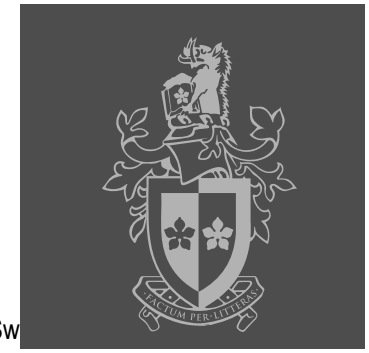
SWINBURNE UNIVERSITY OF TECHNOLOGY

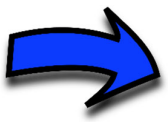
COS10026
Computing Technology Inquiry Project

Lecture 4

CSS 1 - Presentation and CSS Selectors

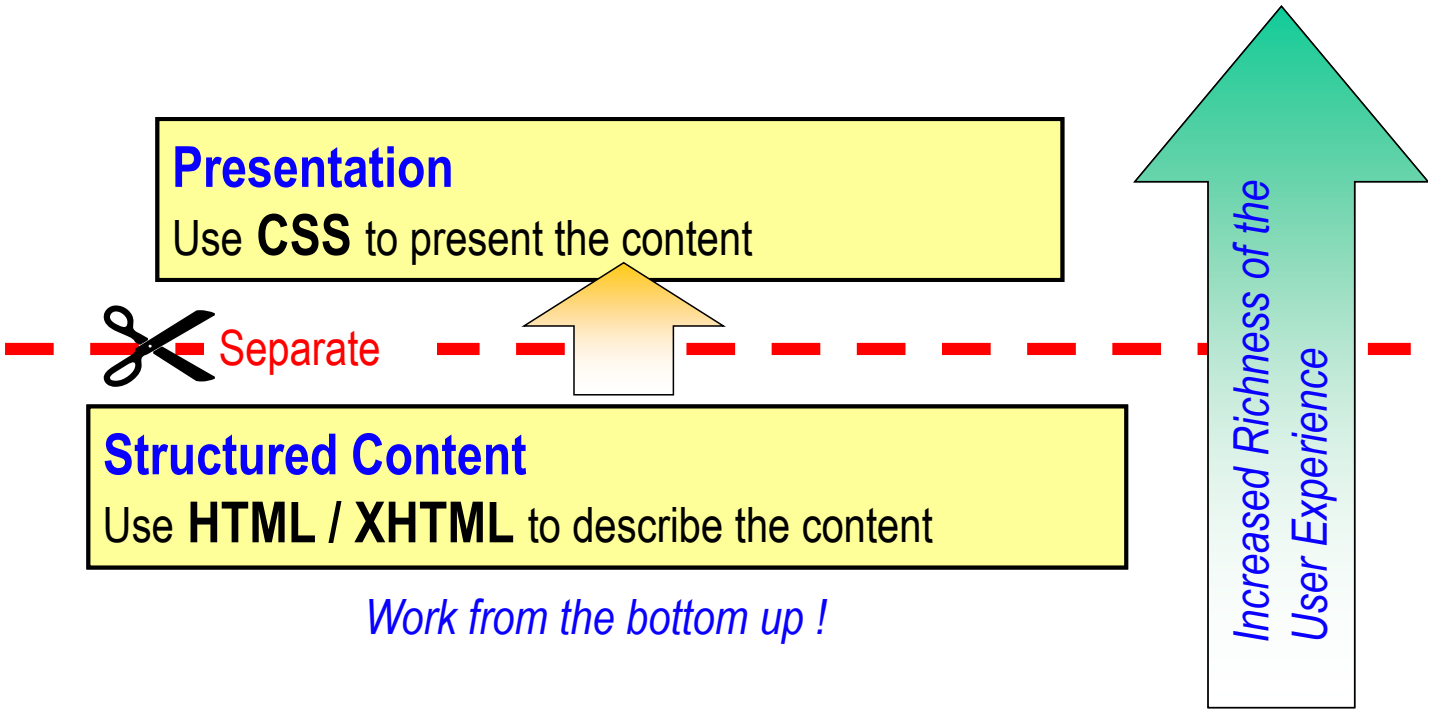


Topics



- What is CSS?
- Linking CSS to HTML
- CSS Selectors

Review: Separate content from presentation

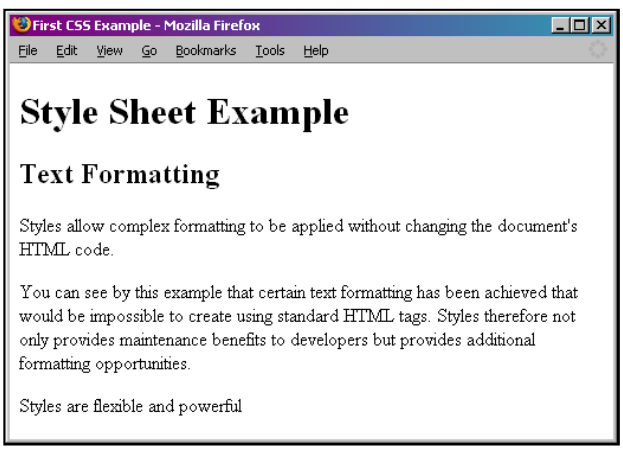


What is CSS?

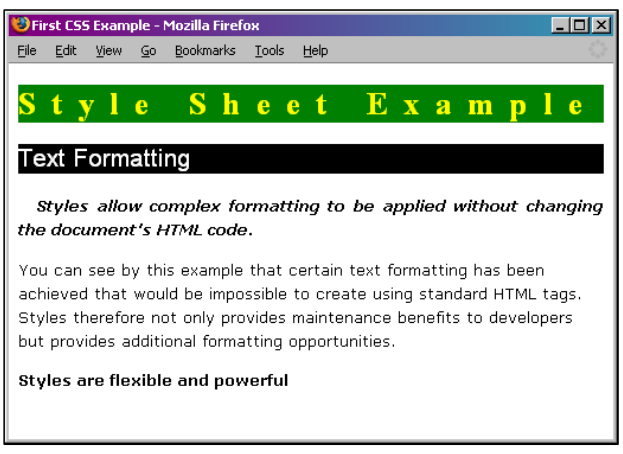
- CSS is the language we use to style an HTML document.
- CSS describes how HTML elements should be displayed.

For example

no CSS



with CSS

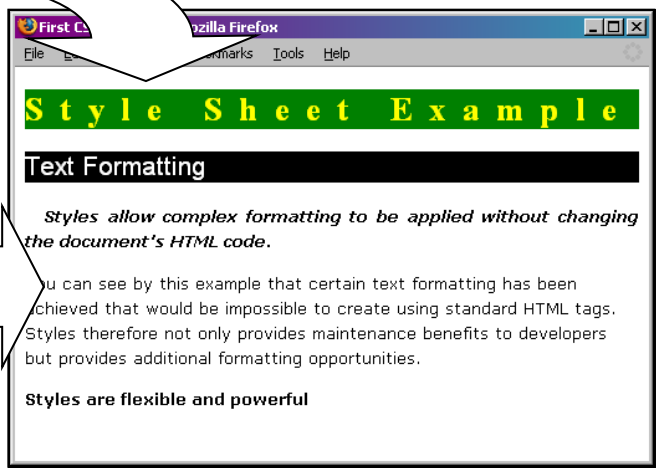


First CSS Example

Remember the simple structure of HTML documents!

```
<!DOCTYPE ...>
<html>
  <head>
    <title>...</title>

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

  </head>

  <body>

    <!-- body content here -->

  </body>
</html>
```

One external stylesheet can be linked and can be re-used for many webpages



Style Sheet Basics

- **Style sheets** contain a collection of "**style rules**"
- **Style rules** start with a `selector` and then contain `properties` and `values`.

You need to know this terminology, so you can talk with other web developers.

```
selector { property1: value1; property2: value2; …}
```

CSS Rule

e.g. `h1 {color: blue; font-size: 1em; }`

A **selector** identifies the **markup elements** that the style property values will be applied to. *eg. element, class, id*

More about the range of selectors later ...

First CSS Example

HTML Structure

```
...
<h1>Style Sheet Example</h1>
<h2>Text Formatting</h2>
<p class="intro">Styles allow
  complex formatting to be applied
  without changing the document's
  HTML code.</p>
<p class="intro">You can see by
  this example that certain text
  formatting has been achieved that
  would be impossible to create
  using standard HTML tags.  Styles
  therefore not only provides
  maintenance benefits to
  developers but provides
  additional formatting
  opportunities.</p>
<p id="keypt">Styles are flexible
  and powerful</p>
...
```

Here both style rules also "inherit" the `p` *element style*

CSS Presentation

```
h1 { letter-spacing: .5em;
     background-color: green;
     color: yellow;
     font-size: 20pt;
     font-family: serif; }

h2 { color: white;
     background: black;
     font: normal 16pt Arial,
     sans-serif; }

p { line-height: 15pt;
    font-size: 10pt;
    font-family: "Verdana",
    sans-serif; }

.intro { text-align: justify;
         text-indent: 12pt;
         font-style: italic;
         font-weight: bold; }

#keypt { font-weight: bold; }
```

element styles

class style

id style

CSS: Quick Start Style Rule Examples

```
h1, h2     { font-family: sans-serif; }
th         { color: #3366CC;}
div, p     { border: 1px solid #FF0000; }
a:hover    { font-weight: bold; }
li         { font-size: 12px; }
a          { text-decoration: underline overline; }
h3         { border-bottom: 2px dashed green; }
p          { text-align: justify; }
p.indent   { text-indent: 20px; }
.upper     { text-transform: uppercase; }
img        { float: right; }
ol         { list-style-type: upper-roman; }
```

```
selector { property1: value1;}
```

CSS Rule

Topics

- What is CSS?
- Linking CSS to HTML
- CSS Selectors

Style Sheet Basics

- Style sheet information can be stored in either:
  - A separate ***external*** CSS file, linked with a `link` element (in the `head` element)

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

and / or

  - an ***embedded*** style sheet within a `style` element (in the `head` element)

e.g.

```
<style type= "text/css">
        h1 {color : blue;}
</style>
```

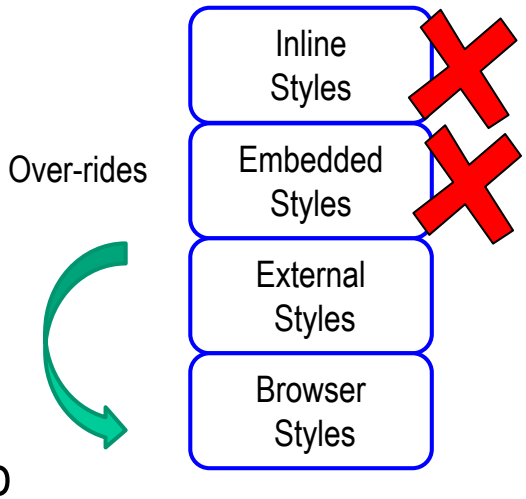
and / or

  - using ***inline*** style with a `style` attribute within ***any*** element (as a core attribute)

e.g. `<h1 style = "color : blue;" >`

CSS: Methods of Incorporating CSS

- **Inline** – coded as an attribute
- **Embedded** – defined in the head section (last defined takes precedence)
- **External** – coded in a separate file
- **Imported** – similar to external, but allows a style to import another style



Rules of precedence in applying styles

CSS1, CSS 2.1, and CSS3 http://www.w3.org/TR/CSS/

- **CSS1** introduced CSS (now superseded by CSS 2.1)
- **CSS2.1** Now largely fully supported by most modern browsers. CSS2.1 was a revision of CCS2.

- **CSS3** builds on CSS2 *module by module*, using the CSS2.1 spec. as its core.
  eg.CSS3 Selectors, CSS3 Colors, CSS3 Media Queries, etc.

Each module is in a different stage of development
(eg. CSS3 Selectors fully developed and supported by most browsers)
CSS3 is being quickly adopted and becoming 'mainstream'.

- **CSS4** modules are being developed as new needs arise.

For current CSS status see: http://www.w3.org/Style/CSS/current-work.html

Validating CSS

- W3C CSS validator

http://jigsaw.w3.org/css-validator/

- Exercise in lab!

Writing CSS Comments

- Comments are enclosed in /* … */
- For example

```
/*
  Typography styles
*/
article {
  color : blue;
}
p {font-family: Arial, Helvetica, sans-serif;
}
```

In your assignment (project) you must have

- **Header comments on your CSS**
- **Line comments on any rules whose application is not obvious**
- **Comments acknowledging sources of any 3rd party CSS**

Topics

- What is CSS?
- Linking CSS to HTML
- CSS Selectors

CSS Selectors

- **CSS1** introduced the initial set of selectors. Supporting:
  - rules for **element** types, specific **id** values, generic **classes**
  - **grouping** and **contextual** selection of rules *(combinators)*
  - some **pseudo classes**
- **CSS2** added several new selector types. Allowing:
  - more **power** and **control** over rule application.
  - element **content** to control rule application.
- **CSS3** provides improved context, including different xmlns

  See overall summary CSS1-CSS3: http://www.w3.org/TR/css3-selectors/#selectors
- **CSS4** evolving additional selectors as user interfaces change

  https://www.w3.org/TR/selectors4/

CSS1 Selectors

■ CSS1 Selectors

Selector	Description	Example
`element`	Applies the style rule to *all elements* that match the **element name.** *Also called "**tag style**"*	`h1 {color: green;}`

`#id`	Applies the rule *only for the **single** element* that has this **id value.**
	eg. <tag id="info">
	*Also called "**id style**"*
		`#info {`
		`  background-color: red;`
		`}`

`.class`	Applies the rule to *any elements* that have the matching **class value.** eg. <<tag class="note"> *Also called "**class style**"*	`.note {color: blue;}`
`element.class`	Applies the rule ***only*** to elements with the specified **element name** that ***also*** have the matching **class value.** eg.<p class="note"> *Also called "**tag specific class**"*	`p.note {` `  border: 1px solid blue;}`

CSS - Basics | Selectors | Properties

CSS1 Selectors

■ CSS1 Selectors - Grouped & Contextual *("combinators")*

Selector	Description	Example

Grouping	Applies the rule to a ***group of selectors,*** (separated by ***commas***)	`h1, h2, p {font: sans-serif;}`
		`header, nav`
		`  {border-style : dotted;}`

Contextual	*Also called* **Descendant combinator**
	Applies the rule to the descendant (contained or 'nested') elements. (separated by ***spaces***)
	Refer to element hierarchy and inheritance concepts.
		`div h1 {`
		
		`}`

Note: *if any one of the selectors is invalid, the whole group may be ignored ☹*

© Swinburne University of Technology

Selectors - Pseudo Classes

The pseudo-class concept was introduced to permit selection based on information that lies outside of the document tree ***or*** *that cannot be expressed using the other simple selectors.*

Selector	Description	Example
`a:link`	An **unvisited hypertext** link	`a:link {color: blue;}`
`a:visited`	A **link** that has already been **visited**	`a:visited { background-color: yellow; }`
`a:active`	An **active link** (as it is being 'clicked')	`a:active {color: red;}`

CSS Selectors - Dynamic Pseudo Classes

- CSS2 examples

Selector	Description	Example
`:hover`	Applied when the browser "cursor" is **hovering** over an element. (similar to a "mouseover" event)	`a:hover {font-weight: bold;}` `p:hover { border: 1px solid red; }` **Demo**
`:focus`	Applies when an element receives "**focus**" – commonly used with form elements like `<input … />`.	`input:focus { background-color: white; }` **Demo**
`:active`	Applies while an element is being **activated** by the user. (eg, the time between when a user presses the mouse button and releases it.)	`#mybutton:active { color: red; }`

CSS Selectors - Pseudo Elements

- Pseudo-elements selects aspects of a document that are not classified by elements

Selector	Description	Example
`:first-line` `::first-line`	The **first line** of content (text) contained within the selected **element** (acts as a pseudo element)	`p::first-line { font-weight: bold; }` Demo from W3C
`:first-letter` `::first-letter`	Treats the very **first character** (letter) of element content as a separate pseudo **element** and applies the rule.	`p::first-letter { color: red; font-size: 150%; }` Demo from W3C

Pseudo Elements format has changed
`:first-line` CSS2,
`::first-line` CSS3.

CSS Selectors

- **CSS2** Selectors

Selector	Description	Example
*	**Wildcard** or **universal** selector, used to apply a rule to any element, ***or*** contextually, any element within a parent element . ie. as a **descendant combinator**	`* { background-color: red; }` `div * span { background-color: blue; }`
>	**Child combinator** Match a **directly** enclosed child element (eg. **only** body > p ***not*** body > div > p)	`body > p { font-size: 12pt; }`
+	**Adjacent sibling combinator** Match an **adjacent sibling** element, (eg. first paragraph following a level 2 heading)	`h2 + p { color: blue; }`

[]	The **attribute** selector. ***Very powerful!***
	= for an exact match,
	~= for partial matches,
	|= for an item in a space separated list
		`a[href] { color: green;}`
		`a[href~="http://"] { … }`
		`p[lang|="en"] { … }`

CSS2 Selectors

CSS2 Selectors - Pseudo Classes

Selector	Description	Example
`:first-child`	Match the **first child** contained in an element.	`p:first-child { color: blue; }`
`:lang`	**Language** dependent style application.	`*:lang(fr) { color: blue; }` `*:lang(en) { color: green; }`

CSS2-3 Selectors - Pseudo Elements

Selector	Description	Example
`:before` `::before`	Place content **before** an element	`div::before { content: url(header.gif); }`
`:after` `::after`	Place content **after** an element	`div::after { content: url(footer.gif); }`

CSS3 Selectors

- CSS3 has introduced a wide range of powerful selectors

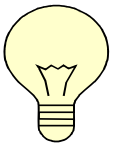
  e.g string selectors, more pseudo-classes, …

Provides **very** powerful access to objects, eg. third row of a table

- Now widely supported by most browsers

Cascading: Hierarchy and Inheritance

- CSS is applied to the ***HTML document structure***.
- Some style properties that are applied to a "**parent**" element will be **inherited** by its "**children**" elements.
- Not all style properties are inherited by children ...
  - ***Foreground*** properties ***are inherited*** (color, font-weight etc),
  - ***Background*** and ***layout*** properties ***are not inherited***
    (unless you specifically set them to be inherited…)



Because the default background properties of an element are usually "transparent", you will still see the parent background properties

CSS: Hierarchy and Inheritance Example

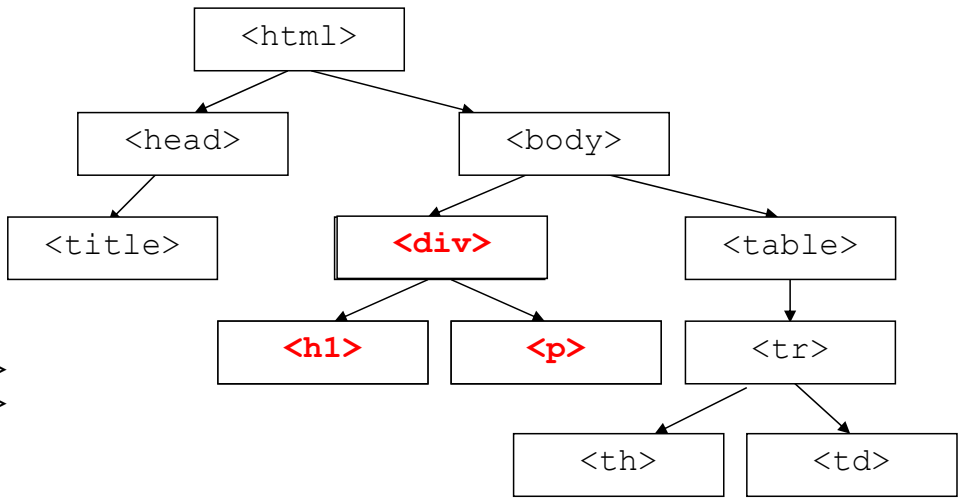
Consider the document hierarchy created in our simple HTML.

- When we apply this style rule to the document:

```
div { color: red; font-weight: bold; }
```

  - The rule will set **all** `div` elements to be a **red** foreground colour with **bold** text.
  - The **red bold** properties will be **inherited** by the child `h1` and `p` elements.

```
<html>
<head>
  <title>...</title>
</head>
<body>
  <div>
    <h1>...</h1>
    <p>...</p>
  </div>
  <table>
    <tr> <th>...</th>
         <td>...</td>
    </tr>
  </table>
</body>
</html>
```



CSS: Hierarchy and Inheritance Example

- If we specify another style rule as well:

```
div { color: red; font-weight: bold; }
h1 { color: blue; }
```

  - This will set **all** `h1` elements to the foreground colour **blue**;
  - This new rule will **override** the existing inherited **red** colour.

```
<html>
<head>
  <title>...</title>
</head>
<body>
  <div>
    <h1>...</h1>
    <p>...</p>
  </div>
  <table>
    <tr> <th>...</th>
         <td>...</td>
    </tr>
  </table>
</body>
</html>
```