

COS10011/60004 Creating Web Applications

Lecture 3

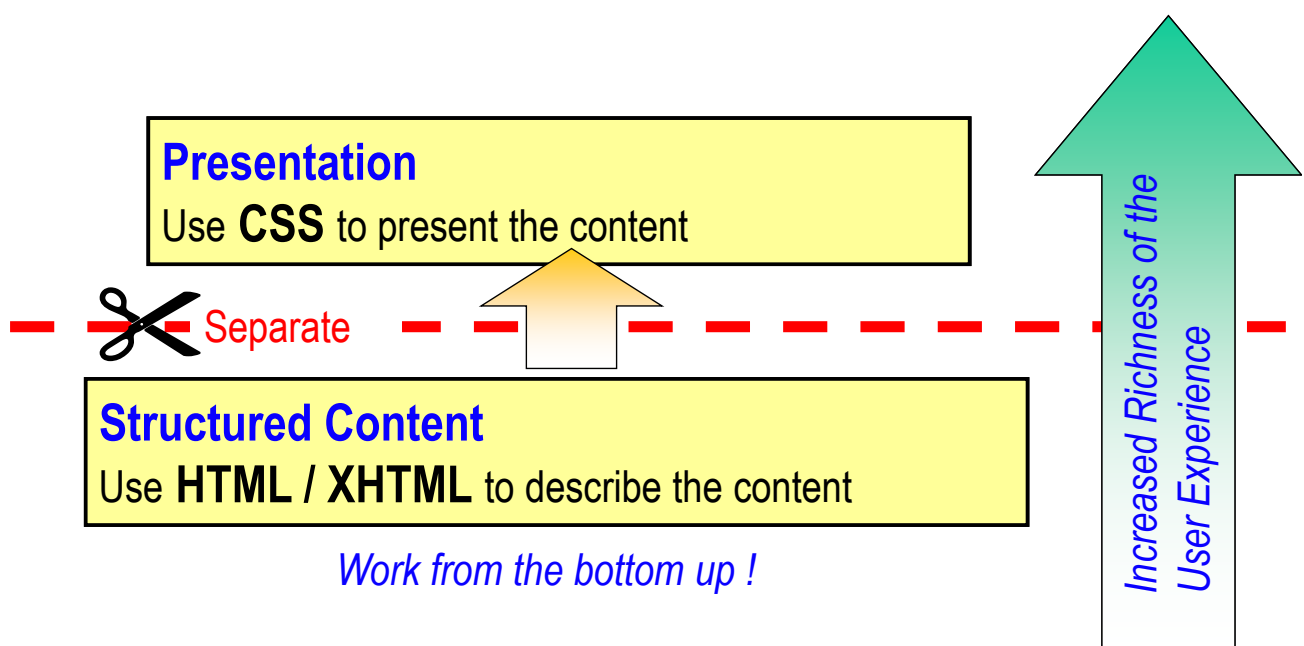
Presentation and CSS



© Sw

CSS - Basics | Selectors | Properties

Review: Separate content from presentation



CSS

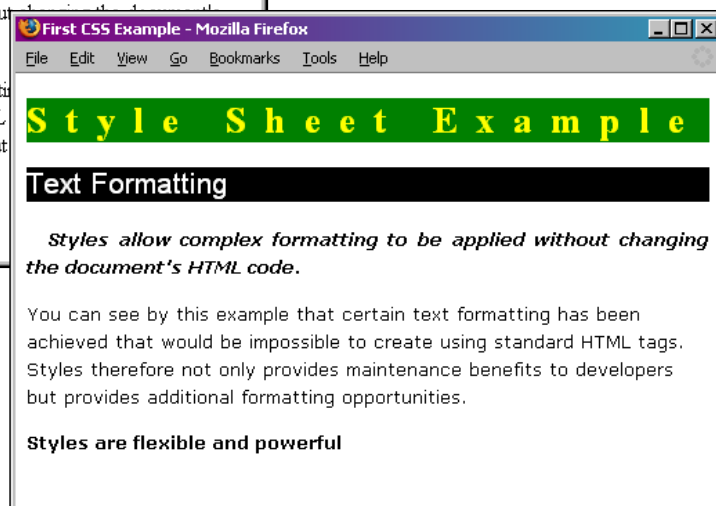
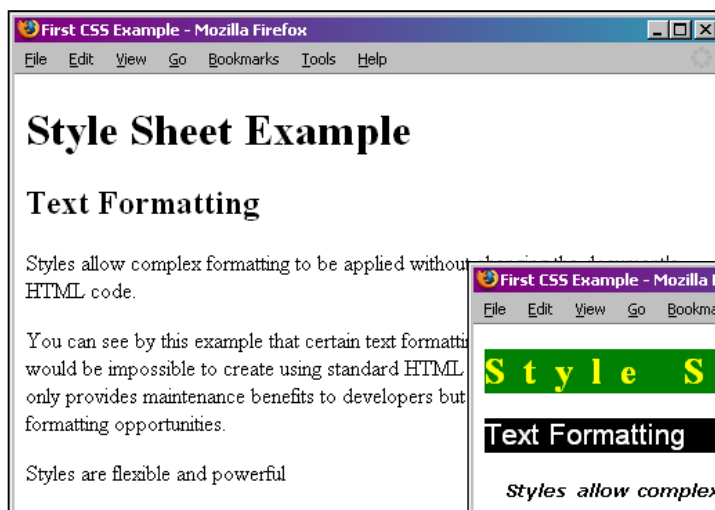
- What is CSS?
- Linking CSS to HTML
- CSS Selectors
- CSS Properties
 - ☐ Typography
 - ☐ Fonts
 - ☐ Lists
 - ☐ Size measurement
 - ☐ Colour
 - ☐ Positioning /Layout
 - ☐ Inline
 - ☐ Block Box model
 - ☐ Page Flow

Next Week ...

© Swinburne University of Technology

CSS - Basics | Selectors | Properties

First CSS Example

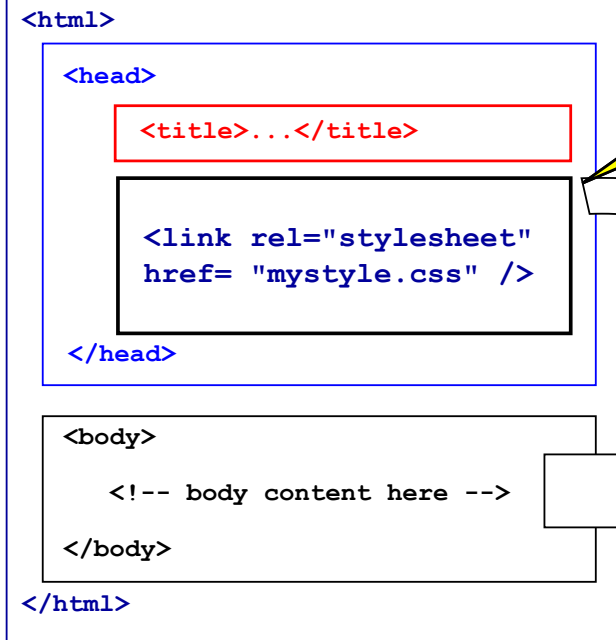


© Swinburne University of Technology

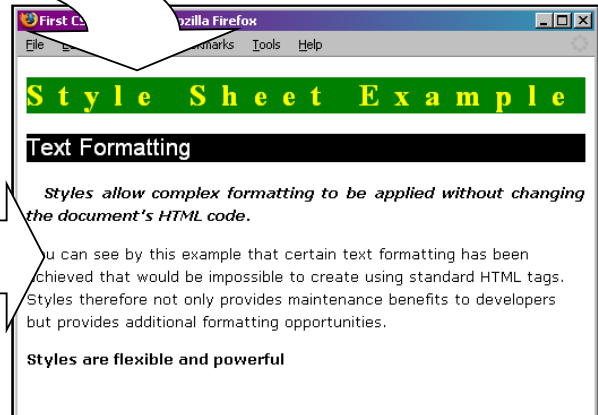
First CSS Example

Remember the simple structure of HTML documents!

```
<!DOCTYPE ...>
```



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



© Swinburne University of Technology

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

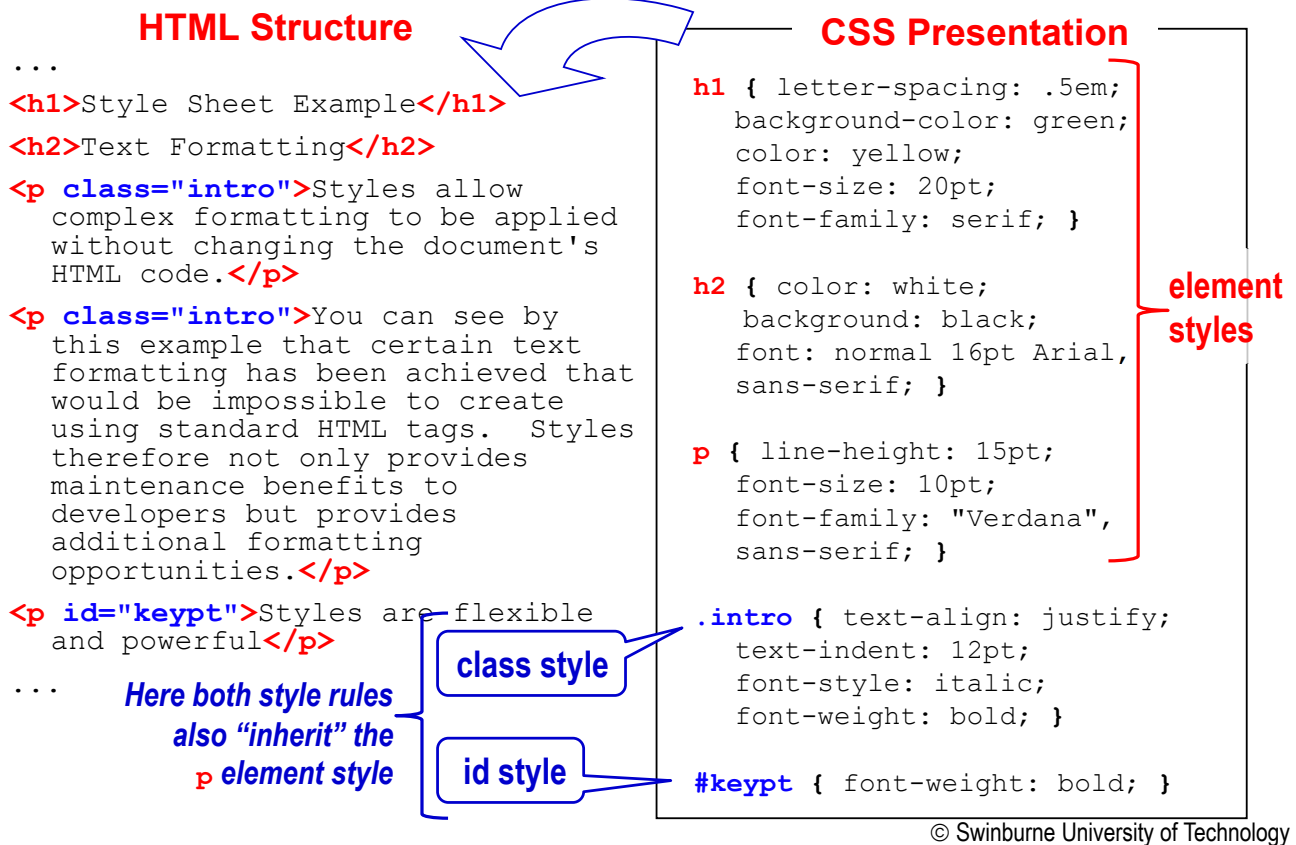
e.g. **h1** {color: blue; font-size: 20em; }

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 ...

© Swinburne University of Technology

First CSS Example



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

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;" >
```

© Swinburne University of Technology

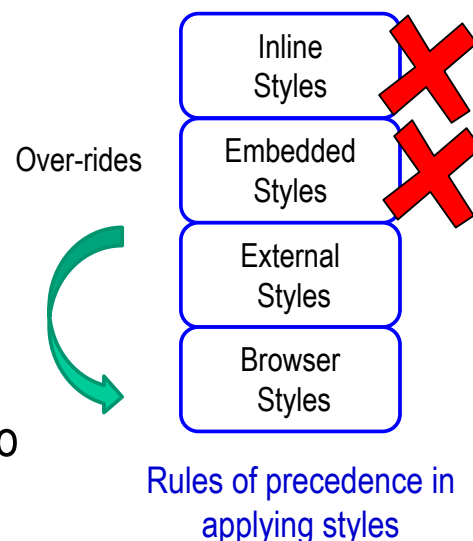
CSS: Methods of Incorporating CSS

■ 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



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 CSS2.

- **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>

© Swinburne University of Technology

Validating CSS

- W3C CSS validator

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

- Exercise in lab!

© Swinburne University of Technology

Writing CSS Comments

- Comments are enclosed in `/* ... */`

- For example

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

In your assignments 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

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 <i>all elements</i> that match the element name . <i>Also called "tag style"</i> | <code>h1 {color: green;}</code> |
| #id | Applies the rule <i>only for the single element</i> that has this id value . eg. <code><tag id="info"></code> <i>Also called "id style"</i> | <code>#info { background-color: red; }</code> |
| .class | Applies the rule to <i>any elements</i> that have the matching class value . eg. <code><tag class="note"></code> <i>Also called "class style"</i> | <code>.note {color: blue;}</code> |
| element.class | Applies the rule <i>only</i> to elements with the specified element name that <i>also</i> have the matching class value . eg. <code><p class="note"></code> <i>Also called "tag specific class"</i> | <code>p.note { border: 1px solid blue;}</code> |

© Swinburne University of Technology

CSS1 Selectors

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

| Selector | Description | Example |
|-------------------|---|---|
| Grouping | Applies the rule to a group of selectors , (separated by commas) | <code>h1, h2, p {font: sans-serif;}</code> <code>header, nav {border-style: dotted;}</code> |
| | |  Note: if any one of the selectors is invalid, the whole group may be ignored ☹ |
| Contextual | <i>Also called Descendant combinator</i> Applies the rule to the descendant (contained or 'nested') elements. (separated by spaces) Refer to element hierarchy and inheritance concepts. | <code>ol ol { list-style-type: upper-alpha; }</code> |

© 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 |
|------------------------|---|--|
| <code>a:link</code> | An unvisited hypertext link | <code>a:link {color: blue;}</code> |
| <code>a:visited</code> | A link that has already been visited | <code>a:visited { background-color: yellow; }</code> |
| <code>a:active</code> | An active link (as it is being 'clicked') | <code>a:active {color: red;}</code> |

CSS Selectors - Dynamic Pseudo Classes

■ CSS2 examples

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

CSS Selectors - Pseudo Elements

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

| Selector | Description | Example |
|---|--|--|
| <code>:first-line</code> <code>::first-line</code> | The first line of content (text) contained within the selected element (acts as a pseudo element) | <pre>p::first-line { font-weight: bold; }</pre> Demo from W3C |
| <code>:first-letter</code> <code>::first-letter</code> | Treats the very first character (letter) of element content as a separate pseudo element and applies the rule. | <pre>p::first-letter { color: red; font-size: 150%; }</pre> 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 | <pre>* { background-color: red; } div * span { background-color: blue; }</pre> |
| > | Child combinator Match a directly enclosed child element (eg. only <code>body > p</code> not <code>body > div > p</code>) | <pre>body > p { font-size: 12pt; }</pre> |
| + | Adjacent sibling combinator Match an adjacent sibling element, (eg. first paragraph following a level 2 heading) | <pre>h2 + p { color: blue; }</pre> |
| [] | The attribute selector. Very powerful! = for an exact match, ~= for partial matches, = for an item in a space separated list | <pre>a[href] { color: green; } a[href~="http://"] { ... } p[lang = "en"] { ... }</pre> |

CSS2 Selectors

■ CSS2 Selectors - Pseudo Classes

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

■ CSS2-3 Selectors - Pseudo Elements

| Selector | Description | Example |
|---|--|--|
| <code>:before</code> <code>::before</code> | Place content before an element | <code>div::before {</code> <code> content: url(header.gif); }</code> |
| <code>:after</code> <code>::after</code> | Place content after an element | <code>div::after {</code> <code> content: url(footer.gif); }</code> |

© Swinburne University of Technology

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

© Swinburne University of Technology

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

© Swinburne University of Technology

CSS - Basics | Selectors | Properties

CSS: Hierarchy and Inheritance Example

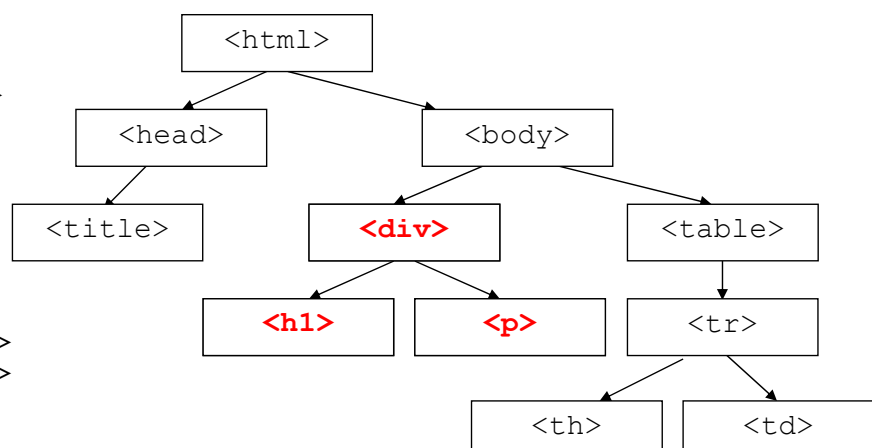
Consider the document hierarchy created in our simple HTML.

- 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>
```



© Swinburne University of Technology

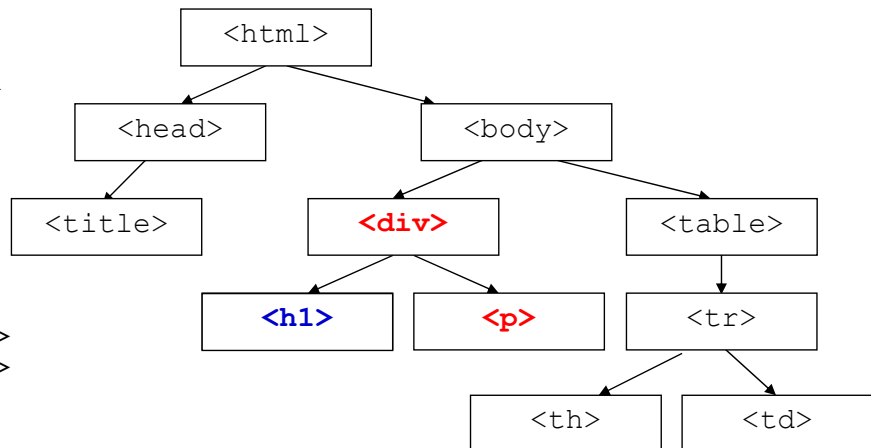
CSS: Hierarchy and Inheritance Example

- 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>
```



© Swinburne University of Technology

CSS Properties

- CSS properties define which aspect of the *selected* HTML will be changed or styled

- ☐ Size measurement
- ☐ Colour
- ☐ Typography
 - ☐ Fonts
 - ☐ Lists
- ☐ Positioning /Layout
 - ☐ Inline
 - ☐ Block Box model
 - ☐ Page Flow

CSS: Property Groups

- Animation
- **Background**
- **Border and outline**
- **Box**
- **Color**
- Content Paged Media
- Dimension
- Flexible Box
- **Font**
- Generated content
- Grid
- Hyperlink
- Linebox
- **List**
- **Margin**
- Marquee
- Multi-column
- **Padding**
- **Paged Media**
- **Positioning**
- Print
- Ruby
- Speech
- Table
- Text
- 2D/3D Transform
- Transition
- User-interface

CSS Units – Size measurement

■ **Relative** is used for styling screen webpages

e.g. **h1** { letter-spacing: .5em; }

| Unit | Abbr | Description | Example |
|------------|------|--|------------------------|
| EM | em | Height of the current font's default size | p {padding: 2em;} |
| Percentage | % | Works like em, where 100% is the default font size | p {line-height: 100%;} |
| Ex | ex | Height of letter x in the current font | p {margin: 25ex;} |
| Pixel | px | Pixel size of screen | p {font-size: 12px;} |

■ **Avoid** units *absolute* or *print* measurements:

cm, in, mm, pt

CSS Units - color

- We can specify **color**: in the following four basic ways:

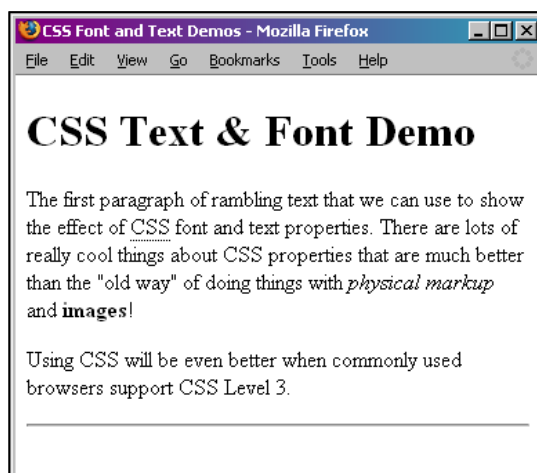
| Format | Description and Examples |
|-------------------------------------|---|
| name | Colour names . There are 16 basic colours (from the Windows VGA palette) Many others are now accepted by popular browsers, but best to use 'hex' colors. <pre>h1 {color: red} p {color: green}</pre> |
| #rrggbb or #rgb | Red, green and blue values in hexadecimal format Written in "hex" format in 6 or concise 3 character versions. Colour values between 00 and FF (or 0 and F) <pre>hr {color: #FF0000} /* red */ td {color: #00F} /* blue concise format - saves bandwidth */</pre> |
| rgb(r, g, b) | rgb (red, green, blue) values in decimal with the rgb() command. Units between 0 and 255 <pre>.info {color: rgb(255, 0, 255); } /* purple info class */</pre> |
| rgb(r%, g%, b%) | rgb (red, green, blue) values in percentage units with the rgb() command. Unit values between 0% and 100%. <pre>em {color: rgb(100%, 0%, 100%); } /* purple emphasised text */</pre> |

CSS Properties

- CSS properties define which aspect of the *selected* HTML will be changed or styled
 - ☐ Size measurement
 - ☐ Colour
 - ☐ Typography
 - ☐ Fonts
 - ☐ Lists
 - ☐ Positioning /Layout
 - ☐ Inline
 - ☐ Block Box model
 - ☐ Page Flow

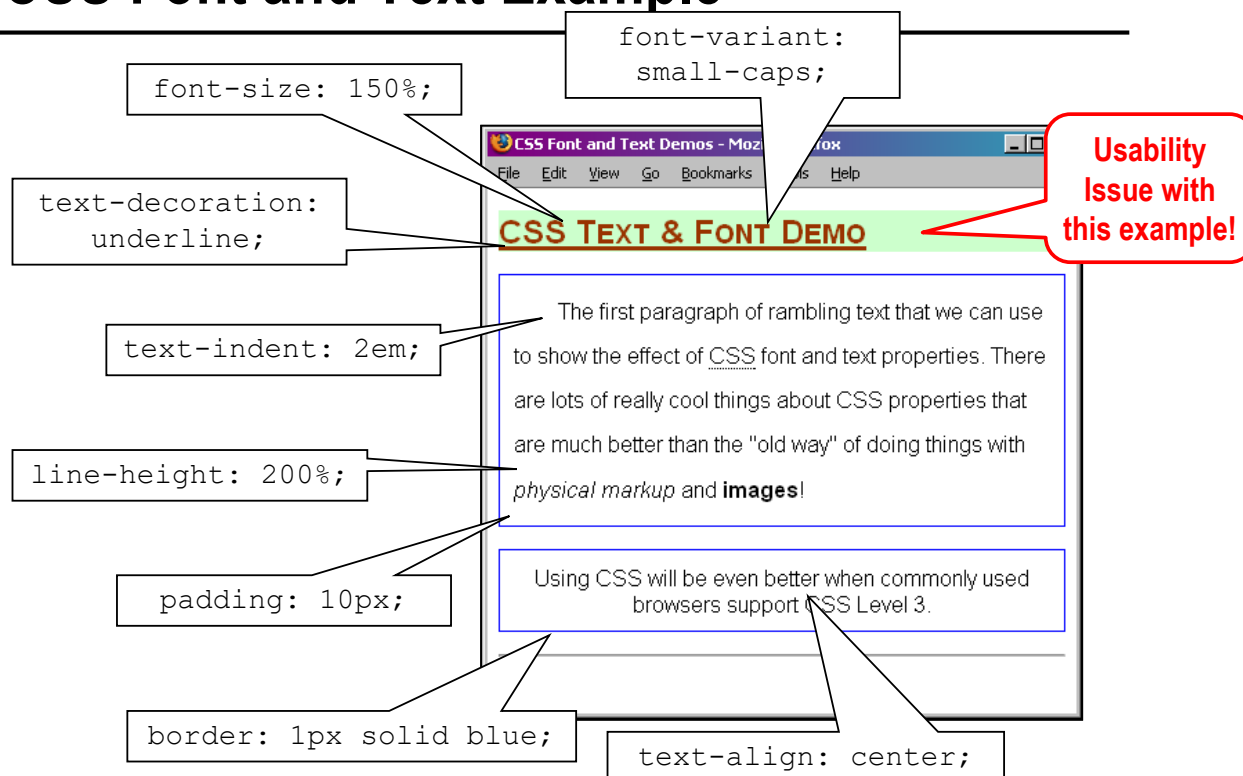
CSS Font and Text Example

```
...
<body>
<h1>CSS Text & Font Demo</h1>
<p class="intro">The first paragraph of rambling text that we can use to show
the effect of <abbr title="Cascading Style Sheets">CSS</abbr> font and text
properties. There are lots of really cool things about CSS properties that
are much better than the &quot; old way &quot; of doing things with
<em>physical markup</em> and <strong>images</strong>!</p>
<p id="tag">Using CSS will be even better when commonly used browsers support
CSS Level 3. </p>
<hr />
</body>
</html>
```



© Swinburne University of Technology

CSS Font and Text Example



© Swinburne University of Technology

CSS Font and Text Example – Family, Colour, Spacing

```
h1, p { font-family: Arial, Helvetica, sans-serif; }
```

grouping selector h1, p

```
/* shows the "block" in a background color */
```

```
h1 { background-color: #CCFFCC; color: #993300; }
```

```
/* percentage of the "normal" text size */
```

```
h1 { font-size: 150%; }
```

```
/* note that the h1 content is NOT in CAPITALS! Cool!*/
```

```
h1 { font-variant: small-caps; }
```

```
/* not good - confuses users - they think it's a hyperlink! */
```

```
h1 { text-decoration: underline; }
```

element selector h1

It would be better if these rules were grouped into one rule.

```
p.intro { line-height: 200%; }
```

```
/* "em" units will scale nicely with font size! */
```

```
p.intro { text-indent: 2em; }
```

```
/* note border values. padding between text and border */
```

```
p { border: 1px solid blue; padding: 10px; }
```

class selector .intro

element selector p

```
/* only effects the #tag element */
```

```
#tag { text-align: center; }
```

id selector #tag

CSS: Generic & Specific Fonts

- A **specific font** is a font such as “Times New Roman”, “Arial”, or “Garamond”. *Specific fonts are installed on a user's computer, so availability depends on the user's machine.*
- A **generic font** refers to the font's general appearance such as: “serif”, “sans-serif”, “monospace”, “cursive” or “fantasy”.

Five
Generic
Fonts

| Font Samples | | | |
|--------------|------|------|------|
| serif | defg | defg | defg |
| sans-serif | defg | defg | defg |
| monospace | defg | defg | defg |
| cursive | defg | defg | defg |
| fantasy | defg | defg | defg |

FF

The lines on letters are called “serifs”. “sans-serif” means “without serifs”.

CSS Font Family

- To specify the font, we use the **font-family** property.

Example:

```
p {
  font-family: Verdana; }
```



Any font names containing characters such as whitespace, font must be quoted.
eg. "Times New Roman"

- If you specify a **"specific font"** a user might not have it on their device, so you **may** list alternatives, and **should** include a final **"generic font"**

```
h1 {
  font-family: Verdana, Arial, Helvetica, sans-serif;
}
```

The preferred
specific font

Alternative
specific fonts

A "generic font"
alternative

Validation
Warning if no
generic font

- Fonts can also be downloaded using **@font**

```
@font-face {
  font-family: myfont;
  src : url("http://www.allfont.com/myfont.ttf");}
```

© Swinburne University of Technology

CSS Font Properties

- **font-size:**

```
xx-small, x-small, small, medium, large, x-large,
xx-large, smaller, larger, [length], [%]
```

- **font-style:**

```
normal, italic, oblique
```

- **font-weight:**

```
normal, bold, bolder, lighter, [100,200, 400, ... , 900]
```

- **font-variant:**

```
normal, small-caps
```

- **font-stretch:**

```
normal, wider, narrower, ultra-condensed,
extra-condensed, condensed, semi-condensed,
semi-expanded, expanded, extra-expanded, ultra-expanded
```

- **line-height:**

```
normal, [number], [length], [%]
```

red = default values

© Swinburne University of Technology

CSS Font Property

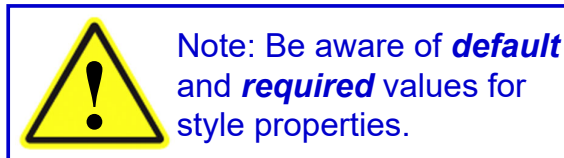
- We can write several font-properties in a shorthand, single declaration, format:

font: [style] [variant] [weight] **size** [/line-height] **family**

- ☐ **size** and **family** values *are required*
- ☐ The values in square brackets `[]` are optional.
- ☐ The first three values can be specified in any order
- ☐ `/line-height`, if used, *must* come straight after `size`

- Example:

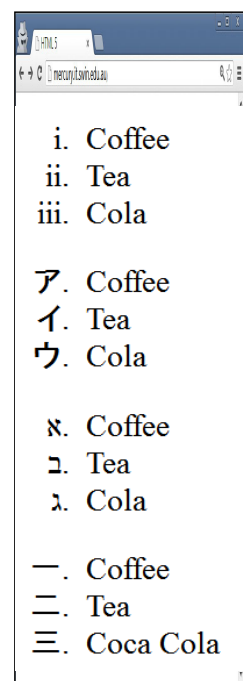
```
p {
    font: italic normal bold 10pt/14pt Helvetica, sans-serif;
}
```



© Swinburne University of Technology

Creating List styles with CSS

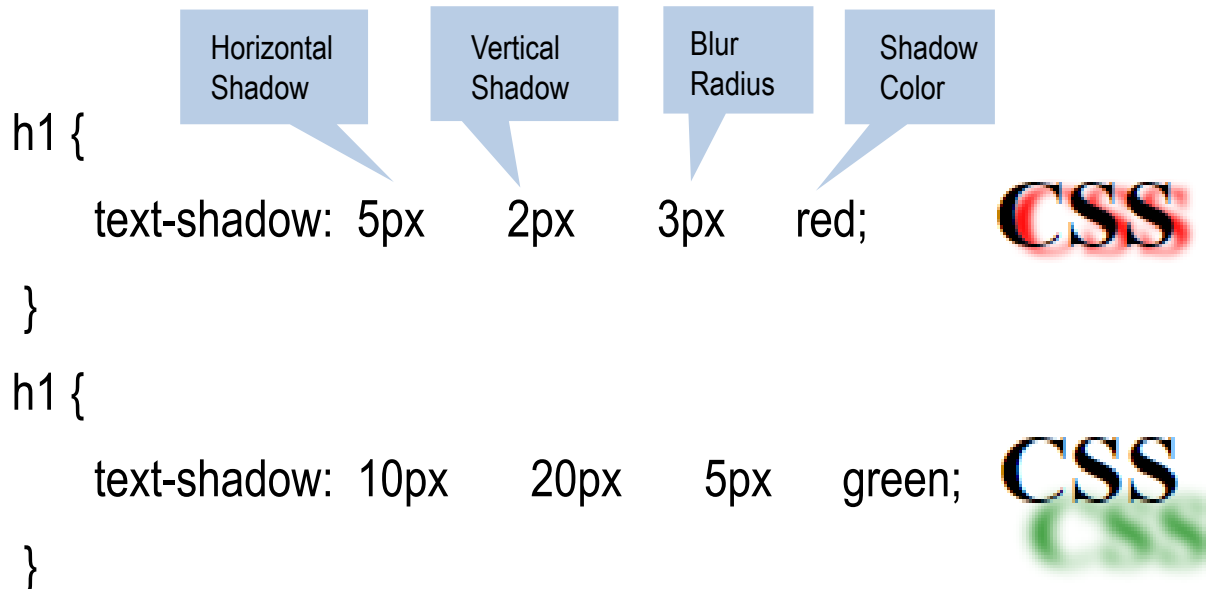
- list-style-type : none | **disc** | circle | square | decimal | decimal-leading-zero | lower-roman | upper-roman | lower-greek | lower-alpha | lower-latin | ...
 - ☐ ul.a {list-style-type:lower-roman;}
 - ☐ ul.b {list-style-type:katakana ;}
 - ☐ ul.c {list-style-type:hebrew ;}
 - ☐ ul.d {list-style-type:cjk-ideographic ;}
- list-style-image: **none** | <url>
 - ☐ list-style-image : url("spade.gif");
- list-style-position : inside | **outside**;
 - ☐ list-style-position : inside;
- list-style : [type] [position] [image];
 - ☐ list-style : lower-alpha inside;



© Swinburne University of Technology

Text Shadow

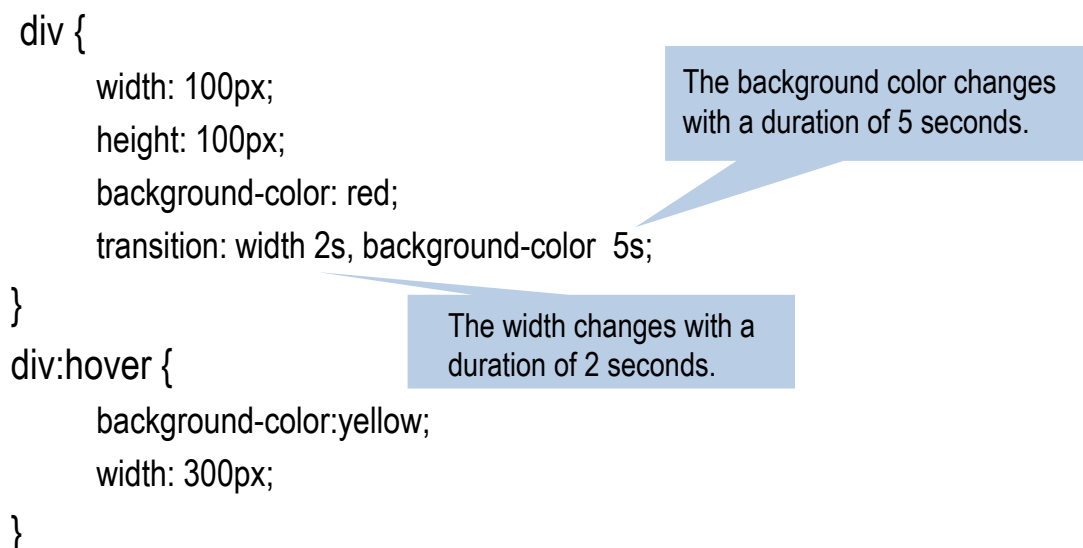
- text-shadow: adds shadow to text



© Swinburne University of Technology

Transition

- transition: change property values smoothly, over a given duration.



© Swinburne University of Technology

What's Next?

- CSS Layout
- Responsive Design