# Comp 125 - Visual Information Processing

Spring Semester 2018 - week 10 - monday

Dr Nick Hayward

# Fun exercise - using CSS to style HTML

using the provided HTML5 template, `index.html`

- create a new directory for your application
  - *e.g.* `fun-application`
- add the HTML file to this application directory
- create a new CSS stylesheet for this application
  - *e.g.* `style.css`
- add the following to the application's HTML file
  - *a* `<link>` *to your CSS stylesheet*
  - *a* `<title>` *for the application*
  - *any other suitable metadata*
- then, add the following structure to this HTML page
  - *header and heading*
  - *image and title - any image is fine...*
  - *brief information about the image*
  - *a link to further information on the image*
    - e.g. a photo of the *Great Pyramid* might link to an article on Wikipedia &c.
  - *a* `<table>` *with data on the image, its content &c.*
  - `<footer>` *for information about the site, its developer, copyright information &c.*
- add appropriate styling using the CSS stylesheet, e.g.
  - *font colour, font size, font-family &c.*
  - *correct use of the box model for the content, e.g.*
    - image and title
    - any textual data
    - links
  - *custom styling for general structure and aesthetics*
  - *aesthetics and design are your own choice*

Assignment must be submitted by Monday 26th March 2018 at 2.30pm using either GitHub or a private message to the course TA, Catherine, on Slack.

**n.b.** correct use of semantic elements is required.

# Fun exercise - using CSS to style HTML

***sample HTML5 template***

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Demo 1</title>
  </head>
  <body>
    <h3>Our first web page</h3>

  </body>
</html>
```

# CSS Basics - cascading rules - part 1

- CSS, or cascading style sheets, employs a set of **cascading** rules
- rules applied by each browser as a ruleset conflict arises
  - *e.g. issue of **specificity***

```
p {
  color: blue;
  }
p.p1 {
  color: red;
  }
```

- the more specific rule, the class, will take precedence
- issue of possible duplication in rulesets

```
h3 {
  color: black;
}

h3 {
  color: blue;
}
```

- **cascading** rules state the later ruleset will be the one applied
  - *blue heading instead of black...*

# CSS Basics - cascading rules - part 2

- simple styling and rulesets can quickly become compounded and complicated
- different styles, in different places, can interact in complex ways
- a powerful feature of CSS
  - *can also create issues with logic, maintenance, and design*
- three primary sources of style information that form this cascade

1. default styles applied by the browser for a given markup language
   - *e.g. colours for links, size of headings...*
2. styles specific to the current user of the document
   - *often affected by browser settings, device, mode...*
3. styles linked to the document by the designer
   - *external file, embedded, and as inline styles per element*

# CSS Basics - cascading rules - part 3

- basic cascading nature creates the following pattern
  - *browser's style will be default*
  - *user's style will modify the browser's default style*
  - *styles of the document's designer modify the styles further*

# CSS Basics - inheritance

- CSS includes inheritance for its styles
- descendants will inherit properties from their ancestors
- style an element
  - *descendants of that element within the DOM inherit that style*

```
body {
  background: blue;
}

p {
  color: white;
}
```

- p is a descendant of body in the DOM
  - *inherits background colour of the body*
- this characteristic of CSS is an important feature
  - *helps to reduce redundancy and repetition of styles*
- useful to maintain outline of document's DOM structure
- most styles follow this pattern but not all
- margin, padding, and border rules for block-level elements **not inherited**

# CSS Basics - fonts - part 1

- fonts can be set for the body or within an element's specific ruleset
- we need to specify our font-family,

```
body {
  font-family: "Times New Roman", Georgia, Serif;
}
```

- value for the font-family property specifies preferred and fall-back fonts
  - *Times New Roman, then the browser will try Georgia and Serif*
  - *" " - quotation marks for names with spaces...*

**n.b.** " " added due to CSS validator requesting this standard - it's believed to be a legacy error with the validator...

# CSS Basics - fonts - part 2

- useful to be able to modify the size of our fonts as well

```
body {
  font-size: 100%;
}

h3 {
  font-size: x-large;
}

p {
  font-size: larger;
}

p.p1 {
  font-size: 1.1em;
}
```

- set base font size to 100% of font size for a user's web browser
- scale our other fonts relative to this base size
  - *CSS absolute size values, such as `x-large`*
  - *font sizes relative to the current context, such as `larger`*
  - *`em` are meta-units, which represent a multiplier on the current font-size*
  - *relative to current element for required font size*
  - *1.5em of 12px is effective 18px*
- `em` font-size scales according to the base font size
  - *modify base font-size, `em` sizes adjust*
- try different examples at
  - *W3 Schools - font-size*

# Demo - CSS Fonts

- Demo - CSS Fonts
- JSFiddle - CSS Fonts

# CSS Basics - fonts - part 3

- `rem` unit for font sizes
- size calculated against root of document

```
body {
  font-size: 100%;
}

p {
  font-size: 1.5rem;
}
```

- element font-size will be `root size * rem size`
  - *e.g. body font-size is currently 16px*
  - *rem will be `16 * 1.5`*

## References

- MDN - CSS
- CSS documentation
- cascade and inheritance
- fonts
- W3Schools - CSS
- CSS
- fonts