>** *e.g.*

```
header  {
   font-family: "Times New Roman", serif;
   margin-top: 4px;
   margin-bottom: 4px;
}
```

Values with spaces must be in quotation marks; multiple values separated by commas

Class selector

- What if we wanted to style the `article` **<header>** differently?
- Give all the `article` headers a **class attribute** –

```
<header class="post">
```

- And use a **class selector** to match the class attribute by its **value** –

```
.post {
  background-color: #069;
  margin-top: 2px;
  margin-bottom: 2px;
}
```

A class selector has a **full stop** followed by the class attribute value

id selector

- Another option would be to give the header for the top of the page – which has only one instance – an **id attribute**

```
<header id="banner">
```

- And use an **id selector** to target this unique element –

An id selector has a # followed by the class attribute value An id value can be used only once in an HTML document

```
#banner {
    background-color: #069;
    margin-top: 10px;
    margin-bottom: 6px;
}
```

Class or id attribute/selector?

- An id attribute value must be unique – only used once in an HTML document *e.g.* **`<header id="banner">`**
- A class attribute can be used one or more times in an HTML document *e.g.* **`<article class="post">`**
- It's OK to use a `class` attribute/selector for unique instances of elements rather than using `id` attributes and selectors – many web developers do this

Pseudo-class selector

- Some elements – such as **`<a>`** anchors (links) have certain **behaviours** when the user interacts with them
- **Pseudo-class selectors** target the element in its different states

Selector	Matched state or behaviour
`a:link`	source anchor of a hyperlink
`a:visited`	source anchor of a *visited* link
`a:hover`	anchor when the user *moves cursor* over it
`a:active`	anchor when the user *clicks down* on it
`a:focus`	when anchor gains *keyboard focus*

CSS – **pseudo-class** cascade

- Anchor pseudo-class selectors must always be written in the stylesheet in this order **lvha** – **l**ink, **v**isited, **h**over, **a**ctive
- This is because the lower level, more specific CSS properties over-ride the properties in the rules above – the **cascade**
- If the pseudo-class rules are written in a different order the styles just aren't applied correctly
- To test link styles interact with them in the browser. Turn off browser history so that visited link styles do not persist.

`normalize.css` style sheet

- Different makes of browser and old versions of browsers may not recognise HTML5 elements, or render them in slightly different ways
- The solution is to use '**`normalize.css`**'
- A CSS file that makes browsers render all elements more consistently and in line with modern standards
- It precisely targets only the styles that need normalizing

normalize.css style sheet

To use normalize.css

1. Make a text file and name it normalize.css; save it in your css folder
2. Go to https://necolas.github.io/normalize.css/
3. Click the 'Download v8.0.0' button; copy the css and paste it into your normalize.css file
4. Link the file in the <head> of ALL your HTML pages **BEFORE** your own style sheet

```
<link href="css/normalize.css" rel="stylesheet">
<link href="css/stylesheet.css" rel="stylesheet">
```

- Once you have linked normalize.css **leave it alone** - don't edit it in any way

This week's lab tutorial

- Please finish tutorials 1, 2 and 3 in your own time
- You need a learning journal page with a range of HTML elements *and content* to target with CSS selectors before moving on to …
- **Tutorial 4** – introduction to style sheets - writing rules with the selectors covered in this lecture

http://jh1033.brighton.domains/ci435/tutorials/tutorial04.html

Reading and links

- Jon Duckett, *HTML & CSS: design and build websites.* Read Chapter 10 (Introducing CSS)
- MDN CSS Reference - https://developer.mozilla.org/en-US/docs/Web/CSS/Reference
- MDN Learn to style using CSS - https://developer.mozilla.org/en-US/docs/Learn/CSS
- MDN Simple selectors- https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS/Simple_selectors
- MDN The cascade and inheritance - https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS/Cascade_and_inheritance
- Studholme, O., 2013. *CSS: reset or normalize?* https://the-pastry-box-project.net/oli-studholme/2013-june-3