CITS1231 Web Technologies
Cascading Style Sheets: Properties Part B
CSS2 Properties: Colours

• CSS properties allow authors to specify the foreground color and background of an element. Backgrounds may be colors or images.

• Background properties allow authors to position a background image, repeat it, and declare whether it should be fixed or scroll along with the document.

  – color (a colour)
    
    em {color:red}
  
  – background-color (a colour, or “transparent”)
    
    h1 {background-color:white}
  
  – background-image (a URI)
    
    body {background-image:URL(“stripe.gif”)}
CSS2 Properties: Colours (cont.)

- CSS colours can either be a named colour or follow a numerical RGB specification:
  - HTML 4.0 Colour names are used in CSS2 as well.
    - *Aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white and yellow.*
  - Colours in numerical RGB specification
    - `em {color: rgb(255,0,0)} /* 0-255 (red) */`
    - `em {color: rgb(100%, 0%, 0%) /* 0.0% - 100.0% (red) */`
  - Colours in hexadecimal RGB specification
    - `em {color: #ff0000} /* #rrggb (red) */`
CSS Properties: Fonts

- font family (can specify order of preference)
  
  ```
  body {font-family: "Book Antiqua", "Times New Roman", serif}
  ```

- font style (normal, italic or oblique)
  
  ```
  h1, h2, h3 {font-style: italic}
  ```

- font variant (normal, small-caps)
  
  ```
  h3 {font-variant: small-caps}
  ```

- font weight (normal, bold, bolder, lighter, 100, ..., 900)
  
  ```
  strong {font-weight: normal}
  ```
CSS Properties: Font Size

• Absolute font sizes used to fix sizes to specific values.

• Five standard units (mm, cm, in, pt, pc):

```css
p {font-size: 0.5in}
p {font-size: 1cm}
p {font-size: 5mm}
p {font-size: 12pt}
p {font-size: 3pc}
```

• 1 inch (in) = 72 points (pt) = 6 picas (pc)
CSS Properties: Font Size

• Relative font size can make web page *scalable* - adapts automatically to font that reader uses.

• Examples of relative units: *percentage* and *em unit*

  ```
  p {font-size: 150%}
  em {font-size: 1.5em}
  ```

• Don’t confuse em selector from em unit.

• 100% or 1em is equal to font size of the parent element.
CSS Properties: Font Size

```html
<html>
  <head>
    <style type="text/css">
      <!--
        h1 {font-size: 2em}
        em {font-size: 1.5em}
      -->
    </style>
  </head>
  <body>
    Normal body text.
    <em>em text nested in body element</em>
    <h1>h1 text nested in body element.<em>em text nested in h1 element</em></h1>
  </body>
</html>
```

Normal body text = 100%
First em text = 150%
h1 text = 200%
Second em text = 300%
CSS Properties: Font Size in Pixels

Using Pixels:

- 1 pixel = 1 dot on output device
- Different devices have different resolutions
- 600 dpi printer has more pixels per inch than PC monitor

\[
\text{h1 \{ font-size:20px \}}
\]

Using keywords:

- xx-small, x-small, small, medium, large, x-large, xx-large
- smaller, larger (relative: 1 size smaller or larger)

\[
\text{h1 \{ font-size:xx-small \}}
\]
\[
\text{h1 \{ font-size:larger \}}
\]
Percentage and em unit

- When percentage and em unit used to specify size other than font size, they have different meaning.
- 1em ~ width of capital M for current element font.
  - To make h1 heading have width about 20 capital M’s:
    ```
    h1 {width: 20em}
    ```
- 100% = width of parent element.
  - To make h1 heading have width 50% of body text width:
    ```
    h1 {width: 50%}
    ```
CSS Properties: Text

• The presentation of text can be adjusted by:
  – text-indent (the amount of indentation using absolute length or percentage)
    
    ```css
    p {text-indent:3em}
    ```
  – text-align (left, center, right, justify)
    
    ```css
    div.center {text-align:center}
    ```
  – text-decoration (none, underline, overline, line-through)
    
    ```css
    a[href] {text-decoration: underline}
    ```
  – Other properties are also available.
CSS2 Properties: Boxes

- Every displayable element considered to fall inside a rectangular box.
- Each box has an external margin, a border, internal padding and content (e.g. text or images, etc.)
  - margin (a length or a percentage)
    ```
    body {margin:2em}
    ```
  - padding (a length or a percentage)
    ```
    td {padding: 5pt}
    ```
  - border-width (thin, medium, thick or a length)
    ```
    colgroup {border-width: 5pt}
    ```
  - border-color (a colour)
    ```
    p {border-color:red}
    ```
  - border-style (solid, double, dashed, ...)
    ```
    h2 {border-style:double}
    ```
  - Individual box sides can also be targeted:
    ```
    div {border-style-left:dashed}
    ```
Working with the Box Model

- The **box model** is an element composed of four sections:
  - Margin
  - Border
  - Padding
  - Content
## Border Styles

<table>
<thead>
<tr>
<th>Border Style</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>border-top-width: value</td>
<td>Width of the top border</td>
<td>Where <code>value</code> is the width of the border in absolute or relative units, or defined with the keyword “thin”, “medium”, or “thick”</td>
</tr>
<tr>
<td>border-right-width: value</td>
<td>Width of the right border</td>
<td></td>
</tr>
<tr>
<td>border-bottom-width: value</td>
<td>Width of the bottom border</td>
<td></td>
</tr>
<tr>
<td>border-left-width: value</td>
<td>Width of the left border</td>
<td></td>
</tr>
<tr>
<td>border-width: top right bottom left</td>
<td>Width of any or all of the borders</td>
<td></td>
</tr>
<tr>
<td>border-top-color: color</td>
<td>Color of the top border</td>
<td>Where <code>color</code> is a color name or color value</td>
</tr>
<tr>
<td>border-right-color: color</td>
<td>Color of the right border</td>
<td></td>
</tr>
<tr>
<td>border-bottom-color: color</td>
<td>Color of the bottom border</td>
<td></td>
</tr>
<tr>
<td>border-left-color: color</td>
<td>Color of the left border</td>
<td></td>
</tr>
<tr>
<td>border-color: top right bottom left</td>
<td>Color of any or all of the borders</td>
<td></td>
</tr>
<tr>
<td>border-top-style: type</td>
<td>Style of top border</td>
<td>Where <code>type</code> is one of the nine border styles: solid, dashed, dotted, double, outset, inset, groove, ridge, or none</td>
</tr>
<tr>
<td>border-right-style: type</td>
<td>Style of right border</td>
<td></td>
</tr>
<tr>
<td>border-bottom-style: type</td>
<td>Style of bottom border</td>
<td></td>
</tr>
<tr>
<td>border-left-style: type</td>
<td>Style of left border</td>
<td></td>
</tr>
<tr>
<td>border-style: top right bottom left</td>
<td>Style of any or all of the borders</td>
<td></td>
</tr>
</tbody>
</table>
Border Style Types

- solid
- dashed
- dotted
- double
- outset
- inset
- groove
- ridge
- none
### CSS2 Properties: Position

<table>
<thead>
<tr>
<th>Property Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>static</td>
<td>This is the default setting – no special positioning.</td>
</tr>
<tr>
<td>absolute</td>
<td>Move element relative to upper left corner of page or a containing element.</td>
</tr>
<tr>
<td>relative</td>
<td>Move element relative to its default position.</td>
</tr>
<tr>
<td>fixed</td>
<td>Move element relative to browser window – ie doesn’t change position if scrolling content.</td>
</tr>
</tbody>
</table>

These used in conjunction with the **top**, **right**, **bottom**, **left** offset properties.
CSS2 Positioning Examples: Relative

<p>This is some text.</p>

<span style="position:relative; top: -1em">Up we go!</span>

Here is some more text.

</p>
CSS2 Positioning Absolute and Fixed

```
<html>
<head>
    <style type="text/css">
        h1 { position:fixed; left:250px; top:50px; background:lime; }
        h2 { position:absolute; left:150px; top:150px; }
    </style>
</head>
<body>
    <h1>This is a h1 with an fixed position</h1>
    <h2>This is a h2 with an absolute position</h2>
    <p>blah blah blah.</p>
    <p>blah blah blah.</p>
    <p>blah blah blah.</p>
    <p>blah blah blah.</p>
    <p>blah blah blah.</p>
    <p>blah blah blah.</p>
    <p>blah blah blah.</p>
    <p>blah blah blah.</p>
    <p>blah blah blah.</p>
    <p>blah blah blah.</p>
</body>
</html>
```

http://undergraduate.csse.uwa.edu.au/units/CITS1231/lectures/lecture7_ex2.html
The Resulting Page

Element with fixed position doesn’t move when scrolling content.
Stacking Elements

• Specify stacking order with:
  – `z-index: value`
CSS2: 3D Layering

- In CSS2, each box has a position in three dimensions. In addition to their horizontal and vertical positions, boxes lie along a “z-axis” and are formatted one on top of the other.

- Positioning on the z-axis is controlled by the `z-index` property (0,1,2,3 ...). The higher `z-index`-ed objects are stacked above objects with lower z-index.

- Given:
  ```css
  .pile {position:absolute; top: 1in; left:1in; width:1in; height:1in}
  ```

How would this HTML snippet to be rendered?

```html
<img class="pile" style="z-index:1" src="aibo.jpg" alt="AIBO" />
<div class="pile" style="z-index:3">Top text</div>
<div class="pile">Bottom text</div>
<div class="pile" style="z-index:2">Middle text</div>
```
With v.s. Without 3D layering
CSS2: Visibility and Overflow

• By default a box is *visible*

• A box is made invisible by setting its visibility to hidden:

  
  ```html
  <p style="visibility: hidden"> ... </p>
  ```

• Invisible boxes still affect layout.

• Display of content that does not fit in box is determined by *overflow*. 
Working with overflow and clipping

The overflow property syntax:

- `overflow: type`
CSS2: Display style

selector {display:Value}

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>The element will generate no box at all.</td>
</tr>
<tr>
<td>block</td>
<td>The element will generate a block box (a line break before and after the element).</td>
</tr>
<tr>
<td>Inline</td>
<td>The element will generate an inline box (no line break before or after the element). This is default.</td>
</tr>
<tr>
<td>inline-block</td>
<td>The element will generate a block box, laid out as an inline box.</td>
</tr>
<tr>
<td>inline-table</td>
<td>The element will generate an inline box (like &lt;table&gt;, with no line break before or after).</td>
</tr>
<tr>
<td>inherit</td>
<td>Specifies that the value of the display property should be inherited from the parent element.</td>
</tr>
</tbody>
</table>
## CSS2: Display style cont.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>list-item</td>
<td>The element will generate a block box, and an inline box for the list marker</td>
</tr>
<tr>
<td>run-in</td>
<td>The element will generate a block or inline box, depending on context</td>
</tr>
<tr>
<td>table</td>
<td>The element will behave like a table (like <code>&lt;table&gt;</code>, with a line break before and after)</td>
</tr>
<tr>
<td>table-caption</td>
<td>The element will behave like a table caption (like <code>&lt;caption&gt;</code>)</td>
</tr>
<tr>
<td>table-cell</td>
<td>The element will behave like a table cell</td>
</tr>
<tr>
<td>table-column</td>
<td>The element will behave like a table column</td>
</tr>
<tr>
<td>table-column-group</td>
<td>The element will behave like a table column group (like <code>&lt;colgroup&gt;</code>)</td>
</tr>
<tr>
<td>table-footer-group</td>
<td>The element will behave like a table footer row group</td>
</tr>
<tr>
<td>table-header-group</td>
<td>The element will behave like a table header row group</td>
</tr>
<tr>
<td>table-row</td>
<td>The element will behave like a table row</td>
</tr>
<tr>
<td>table-row-group</td>
<td>The element will behave like a table row group</td>
</tr>
</tbody>
</table>
Media Types

• By default a style sheet is applied to all devices.

• In CSS2 can specify style for specific devices:

  <style type="text/css" media="type">
    ...
  </style>

  or

  <link href="url" type="text/css" media="type" ... />

• If no media attribute, then style used for all media.
### Media Attribute Values

<table>
<thead>
<tr>
<th>Media Value</th>
<th>Used For</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>All output devices (the default)</td>
</tr>
<tr>
<td>aural</td>
<td>Speech and sound synthesizers</td>
</tr>
<tr>
<td>braille</td>
<td>Braille tactile feedback devices</td>
</tr>
<tr>
<td>embossed</td>
<td>Paged Braille printers</td>
</tr>
<tr>
<td>handheld</td>
<td>Small or handheld devices with small screens, monochrome graphics, and limited bandwidth</td>
</tr>
<tr>
<td>print</td>
<td>Printers</td>
</tr>
<tr>
<td>projection</td>
<td>Projectors</td>
</tr>
<tr>
<td>screen</td>
<td>Computer screens</td>
</tr>
<tr>
<td>tty</td>
<td>Fixed-width devices like teletype machines and terminals</td>
</tr>
<tr>
<td>tv</td>
<td>Television-type devices with low resolution, color, and limited scrollability</td>
</tr>
</tbody>
</table>
Media Types: Examples

- Style for aural devices:

  `<style type="text/css" media="aural">`  
  
  ...  
  
  `</style>`

  or

  `<link href="sounds.css" type="text/css" media="aural" />`

- Style for printer:

  `<link href="print.css" type="text/css" media="print" />`
Cascade Through Media Types

- Style sheets cascade through the media types.

- In following, h1 headings will be red sans-serif for computer screens, but black sans-serif for printed pages:

```html
<style type="text/css">
  h1 {font-family: sans-serif}
</style>
<style type="text/css" media="screen">
  h1 {color: red}
</style>
<style type="text/css" media="print">
  h1 {color: black}
</style>
```
The @media Rule and media groups

• Can also specify the output media within a style sheet using:

```css
@media type {style declarations}
```

where media is one of the supported media types and style declarations are the styles associated with that media type

• CSS2 uses media groups to describe basic facets of different media— and to differentiate between different types of media based on the ways they render content
  - Continuous or paged
  - Visual, aural, or tactile
  - Grid (for character grid devices) or bitmap
  - Interactive or static
## Media Groups

<table>
<thead>
<tr>
<th>Media Types</th>
<th>continuous/paged</th>
<th>Media Groups</th>
<th>grid/bitmap</th>
<th>interactive/static</th>
</tr>
</thead>
<tbody>
<tr>
<td>aural</td>
<td>continuous</td>
<td>aural</td>
<td>N/A</td>
<td>both</td>
</tr>
<tr>
<td>braille</td>
<td>continuous</td>
<td>tactile</td>
<td>grid</td>
<td>both</td>
</tr>
<tr>
<td>embossed</td>
<td>paged</td>
<td>tactile</td>
<td>grid</td>
<td>both</td>
</tr>
<tr>
<td>handheld</td>
<td>both</td>
<td>visual</td>
<td>both</td>
<td>both</td>
</tr>
<tr>
<td>print</td>
<td>paged</td>
<td>visual</td>
<td>bitmap</td>
<td>static</td>
</tr>
<tr>
<td>projection</td>
<td>paged</td>
<td>visual</td>
<td>bitmap</td>
<td>static</td>
</tr>
<tr>
<td>screen</td>
<td>continuous</td>
<td>visual</td>
<td>bitmap</td>
<td>both</td>
</tr>
<tr>
<td>tty</td>
<td>continuous</td>
<td>visual</td>
<td>grid</td>
<td>both</td>
</tr>
<tr>
<td>tv</td>
<td>both</td>
<td>visual, aural</td>
<td>bitmap</td>
<td>both</td>
</tr>
</tbody>
</table>
Hiding Elements

- Two different styles that allow you to hide elements:
  - Display style
  - Visibility style

Visibility hidden
Object is hidden but still is part of the page flow

Display: none
Object is hidden and is removed from the page flow
Using Print Styles

- To use a print style sheet, your web page should have:

  ```html
  <link rel="stylesheet" type="text/css" href="print.css" media="print" />
  ```

- Put `<div>` elements with id’s around sections you don’t want printed. For example:

  ```html
  <div id="navigation" name="navigation">
  <div id="advertising" name="advertising">
  <div id="content" name="content">
  <div id="other" name="other">
  ```

- Your print.css can now use these id’s to avoid printing their associated elements.
Example: print.css

body {
    margin:0;
    color : #000000;
    background : #ffffff;
    font-family : "Times New Roman", Times, serif;
    font-size : 12pt; }

a {
    text-decoration : underline;
    color : #0000ff; }

#navigation, #advertising, #other {
    display : none; }
Hexadecimal Numbers

- Decimal numbers have ten symbols 0-9 (base 10).
- Value of decimal numbers:
  \[ 56 = 5 \times 10^1 + 6 \times 10^0 \]
- Hexadecimal numbers have sixteen symbols (base 16):
  0 – 9, A, B, C, D, E, F (lower case ok)
- Value of hexadecimal numbers:
  \[ a6 = 10 \times 16^1 + 6 \times 16^0 = 166 \]
  \[ ff = 15 \times 16^1 + 15 \times 16^0 = 255 \]
- red = rgb(255,0,0) = #ff0000