CSS Essentials

Getting started with Cascading Style Sheets

JISC netskills

Topics

- HTML Formatting – why not?
- CSS Essentials
  - Rules, Selectors and Declarations
  - Adding style to documents
- CSS Cascade
  - Selectors & specificity
- Units & Colours

Evolution of HTML Formatting

- (X)HTML only for structuring content
  - Specification only contains *guidelines* for visual browsers

- Some tags/attributes added for visual formatting

```
<font face="Arial" color="red">Hello</font>
```

→ Hello

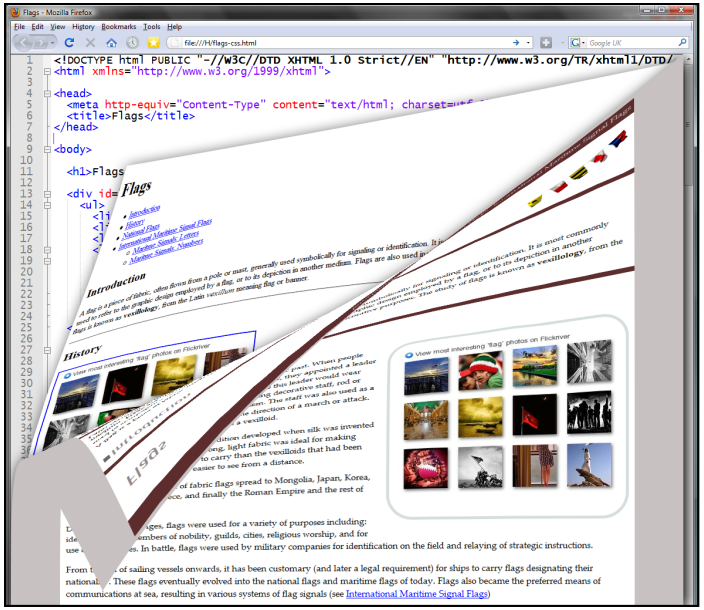
- This mixes style and structure
  - Often using proprietary mark up with limitations on what can be applied

JISC netskills

The Solution: CSS

- **C**ascading **S**tyle **S**heets
  - Separation of style from structure
  - Control – potentially over every item in the page
  - Easier style management

- *Strict* XHTML (and HTML 4.01) *deprecated* HTML formatting in favour of CSS



JISC netskills

Same content... different view

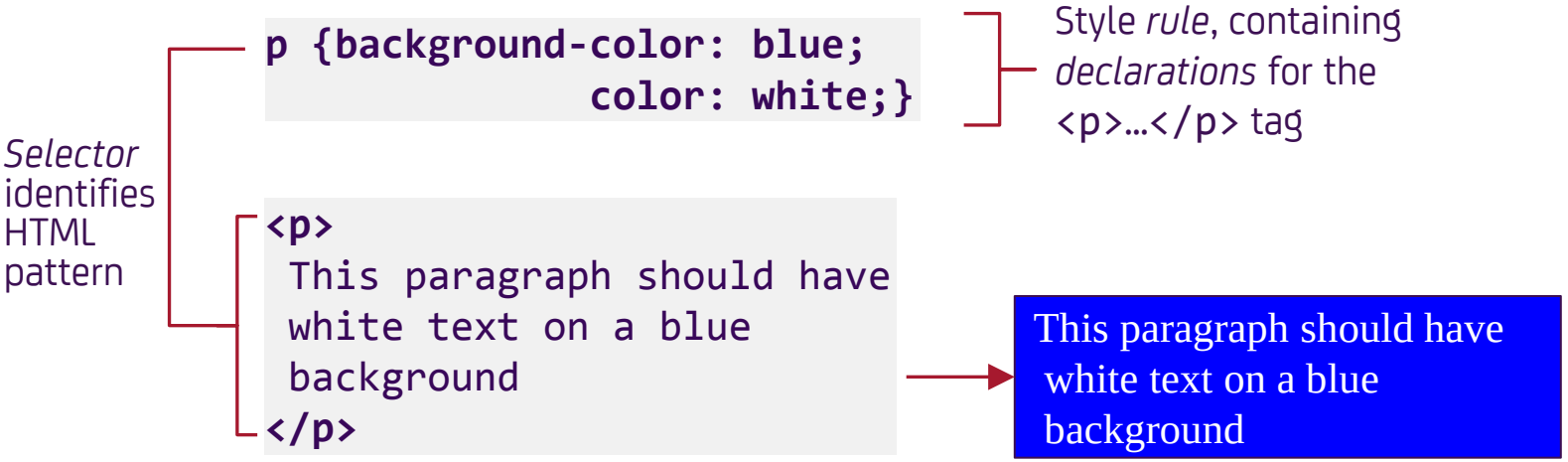
http://www.csszengarden.com

http://www.mezzoblue.com/zengarden/alldesigns/

JISC netskills

CSS Style Sheets

- Style sheets specify formatting *rules*
- Rules consist of *selectors* and *declarations*



JISC netskills

Basic Style Sheet Syntax

Declaration(s) defined inside curly braces as `style-property: value;`

Selector → `p {background-color: blue;}`

Semi-colon `;` separates declarations

```
ul {margin-left: 15%; font-weight: bold;}
```

Multiple selectors as comma separated list

"Apply declarations to h1 and h2 and h3 and h4"

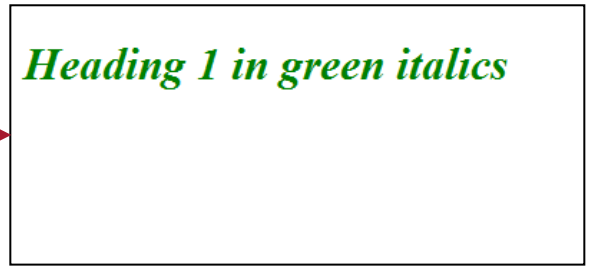
```
h1,h2,h3,h4 {background-color: white;
                        color: blue;
                   font-style: italic;}
```

JISC netskills

Internal Style Sheets

- Rules set out in `<style>` tags in the `<head>` section of the page

```
<html>
<head>
 <title>Internal Example</title>
 <style type="text/css">
  h1 {color: green; font-style: italic;}
 </style>
</head>
<body>
 <h1>Heading 1 in green italics</h1>
</body>
</html>
```



JISC netskills

External Style Sheets

- Style sheets are stored in separate files
  - Linked to current document
  - Multiple style sheets can be linked to a single page

```
<html>
<head>
 <title>CSS example</title>
 <link rel="stylesheet" type="text/css" href="mystyle.css" />
</head>
<body>
 <h1>Heading 1 in green italics</h1>
</body>
</html>
```

JISC netskills

Using @import Rules

- Alternative way to include external style sheets

```
<style type="text/css">
 @import url("styles.css");
</style>
```

- No difference in effect or behaviour, but can be more convenient
  - Only need one hard-coded `<link>` in XHTML document
  - Style sheets can be edited/attached/renamed without touching XHTML document

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

```
@import url("default.css");
@import url("navbar.css");
@import url("print.css");
```

Single linked style sheet used to import actual styles from separate files

JISC netskills

Inline Styles

- Style can also be added *inline*
  - Uses `style` attribute with CSS rule(s) as value

```
<p style="color:white; background-color: blue;">Hello</p>
```

Hello

- Try and avoid if possible – mixes style and structure back up
- Can be a useful option if needed to overcome a *specificity* issue

JISC netskills

More on CSS Selectors

- Three basic selector types define patterns to find in the mark-up

  Tag – match all instances of the tag e.g. every `<p>...</p>`
  Class – match tags containing this `class` attribute
  Id – match the unique tag containing this `id` attribute

- Can be combined for more *specific* matches
- Additional syntax and operators allow precise control
- Combine with `<div>` and `<span>` to build a *framework* for display

JISC netskills

Classes as Selectors

- Used to apply styles to specific sub-sets of HTML tags
  - Tags are grouped using a **`class`** attribute
  - Tags can be in more than one class

```
<h1 class="special">A heading</h1>
<p>This is a normal paragraph</p>
<p class="special">A different class of paragraph</p>
```

- Define style rule(s) in the style sheet

```
p {text-align: left; color: red;}
.special {text-transform: uppercase;}
p.special {text-align: right; color: green;}
h1.special {text-decoration: underline;}
```

Dot (`.`) in selector pattern indicates a class e.g.

`p.special`

matches

`<p class="special">`

JISC netskills

ID as a Selector

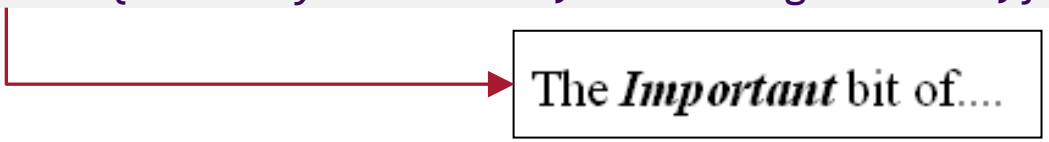
- Used to identify *unique* elements in the page
  - Uses an `id` attribute in the tag
  - Each `id` value can only be used *once* in any page (same `id` can be used on multiple pages though)

```
<p>The <span id="oneoff">Important</span> bit of…</p>
```

`oneoff` now provides a unique `id` for a single element in this document

- Hash (#) in the CSS selector pattern indicates an `id`

```
#oneoff {font-style: italic; font-weight: bold;}
```



JISC netskills

More Selector Syntax

Selector	Pattern matched
`p`	All `<p>`
`.special`	`<anytag class="special">`
`p.special`	All `<p class="special">`
`#thisBox`	The only `<anytag id="thisBox">`
`#thisBox p`	All `<p>` nested *anywhere* inside the only `<anytag id="thisBox">`
`#thisBox > p`	All `<p>` that are *direct children* of `<anytag id="thisBox">`
`#thisBox p.special`	All `<p class="special">` nested *anywhere* inside the *only* tag with the `id` of `thisBox`
`div#thisBox p`	All `<p>` nested *anywhere* inside the *only* `<div id="thisBox">`

http://www.w3.org/TR/CSS2/selector.html

JISC netskills

Combining Selectors

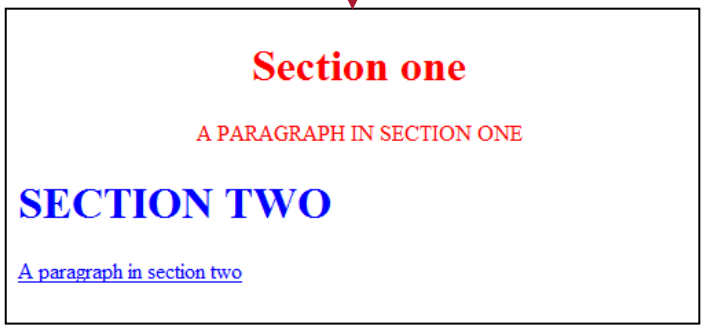
CSS Rules

```
#section1 {color:red; text-align:center;}
#section2 {color:blue;}
.caps {text-transform:uppercase;}
#section2 p {text-decoration:underline;}
```

HTML

```
<div id="section1">
 <h1>Section one</h1>
 <p class="caps">A paragraph in
   section one</p>
</div>
<div id="section2">
 <h1 class="caps">Section two</h1>
 <p>A paragraph in section two</p>
</div>
```

`#section2 p` styles only applied to `<p>…</p>` nested inside `#section2`

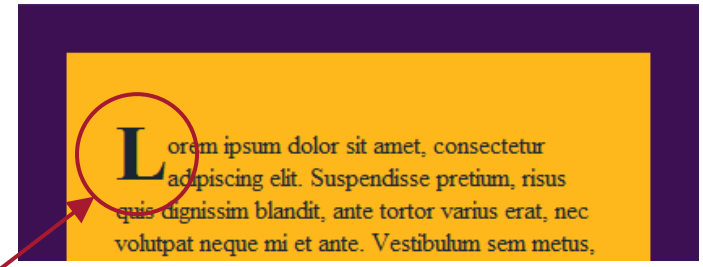


JISC netskills

Pseudo Elements

Selectors for special parts of some elements

```
p.opening:first-letter {
        font-size: 300%;
        font-weight: bold;
        float: left; }
```




```
<p class="opening">Lorem ipsum dolor sit amet,
consectetur adipiscing elit. Suspendisse pretium,
risus quis dignissim blandit, ante tortor... etc
</p>
```

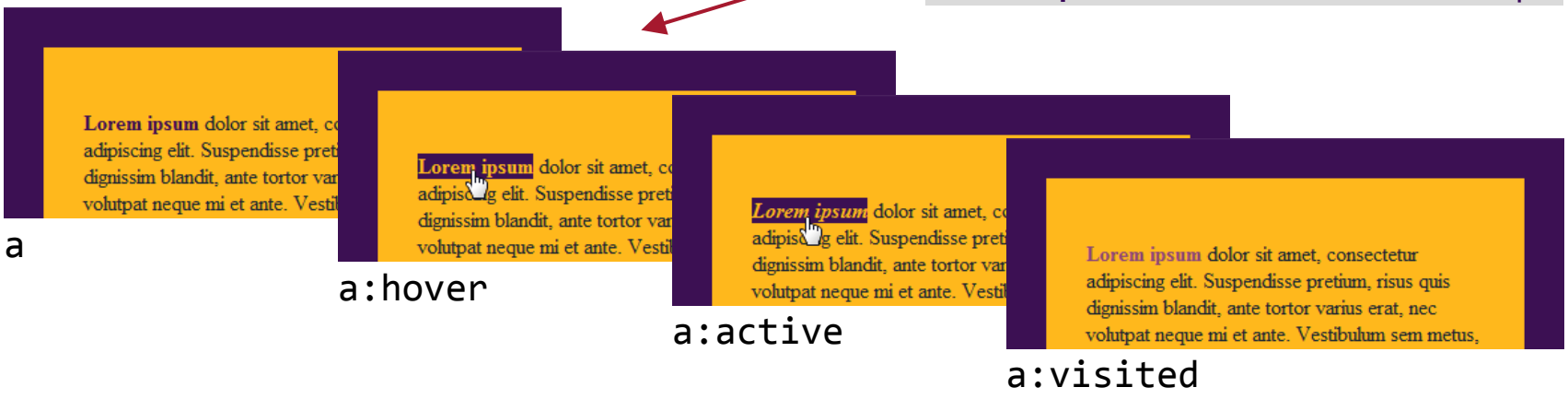
JISC netskills

Pseudo Classes

Selectors for special status of some elements

```
a           {text-decoration: none; font-weight: bold; color: #3c1053;}
a:hover     {background-color: #3c1053; color: #ffb81c;}
a:active    {font-style: italic;}
a:visited   {color: #8c4779;}
```

```
<p>
<a href="http://lipsum.com">
Lorem ipsum</a> dolor... etc </p>
```



Cascading Style Sheets

- All available styles for a page are combined as it loads
  - Final appearance for each element is composite of all appropriate rules
- Conflicting property values resolved by simple rules
  1. **Source:** User-specified styles (in the browser) are more specific
  2. **Specificity:** Relative weighting of selector priority
  3. **Order declared**: If specificity value are the same then "last one wins" (means inline styles are always more specific)
- Specificity – a measure if importance
  - The more *specific* the rule is the greater priority its declarations have
  - Easy to calculate...

JISC netskills

Specificity Calculator

Count the number of ID, class and tag names in each selector

Selector	IDs	Classes	Tags
`#thisbox`	1	0	0
`.special`	0	1	0
`p`	0	0	1
`p.special`	0	1	1
`#thisbox p`	1	0	1
`#thisBox > p`	1	0	1
`div#thisBox p`	1	0	2
`#thisBox p.special`	1	1	1

Basic values of:
ID = 100
Class = 10
Tag = 1

More *specific* combinations have higher values

JISC netskills

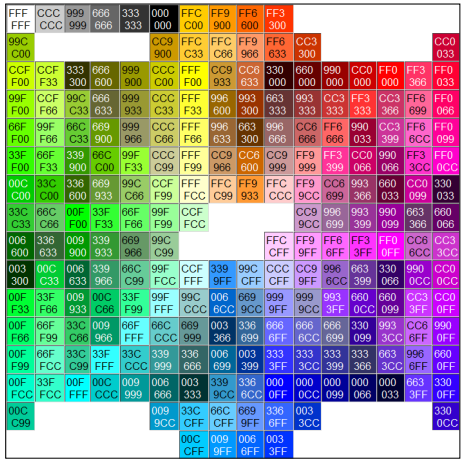
CSS Units

- CSS supports many types of measurement unit
- **Absolute** units calculated independently of other page content and/or browser defaults
  - Useful for precise layout
  - Include Pixels (`px`), Points (`pt`), Millimetres (`mm`)
- **Relative** units calculated proportionally against other page content or a browser default
  - e.g. currently available width, default text size etc.
  - Include Percentages (`%`), Ems (`em`), Exes (`ex`)
  - Also special relative units for text e.g. `small`, `large`, `x-large` ... etc.
- Good design uses a combination of both

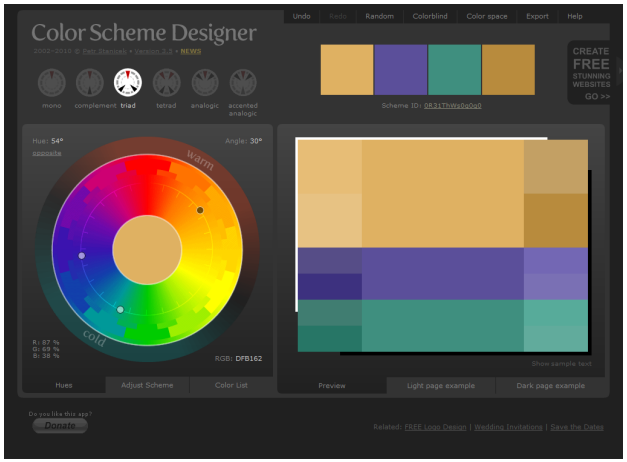
JISC netskills

CSS Colours

- CSS allows rich control over colour (color!)
  - *Any* colour can be specified using RGB or Hex (hexadecimal) codes
  - Only 17 *names* are actually valid in CSS2



```
color: red;          ✓
color: magenta;      ✗
color: rgb(0,32,234); ✓
color: #0000ff;      ✓
```



http://www.w3.org/MarkUp/Guide/Style.html

http://colorschemedesigner.com/