COS10005 Web Development

Module 5 – CSS Presentation

Contents

- Recap – HTML and CSS Basics
- CSS Presentation - continuation
  - Web Typography
    - Colour
    - Fonts
    - Text
  - Model
    - Visual Formatting
    - Box
  - Page Layout
    - Fixed and Fluid
    - Position

Notes:

Embedded CCS is used throughout these lecture notes for single slide reference and discussion purposes only.

It is ***strongly recommended*** *that external CCS files be used in practice, as demonstrated in the labs.*

RECAP

Building an HTML page

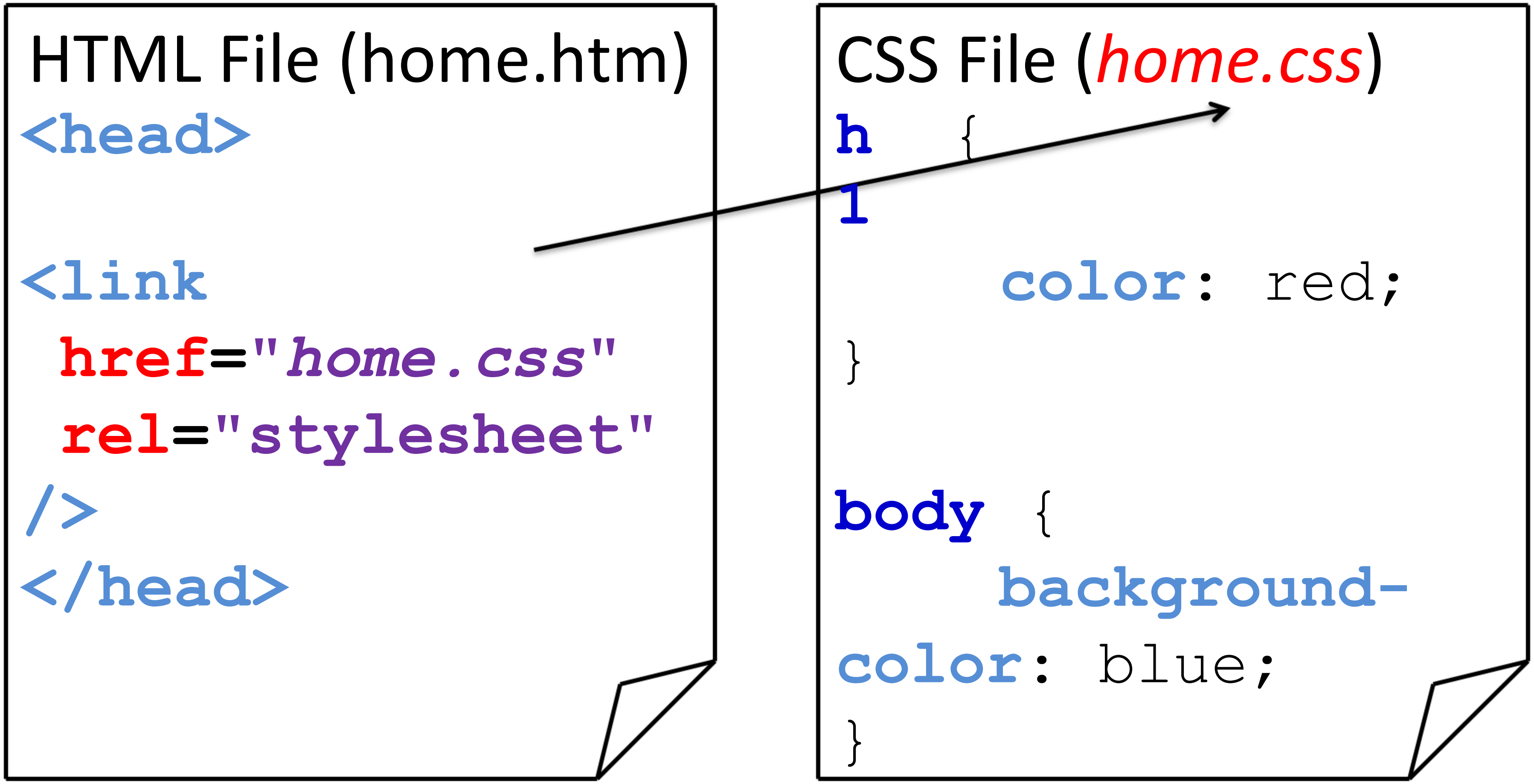
HTML **Structural** Elements

- HTML <head>, <body>
- Header <header>
- Navigation <nav>
- Main <main>
- Article <article>
- Section <section>
- Aside <aside>
- Footer <footer>
- Others <div>

HTML **Content** Elements

- Heading <h1>, <h2> …<h6>
- Paragraph <p>
- List <ul>, <ol>
- Table <table>
- Anchor <a>
- Image <img />
- Horizontal Rule <hr />
- Form <form>

Linking HTML file to CSS file



HTML File (home.htm)

```
<head>

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

CSS File (*home.css*)

```
h   {
1
      color: red;
}

body {
      background-
color: blue;
}
```

Writing CSS Codes

- Format of style rules

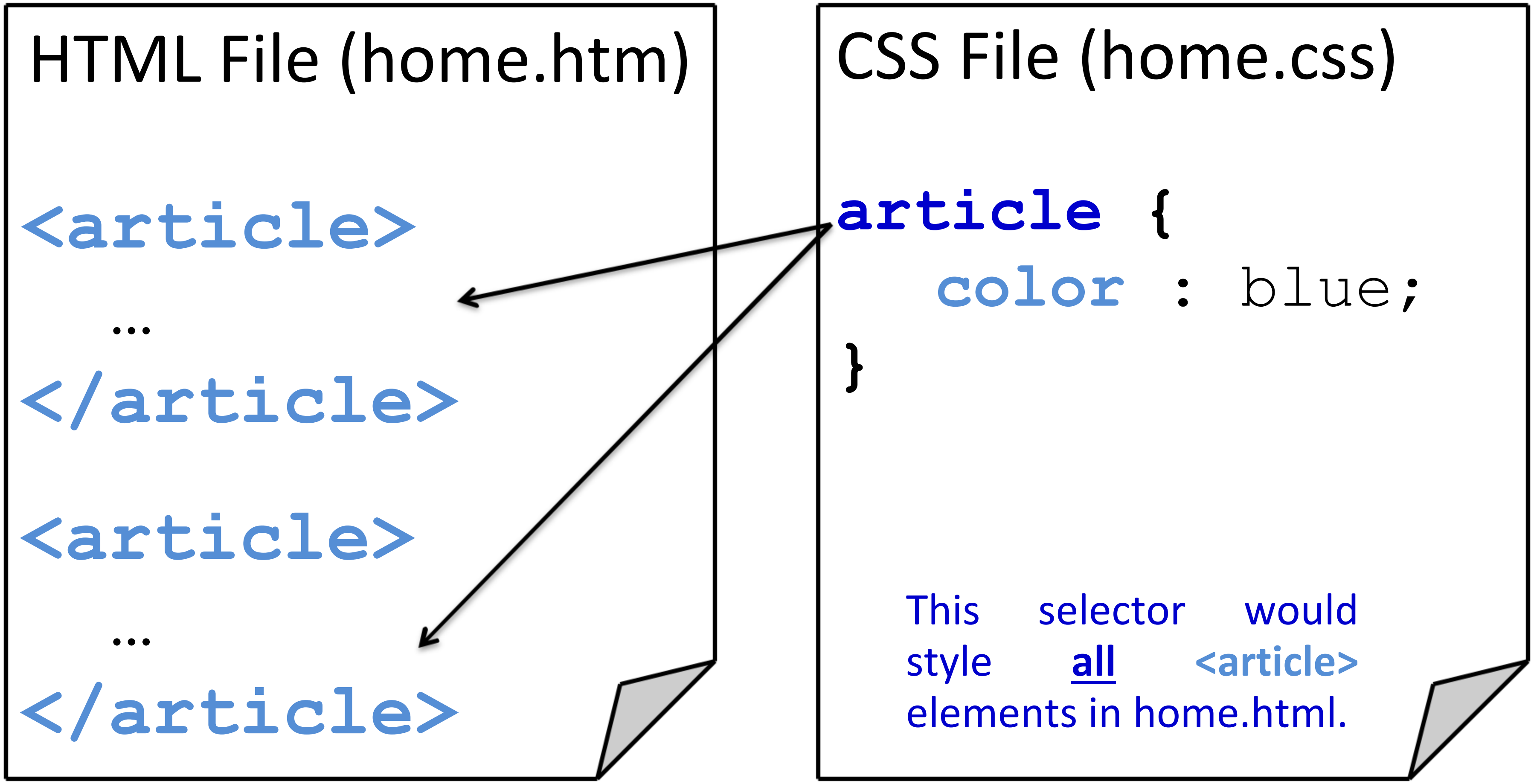
```
selector { property : value;
     …
   }
```

Declaration Block

- Example

```
article {
    color : blue;
    background-color:black;
}
```

Applying CSS Rules to HTML Elements



HTML File (home.htm)

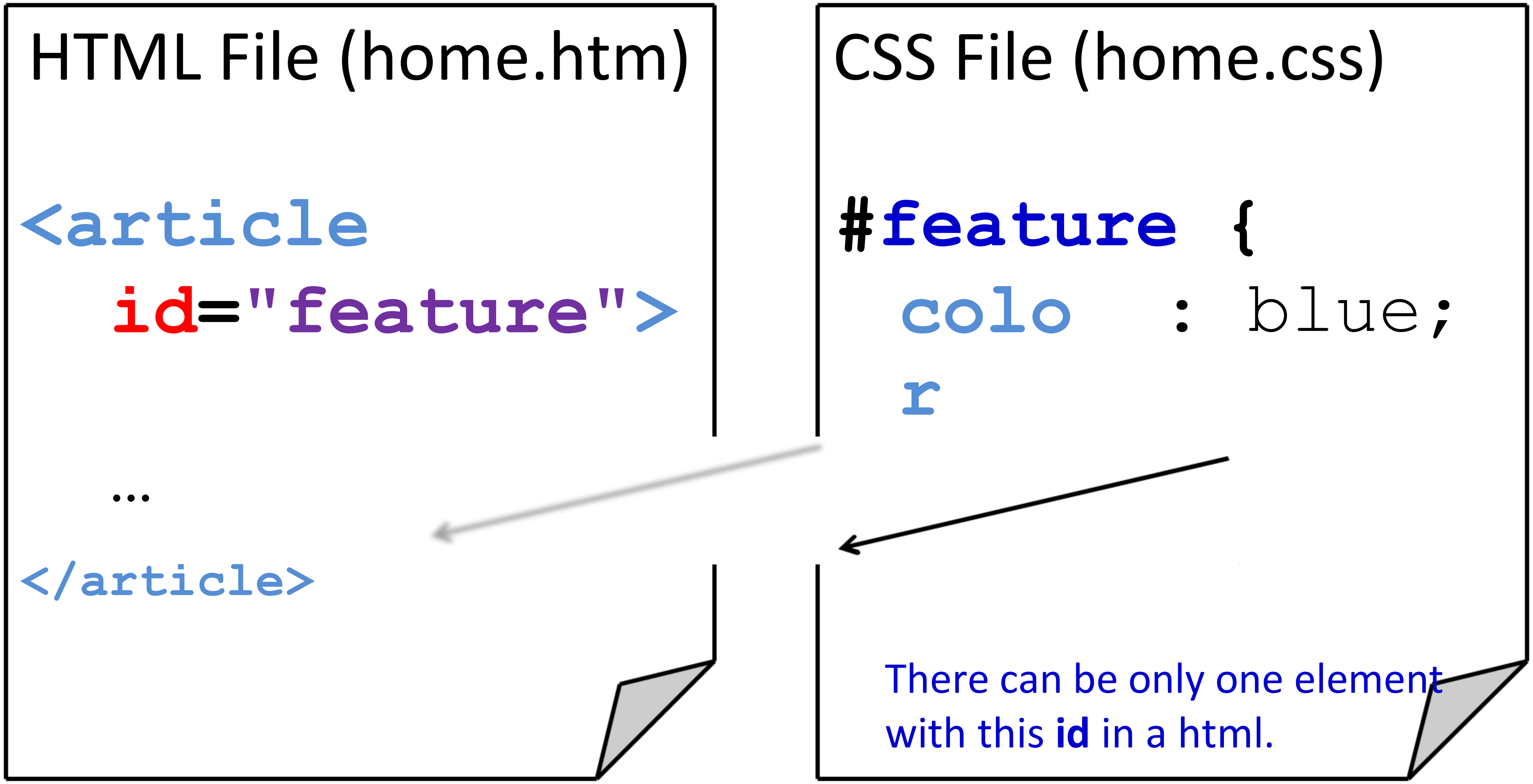
```
<article>
   …
</article>

<article>
   …
</article>
```

CSS File (home.css)

```
article {
   color : blue;
}
```

This selector would style **all** <article> elements in home.html.

Selecting HTML Elements using ***id***



HTML File (home.htm)

```
<article
   id="feature">

   …
</article>
```

CSS File (home.css)

```
#feature {
  colo  : blue;
  r
```

There can be only one element with this **id** in a html.

Selecting HTML Elements using ***class***

HTML File (home.htm)

```
<article
 class="regular">
   …
</article>

<h1
class="regular">
   …
</h1>
```

CSS File (home.css)

```
.regular {
   color : blue;
}
```

If you want to select only the <article> elements in class *regular*:

```
article.regular {
   color : blue;
}
```

Writing CSS Comments

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

```
/*
   defines the style for all articles
*/
article {
   color : blue; /* font color*/
}
```

Writing CSS Comments

- Comments at the beginning of a CSS file are often used to help maintenance and Quality Assurance
- Example

```
style.css
/* author:         [your name]
last modified:     [date]
validated:         [date]
description: [description]
*/

h1 {
    ......
}
```

CSS PRESENTATION:

WEB TYPOGRAPHY

Dimension, Color, Fonts, Text

Web Typography: Dimension

- ***Absolute*** is used for printed media

Unit	Abbr	Description	Example
centimetre	cm	metric centimetre	**p** {**padding** :1cm;}
inch	in	US inch	**p** {**margin**: 1.25in;}
millimetre	mm	metric millimetre	**p** {**word-spacing**: 10mm;}
pica	pc	Equal to 12 points	**p** {**font-size**: 20pc;}
point	pt	Equal to 72 points in an inch	**p** {**font-size**: 24pt;

Web Typography: Measurement

- ***Relative*** is used for webpage

Unit	Abbr	Description	Example
EM	**em**	Height of the current font size	**p** {**padding** :2em;}
Percentage	**%**	Percentage as relative to parent element	**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;}

DEMO! - measurement.html

Web Typography: Colour

- color : <colour values>;
  - Colour values can be in **text** or **numerical** format
    - aqua | black | blue | fuchsia | gray | grey | green | lime | maroon | navy | olive | purple |red | silver | teal | white | yellow
    - #<6-digit hexadecimal> representing rrggbb
    - rgb|rgba|hsl|hsla (<0-255>, <0-255>, <0-255>)
  - **h1** {color : blue;}
  - **h1** {color : #0000FF;}
  - **h1** {color : rgb (0, 0, 255);}

Web Typography: Fonts

- A ***specific font*** is a font such as “Times New Roman”, “Arial”, or “Courier New”. *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”.

FF

Five Generic Fonts

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

Web Typography: Font Principles

- Use commonly available fonts
  - Times New Roman
  - Arial
  - Courier New
- Use **font-family** safely
  - **article** {**font-family**: Times New Roman, serif}
  - **article** {**font-family**: Arial, sans-serif}
  - **article** {**font-family**: Courier New, monospace}

DEMO!
font.html

Times New Roman
Garamond
Georgia
} serif

Arial
Verdana
Trebuchet
} sans-serif

Arial
Verdana
Trebuchet
} serif

Web Typography: Fonts

- **font-family**: <font name>;
  - **font-family**: Arial, "Times New Roman";
- **font-size**: <value> | <keyword>;
  - h1 **{font-size**: 0.8em;}
  - h1 **{font-size**: small;}
    - xx-small/x-small/small/**medium**/large/x-large/xx-large
- **font-style**: **normal** | italic;
  - h1 **{font-style**: italic;}

DEMO!
font.html

Web Typography: Fonts (continued)

- font-variant: **normal** | small-caps;
  - font-variant: small-caps;
- font-weight: **normal** | bold | bolder | lighter | 100|200|300|400|500|600|700|800|900;
  - font-weight : bold;
- font: <short cut property>;
  - font: bold 1.2em Arial;

font-weight, font-size and font-family on one line.

DEMO!
font.html

Web Typography: Design Principles

- Font Size
  - Size should be big enough, as screen resolution is lower than paper
- Font Color
  - Provide enough contrast between text color and background color
- Avoid using graphics as text
  - Avoid using images to display text
  - Use styled text

Web Typography: Text Alignment

- text-align : **left** | right | center | justify;
  - text-align : center;
    - *Justify is not supported by all browser*
- text-indent: <value>; (first line of paragraph)
  - text-indent : 2em;
  - text-indent : -2em; (for hanging indent)

DEMO! – font.html

Web Typography: Text Alignment

- **vertical-align** : **baseline** | sub | super | top | text-top | middle | bottom | text-bottom | <value>
  - **vertical-align** : super; (superscript)
  - **vertical-align** : middle; (used on table cell)
  - **vertical-align** : text-top; (used on images)

Used together with **display**: table-cell; to vertically align text.

Alternatively, you can use **line-height**.

DEMO! – font.html

Web Typography: Text Spacing

- letter-spacing: **normal** | <value>;
  - letter-space : 6px;
- word-spacing: **normal** | <value>;
  - word-spacing : 2em;
- text-decoration : **none** | underline | overline | line-through;
  - text-decoration : underline;

DEMO!

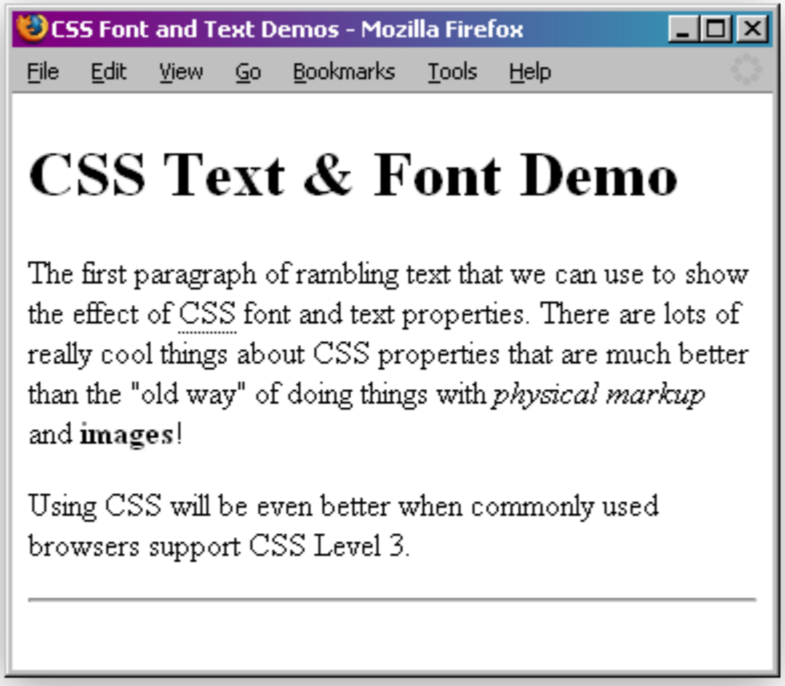
Web Typography: Text

- **text-transform** : **none** | capitalize | uppercase | lowercase;
  - text-transform : capitalize; (first letter of every word)
- **text-shadow** : **none** | <horizontal> <vertical> <blur> <colour>;
  - **text-shadow**:2px 2px 4px red;

DEMO! –
font.html

CSS Font and Text Example

```
...
<body>
<h1>CSS Text &amp; 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>
```



CSS Text & Font Demo

The first paragraph of rambling text that we can use to show the effect of CSS font and text properties. There are lots of really cool things about CSS properties that are much better than the "old way" of doing things with *physical markup* and **images**!

Using CSS will be even better when commonly used browsers support CSS Level 3.

CSS Font and Text Example

font-variant:
small-caps;

font-size: 150%;

text-decoration:
underline;

text-indent: 2em;

line-height: 200%;

padding: 10px;

border: 1px solid blue;

text-align: center;

CSS Font and Text Example

```
<!DOCTYPE html>
<html>
<head>
<title>CSS Font and Text Demos</title> <style
type="text/css" >
h1, p { font-family: Arial, Helvetica, sans-serif; }

/* 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; }

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; }

/* only effects the #tag element */
#tag { text-align: center;}
</style>
</head>
...
```

grouping selector **h1, p**

element selector **h1**

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

class selector **.intro**

element selector **p**

id selector **#tag**

Web Typography: List

- list-style-type : none | **disc** | circle | square | decimal | decimal-leading-zero | **lower-roman** | upper-roman | lower greek | lower-alpha | lower-latin | upper-alpha | upper-latin | hebrew | armenian | georgian | cjk-ideographic | hiragana | katakana | hiragana-iroha | katakana-iroha
  - ol.a {list-style-type: lower-roman;}
  - ol.b {list-style-type: katakana;}
  - ol.c {list-style-type: hebrew;}
  - ol.d {list-style-type: cjk-ideographic;}

Web Typography: List (continued)

- list-style-image: **none** | <url>
  - – list-style-image : url("circle.png");
- list-style-position : inside | **outside**;
  - – list-style-position : inside;

DEMO! – font.html

- list-style : <type> <position> <image>;
  - list-style : lower-alpha inside url("circle.png");

list-style-type, list-style-position and list-style-image on one line.

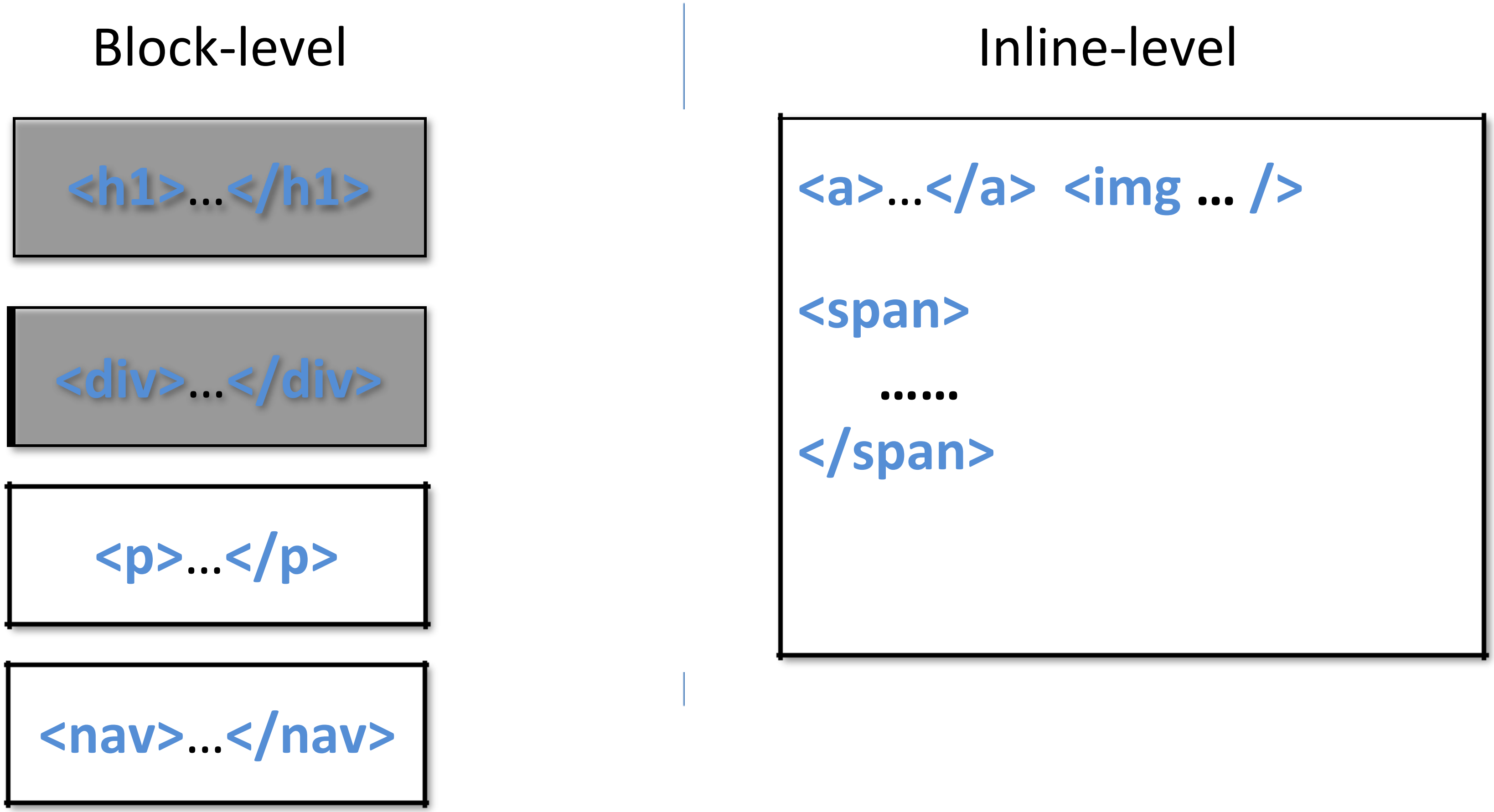
CSS PRESENTATION:
BOX MODEL AND VISUAL FORMATTING

Models: Visual Format and Box Model

- Visual formatting model describes how the element content boxes should be displayed
  - Block-level elements appear as blocks *such as* ***<p>***, ***<div>***, ***<header>***, ***<article>***, *etc.*
  - Inline-level elements are contained within block-level elements, *such as* ***<a>***, ***<img>***, *etc.*
- ***Box model*** describes the rectangular boxes that contain content on a web page

Model: Visual Formatting

- How will the boxes be arranged?



Block-level

<h1>…</h1>

<div>…</div>

<p>…</p>

<nav>…</nav>

Inline-level

<a>…</a> <img … />

<span>

……

</span>

Model: Visual Formatting

- Default arrangement is top to bottom left to right according to how the elements are ordered

```
<h1>…</h1>
<div>…</div>
<p>…
  <a>…</a>
  <img … />
  <span> … </span>
</p>
```

<h1>…</h1>

<div>…</div>

<p>…

<a>…</a>   <img … >   <span>…</span>

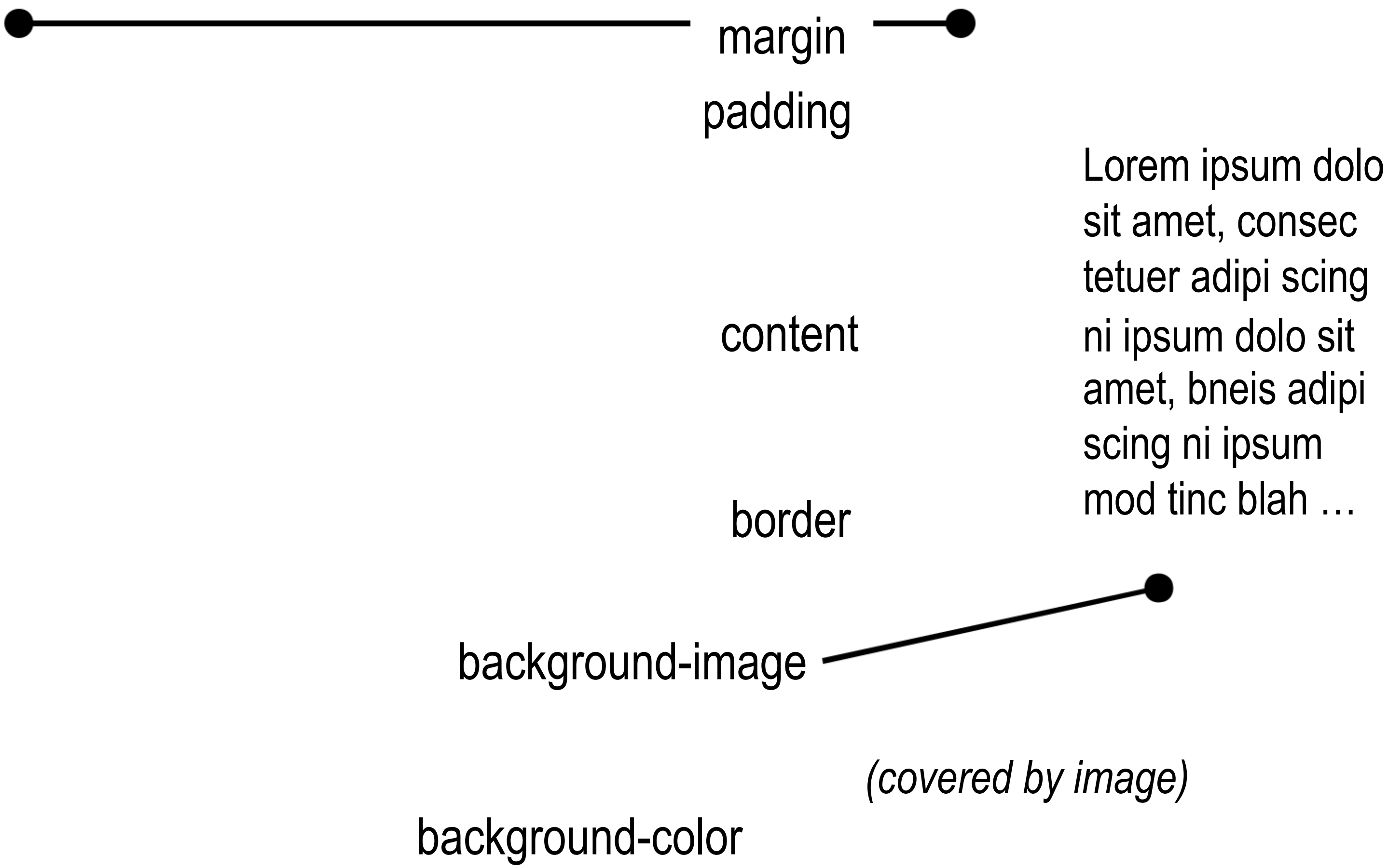
</p>

Block-level

Inline-level

The CSS Box Model

- Below is the CSS box model of a block-level element.



margin

padding

content

Lorem ipsum dolo
sit amet, consec
tetuer adipi scing
ni ipsum dolo sit
amet, bneis adipi
scing ni ipsum
mod tinc blah …

border

background-image

(covered by image)

background-color

CSS Margin

- Margin is the space **outside** the element's border.

`margin: 6px`

`margin: 10px`

Lorem ipsum dolo sit amet, consec tetuer adipi scing ni ipsum dolo sit amet, bneis adipi scing ni ipsum mod tinc blah …

Lorem ipsum dolo sit amet, consec tetuer adipi scing ni ipsum dolo sit amet, bneis adipi scing ni ipsum mod tinc blah …

DEMO! – font.html

The separation will be 10px
the biggest of the margins set

Model: Box – Margin

- **margin** : <values>;
  - Margin is always transparent
  - Margin values can be negative

Individual	margin-	margin-	margin-	margin-
margin:	**top:**	**right:**	**bottom:**	**left:**
1em	1em	1em	1em	1em
1em 2em	1em	2em	1em	2em
1em 2em 3em	1em	2em	3em	2em
1em 2em 3em 4em	1em	2em	3em	4em

```
p {margin: 1em 2em 3em 4em;}
```

Model: Box – Padding

- Padding is the space **between** the element's border and its content.

Margin-right: 12px

Margin-left: 12px

Lorem ipsum dolo
sit amet, consec
tetuer adipi scing
ni ipsum dolo sit
amet, bneis adipi
scing ni ipsum
mod tinc blah …

Lorem ipsum dolo
sit amet, consec
tetuer adipi scing
ni ipsum dolo sit
amet, bneis adipi
scing ni ipsum
mod tinc blah …

DEMO! – font.html

Padding-bottom: 8px;

Padding-bottom: 6px

Model: Box – Padding

- **padding** : <values>;
  - Padding is the space inside the element's border and takes the colour of the element's background colour

Individual	padding-	padding-	padding-	padding-
padding:	**top:**	**right:**	**bottom:**	**left:**
1em	1em	1em	1em	1em
1em 2em	1em	2em	1em	2em
1em 2em 3em	1em	2em	3em	2em
1em 2em 3em 4em	1em	2em	3em	4em

```
p {padding: 1em 2em 3em 4em}
```

Model: Box – Border

- **border** : <style> <width> <colour>;
  - style values are
    - none | dashed | solid | double | groove | ridge | inset | outset
  - width values are
    - thin | medium | thick | <size values>
  - colour values are
    - aqua | black | blue | fuchsia | gray | grey | green | lime | maroon | navy | olive | purple |red | silver | teal | white | yellow
    - #<6-digit hexadecimal>
    - rgb|rgba|hsl|hsla (<0-255>, <0-255>, <0-255>)

Model: Box – Border Shorthand

- **border-style**
  - **div {border-style: solid|double|dotted|dashed|……}**
- **border-width**
  - **div {border-width: 10px|thin|thick|}**
- **border-color**
  - **div {border-color:red|#FF0000; }**
- Using the border shorthand

div {border : solid 1px blue;}

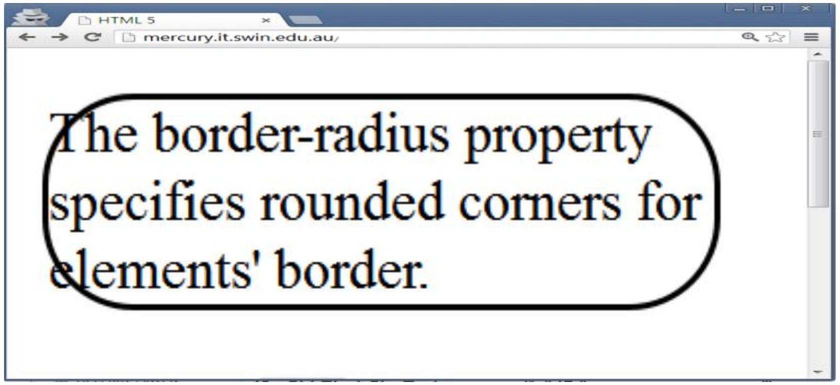
border-style, **border-width** and **border-color** on one line.

DEMO! –
font.htm
l

Model: Box – Border Corners

- **border-radius** : <values>;
  - **border-radius** specifies rounded corners for the elements' border. A border is required.

```
p {
     border : 1px solid;
     border-radius:25px;
}

<p>The border-radius
property specifies
rounded corners for
elements' border.</p>
```



The border-radius property specifies rounded corners for elements' border.

Remember to adjust the padding accordingly.

DEMO! – font.html

Model: Box – Background

- **background-color** : <colour values>;
- **background-image** : url("URL") | **none**;

Background properties provides control over backgrounds of block-level elements

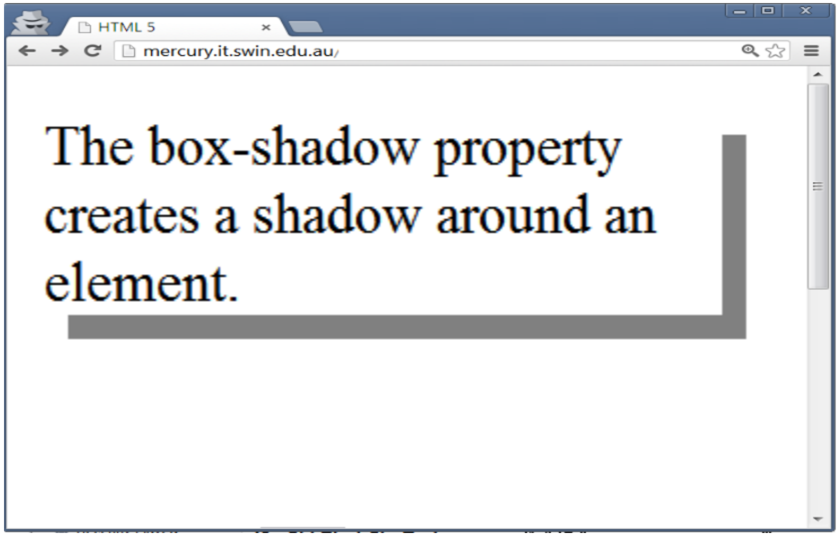
  - div {**background-color**:red;}
  - body {**background-image**:url("bgimg.jpg")}

DEMO!

Model: Box – Shadow

- **box-shadow** : <horizontal> <vertical> <colour values>;
  - box-shadow specifies both the horizontal and vertical measurement and the colour for the shadow

```
p {
   box-shadow :
        .4em .4em grey;
}

<p>The box-shadow
property creates
a shadow around
an element.</p>
```



The box-shadow property creates a shadow around an element.

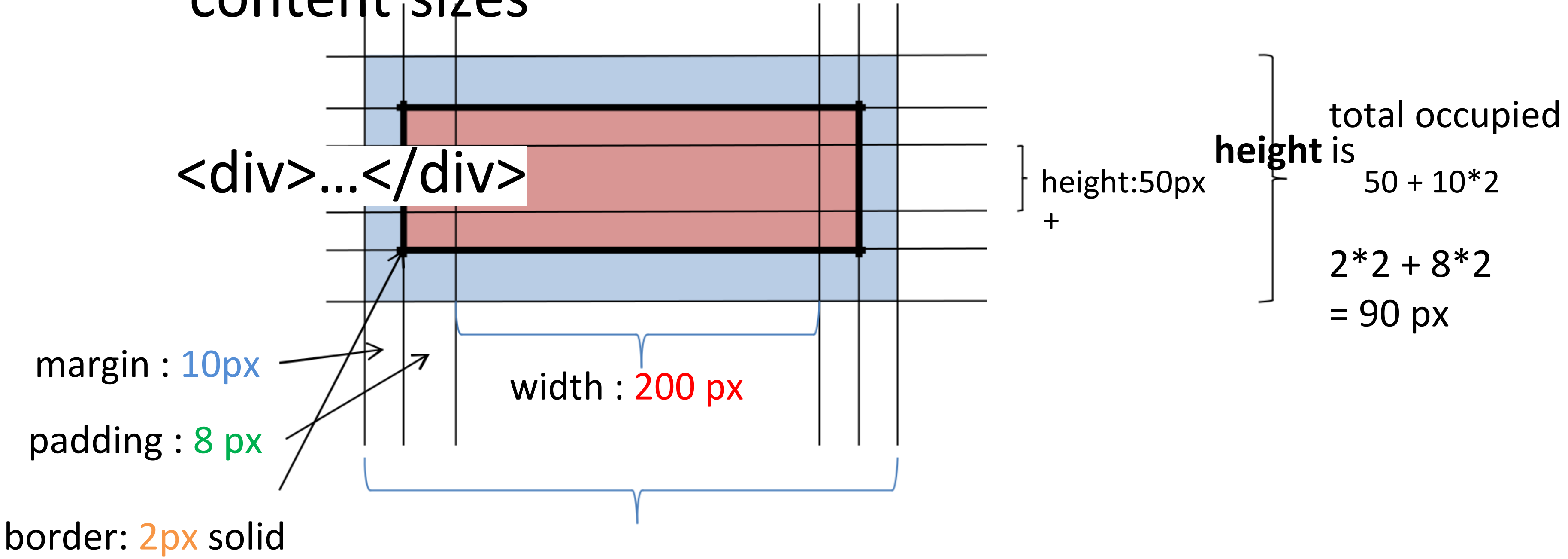
DEMO!

Model: Box – Width and Height

- **width** : <values>;
- **height** : <values>;
  - Width and height specify the horizontal width and vertical height of an element respectively
  - min-width/min-height and max-width/max-height set the allowable values of width and height based on the browser's screen size.
  - Note that desktop browser may be resized by the user.

Model: Box – Width and Height

- Calculating occupied width and height
  - – Factor in the margin, border, padding and content sizes



<div>...</div>

margin : 10px

padding : 8 px

border: 2px solid

width : 200 px

height:50px +

total occupied **height** is 50 + 10*2

2*2 + 8*2

= 90 px

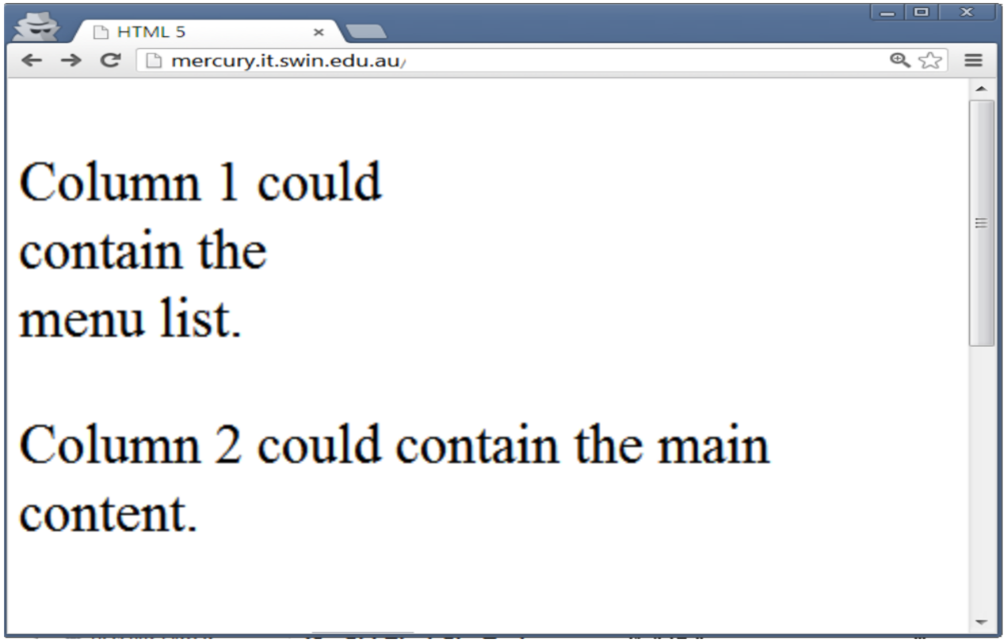
total occupied **width** is 200 + 2*2 + 10*2 + 8*2 = 240 px

width border margin padding

Model: Box – Width and Height

style.css

```
.column1 { width
   : 100px;
}
.column2 { width
   : 250px;
}
```



Column 1 could
contain the
menu list.

Column 2 could contain the main
content.

page.html

```
<p class ="column1">Column 1 could
contain the menu list.</p>
<p class ="column2">Column 2 could
contain the main content.</p>
```

DEMO!

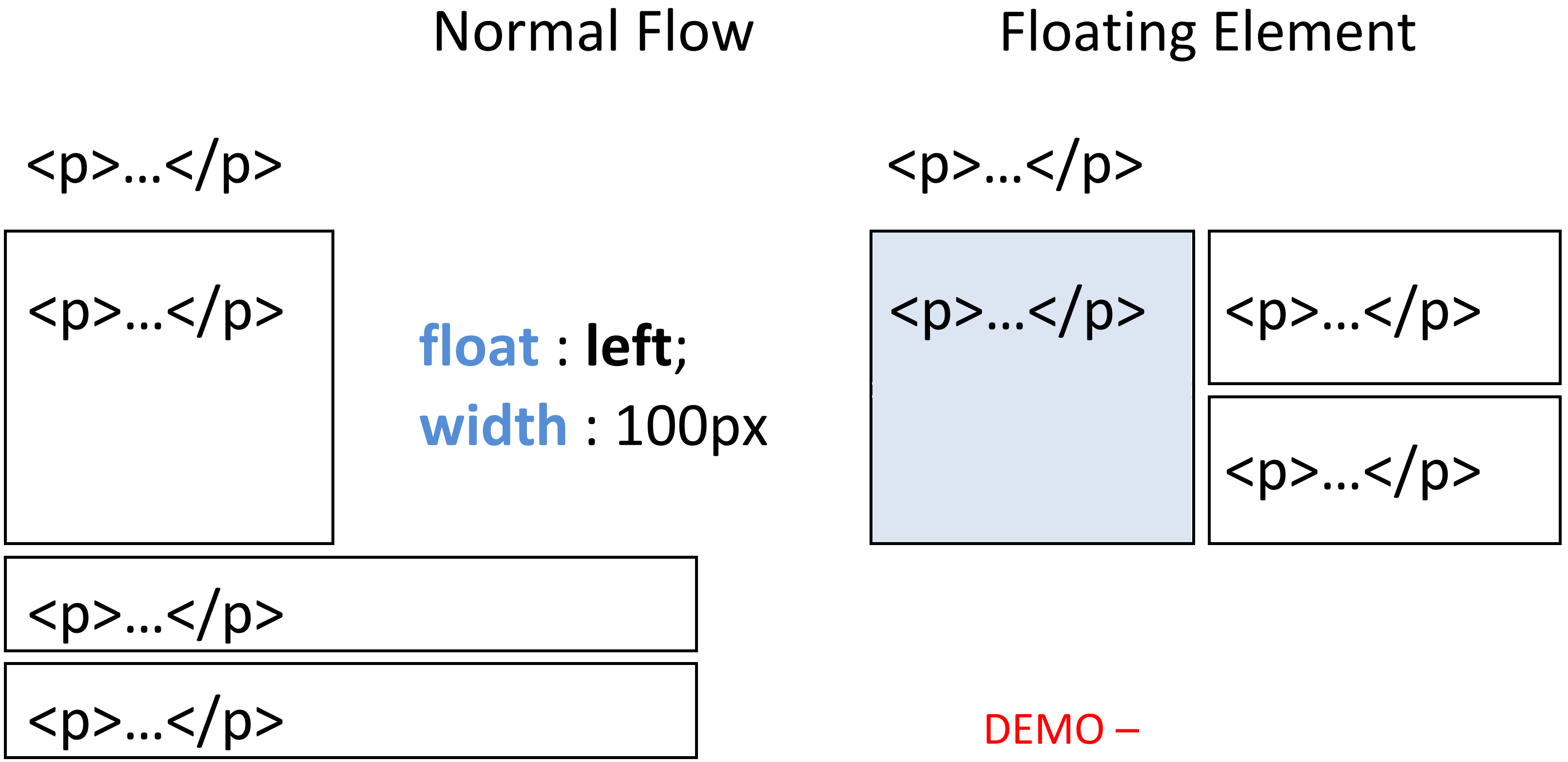
CSS PRESENTATION:
PAGE LAYOUT – A

Page Layout: Flow

- **Normal** is the default browser display of elements, this is one after the other
  - **Block-level** – **vertically** from top to bottom
  - **Inline-level** – **horizontally** from left to right
- CSS property **float** takes an element out of the normal flow
  - Non-floating elements remain in the normal flow

Page Layout: Flow

- **float** : **none** | left | right;



float : **left**;
width : 100px

DEMO – float.html!

Page Layout: Container Elements

- Container elements are used to holds content elements together
- **<div>** can be used as a container element. With HTML5, we can also use **<header>**, **<nav>**, **<article>**, **<section>**, **<aside>** and **<footer>**.
- If the elements inside a container element are floated,
  - margins are used to set the space between elements
  - **the height of the container element is based on the maximum height of a *non-floating* element**

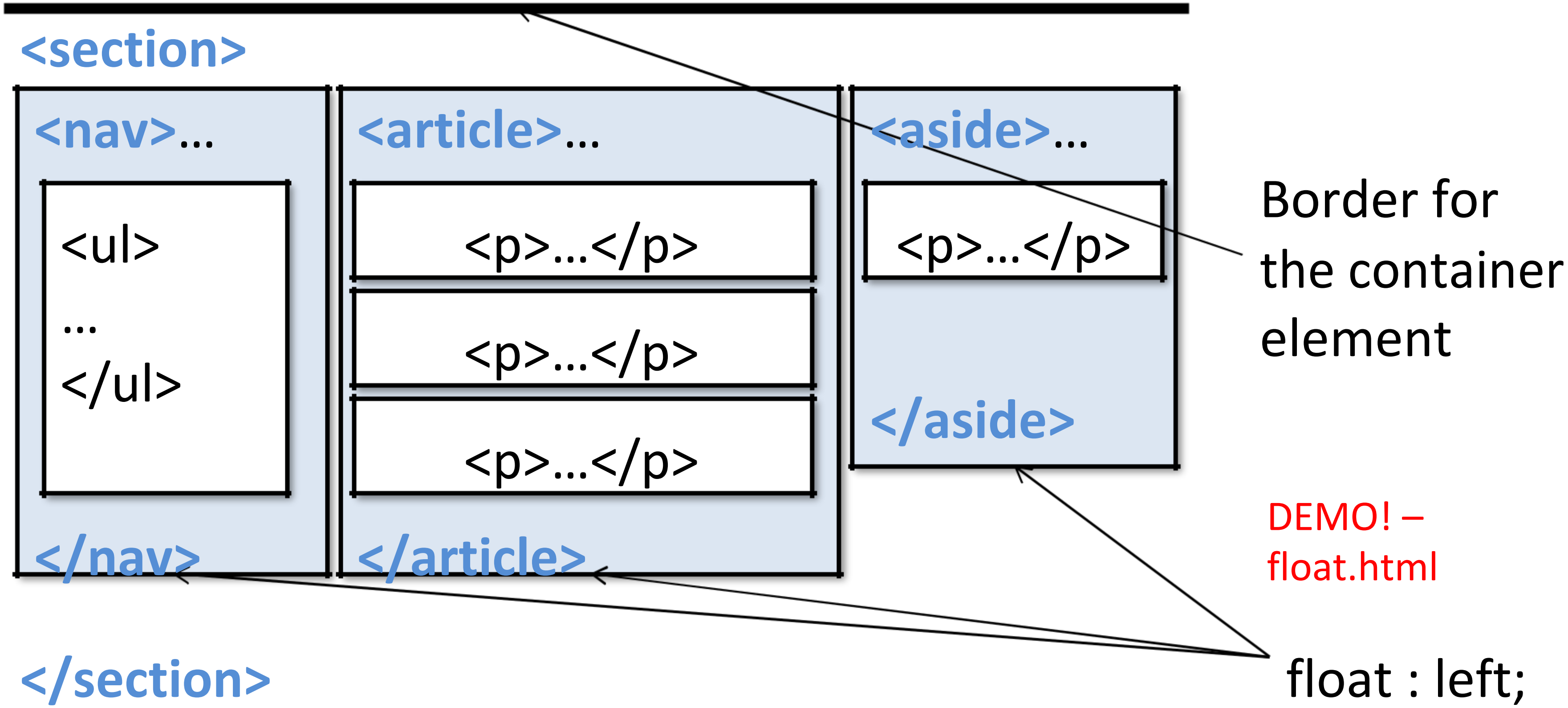
Page Layout: Container Elements

```
<header>
        <h1>…</h1>
</header>

<nav>           <article>              <aside>

  <ul>            <p>…</p>               <p>…</p>
  …               <p>…</p>
  </ul>
                  <p>…</p>

</nav>          </article>             </aside>

<footer>
        <p>…</p>
</footer>
```

only float the container elements

float: left;

Page Layout: Container Element's Height



Floating the three structural elements inside the section element will set its height to 0. This is visually evident if border is displayed.

Page Layout: Solution

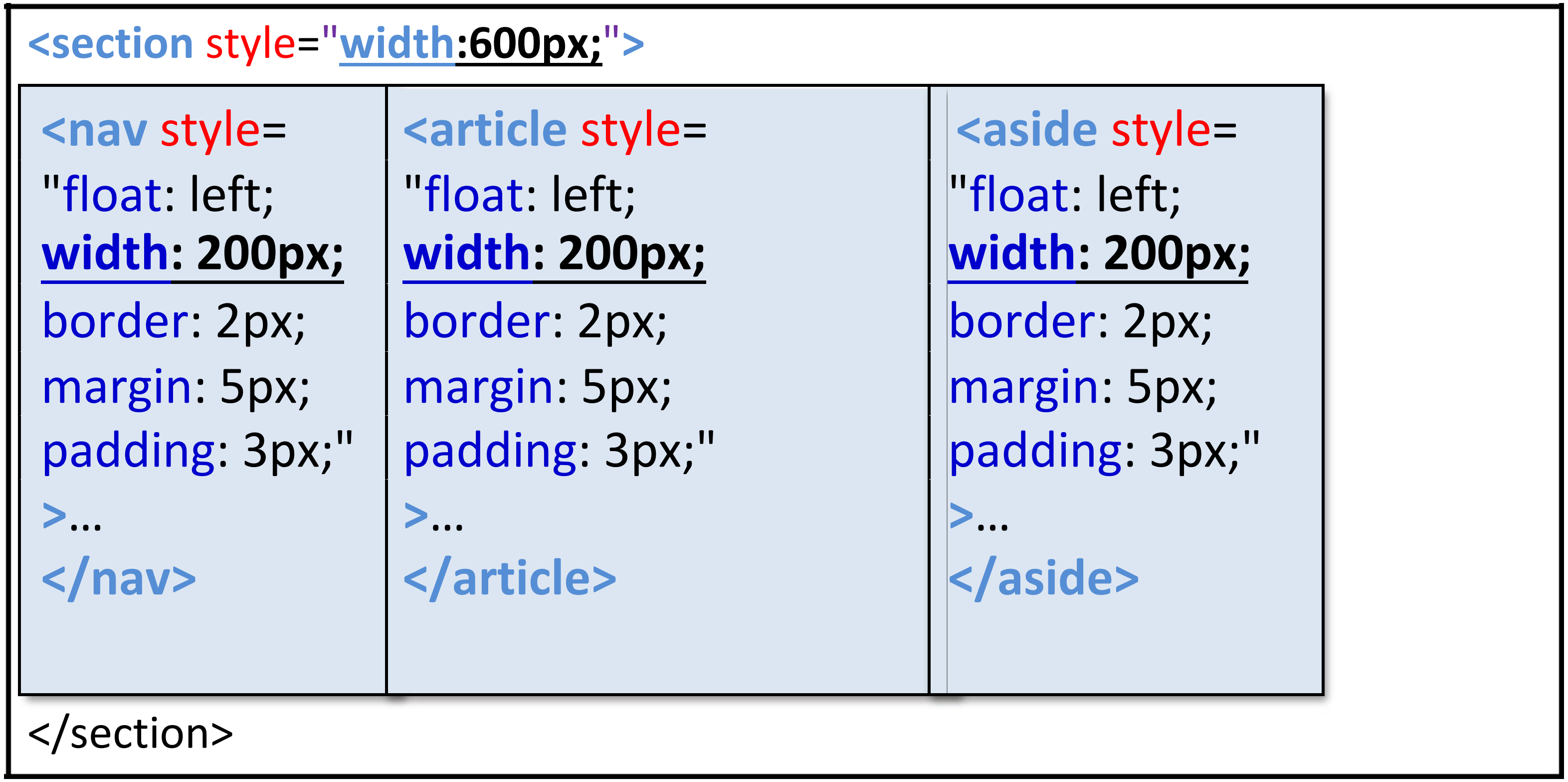
- Specify the CSS property overflow of the container element to **auto**;

```
section {
        overflow: auto;
}
```

DEMO! - float.html

Page Layout: Width and Margin

- Does the combined element width exceeds the container width?



```
<section style="width:600px;">
<nav style=
"float: left;
width: 200px;
border: 2px;
margin: 5px;
padding: 3px;"
>...
</nav>
<article style=
"float: left;
width: 200px;
border: 2px;
margin: 5px;
padding: 3px;"
>...
</article>
<aside style=
"float: left;
width: 200px;
border: 2px;
margin: 5px;
padding: 3px;"
>...
</aside>
</section>
```

Page Layout: Width and Margin

- Yes, and this will result to a drop column.

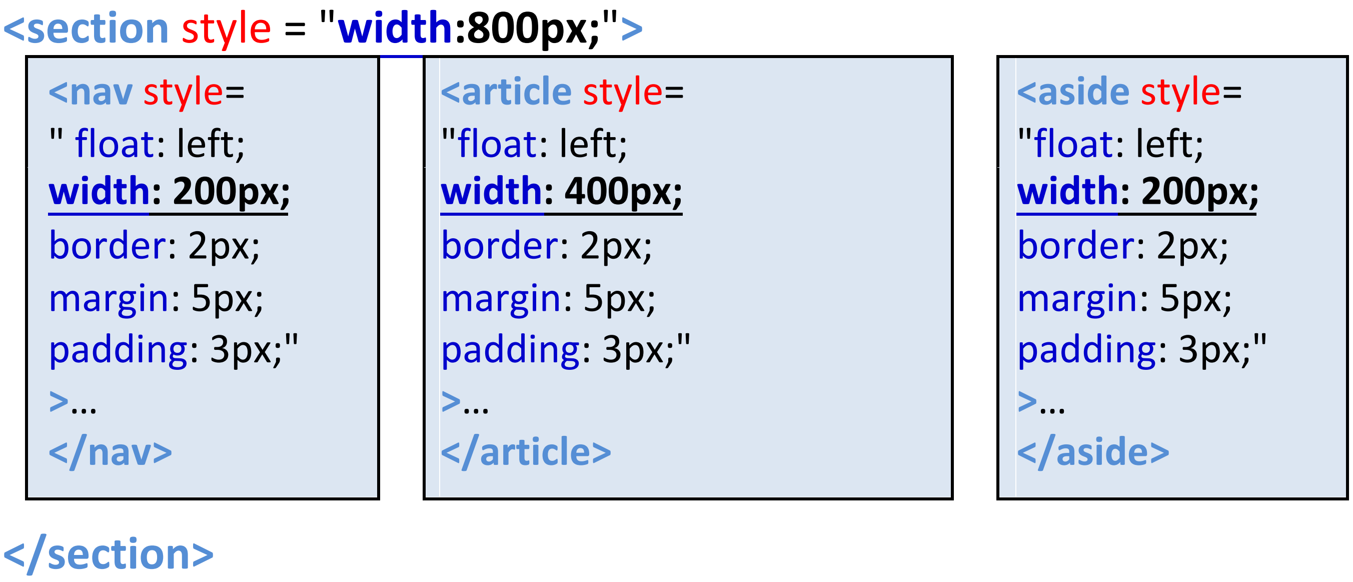
```
<section style = "width:600px;">
    <nav style=
    "float: left;
    width: 200px;
    border: 2px;
    margin: 5px;
    padding: 3px;"
    >…
    </nav>
    <article style=
    "float: left;
    width: 400px;
    border: 2px;
    margin: 5px;
    padding: 3px;"
    >…
    </article>

    <aside style=
    "float: left;
    width: 200px;
    border: 2px;
    margin: 5px;
    padding: 3px;"
    >…
    </aside>
</section>
```

Recall the discussion on computing the occupied width in the slide Model: Box – Width and Height

Page Layout: Width and Margin

- Always include margin, border and padding

$$
\begin{aligned}
&= (200 + (2+5+3) \times 2) + (400 + (2+5+3) \times 2) \\
&\qquad + (200 + (2+5+3) \times 2) \\
&= 860\text{px}
\end{aligned}
$$



```
<section style = "width:800px;">
  <nav style=
  " float: left;
  width: 200px;
  border: 2px;
  margin: 5px;
  padding: 3px;"
  >...
  </nav>

  <article style=
  "float: left;
  width: 400px;
  border: 2px;
  margin: 5px;
  padding: 3px;"
  >...
  </article>

  <aside style=
  "float: left;
  width: 200px;
  border: 2px;
  margin: 5px;
  padding: 3px;"
  >...
  </aside>
</section>
```

Page Layout: Width and Margin

- If using **relative units** for the width, adjust the last column accordingly

25% + 50% + 20% < 100%

with 5% for the margin, border and padding.

```
<section style = "width:100%;">
    <nav style=
    "float: left;
    width: 25%;
    border: 2px;
    margin: 5px;
    padding: 3px;"
    >…
    </nav>

    <article style=
    "float: left;
    width: 50%;
    border: 2px;
    margin: 5px;
    padding: 3px;"
    >…
    </article>

    <aside style=
    "float: left;
    width: 20%;
    border: 2px;
    margin: 5px;
    padding: 3px;"
    >…
    </aside>

</section>
```

Page Layout: Design

- **Fixed layout:** defines exact size of every element in **absolute** units such as pixels.
  - Gives precise control over appearance
  - Does not adapt to the size of the browser window
- **Fluid (Flexible/Liquid) layout:** one or more elements are set with **relative** units.

DEMO!

  - Layout adapts to the size of the browser window.
  - Typically related to **width** rather than height
  - Page content "flows" into free areas of the browser window

Page Layout: Design – **Fixed** #1

- Big window

<header>…
</header>

<nav> <article>… <aside>…

</nav> </article> </aside>

<footer>…
</footer>

Browser window

- Small window

<header>…
</header>

<nav>

</nav>

<article>…

</article>

<aside>…

</aside>

<footer>…
</footer>

Page Layout: Design – **Fixed** #2

- Big window

<header>…
</header>

<nav> <article>… <aside>…

</nav> </article> </aside>

<footer>…
</footer>

Browser window

- Small window

<header>…
</header>

<nav> <article>…

</nav> </article>

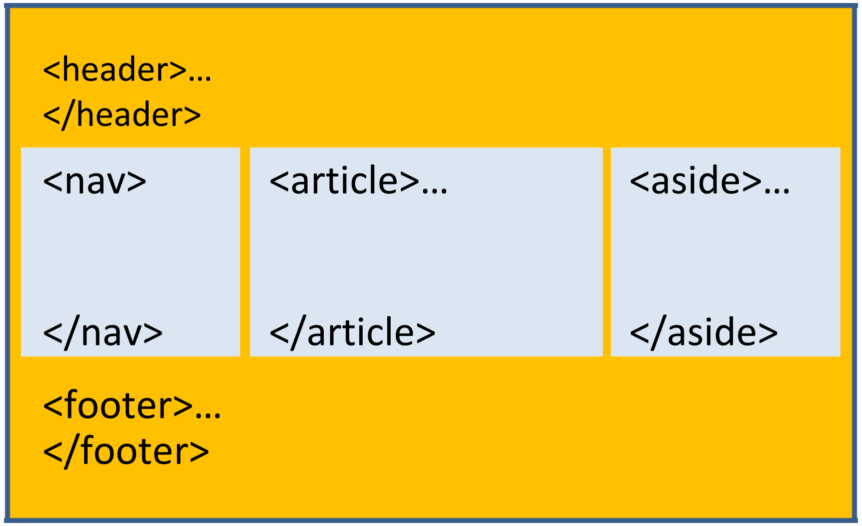
<aside>…

</aside> pushed to the next line

<footer>…
</footer>

Page Layout: Design - Fluid

```
<header
   style="width: 100%;">…
</header>
<nav style="width: 25%;
         float: left;">…
</nav>
<article
  style="width: 50%;
         float: left;">…
</article>
<aside
  style="width: 20%;
         float: left;">…
</aside>
<footer
   style="width: 100%;
         clear: both;">…
</footer>
```



<header>…
</header>

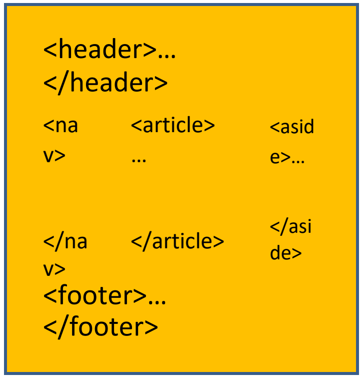
<nav>
</nav>

<article>…
</article>

<aside>…
</aside>

<footer>…
</footer>

- Adapts to the size of the browser window



<header>…
</header>

<na
v>

<article>
…

<asid
e>…

</na
v>

</article>

</asi
de>

<footer>…
</footer>

DEMO!

CSS (PRESENTATION): PAGE LAYOUT – B

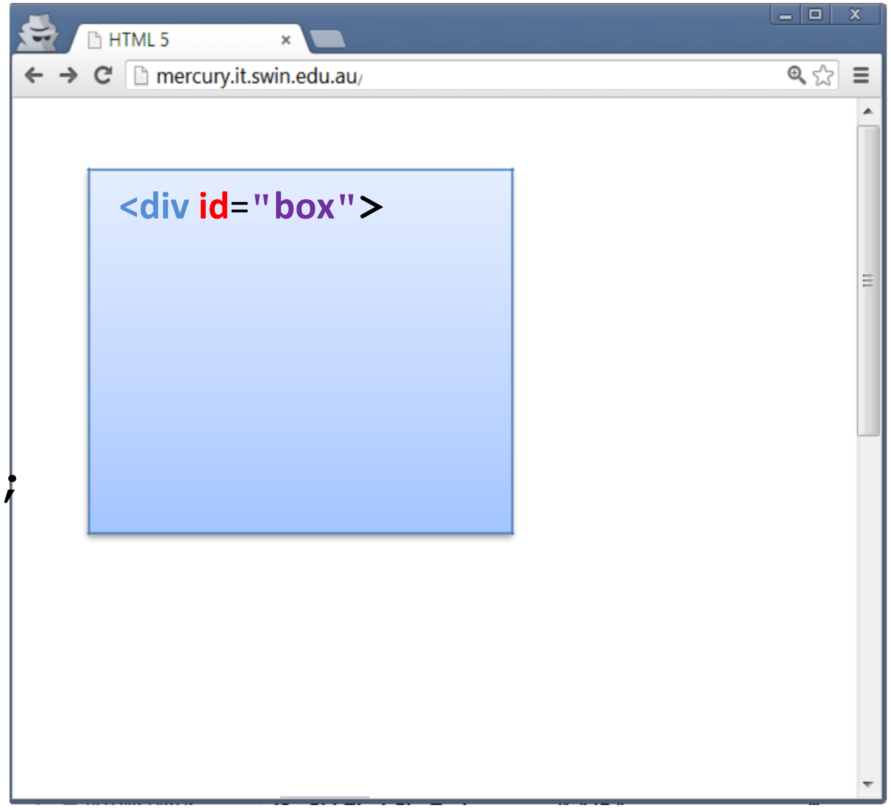
Page Layout: Position, Top and Left

- position: **static** | absolution | fixed | relative;
  - "static" is the default positioning of the elements as they appear in the ***original*** document flow
  - "**relative**" positions the element relative to its normal position, (offsetting from **static**)
  - "**absolute**" positions the element relative to its first positioned ancestor element
  - "**fixed**" positions the element relative to the view port or browser window
- Used with **top|bottom**, **left|right** property

Page Layout: Position, Top and Left

- top: **auto** | <value>;
- left: **auto** | <value>;

```
#box {
   width:100px;
   height:100px;
   border:1px solid #black;
   background-color:skyblue;

   position: absolute;
   top: 100px;
   left: 100px;">
```

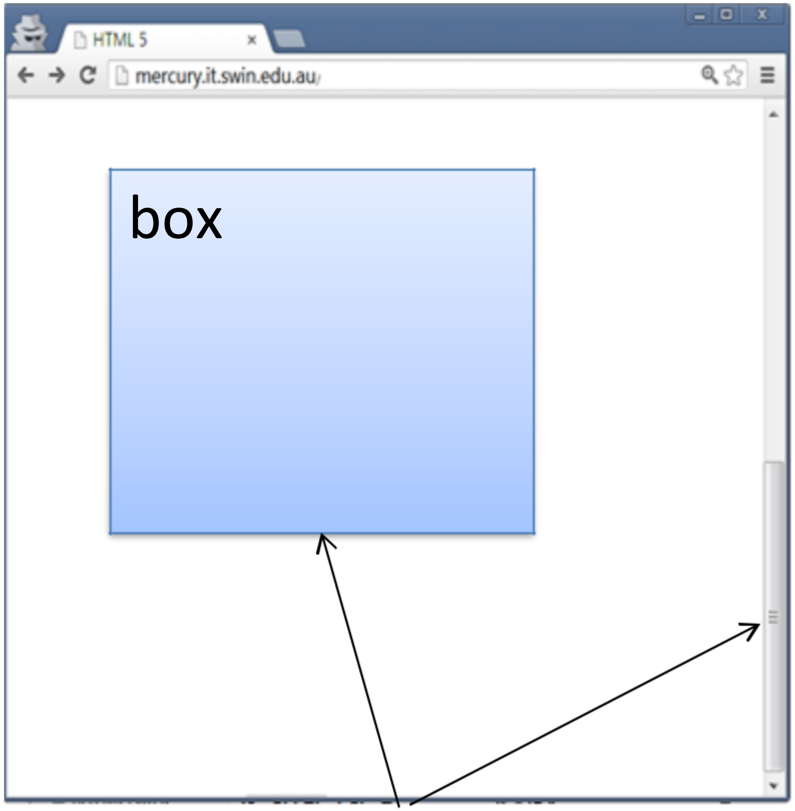


<div id="box">

DEMO!

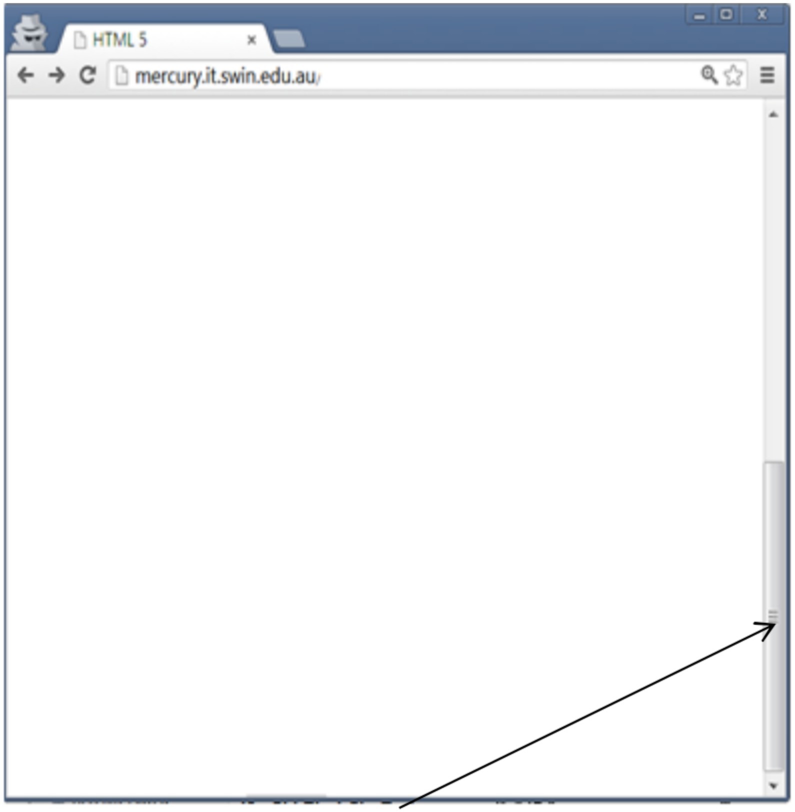
Page Layout: Position, Top and Left

- fixed



Relative to the window, stays on screen
Even if user scrolls down

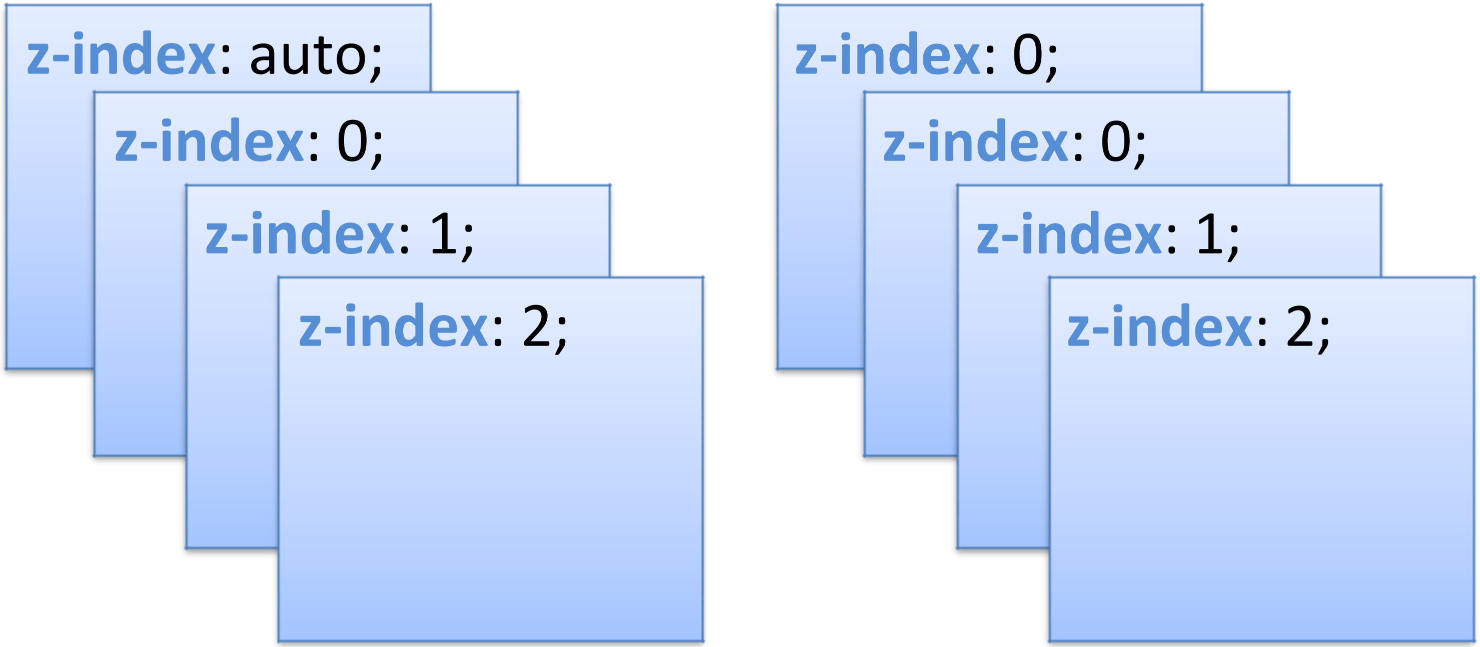
- absolute



Relative to the page, scrolls with the webpage

Page Layout: z-index

- **z-index** : **auto** | <number>;
  - – Modifies the stacking order of the elements



z-index: auto;

z-index: 0;

z-index: 1;

z-index: 2;

z-index: 0;

z-index: 0;

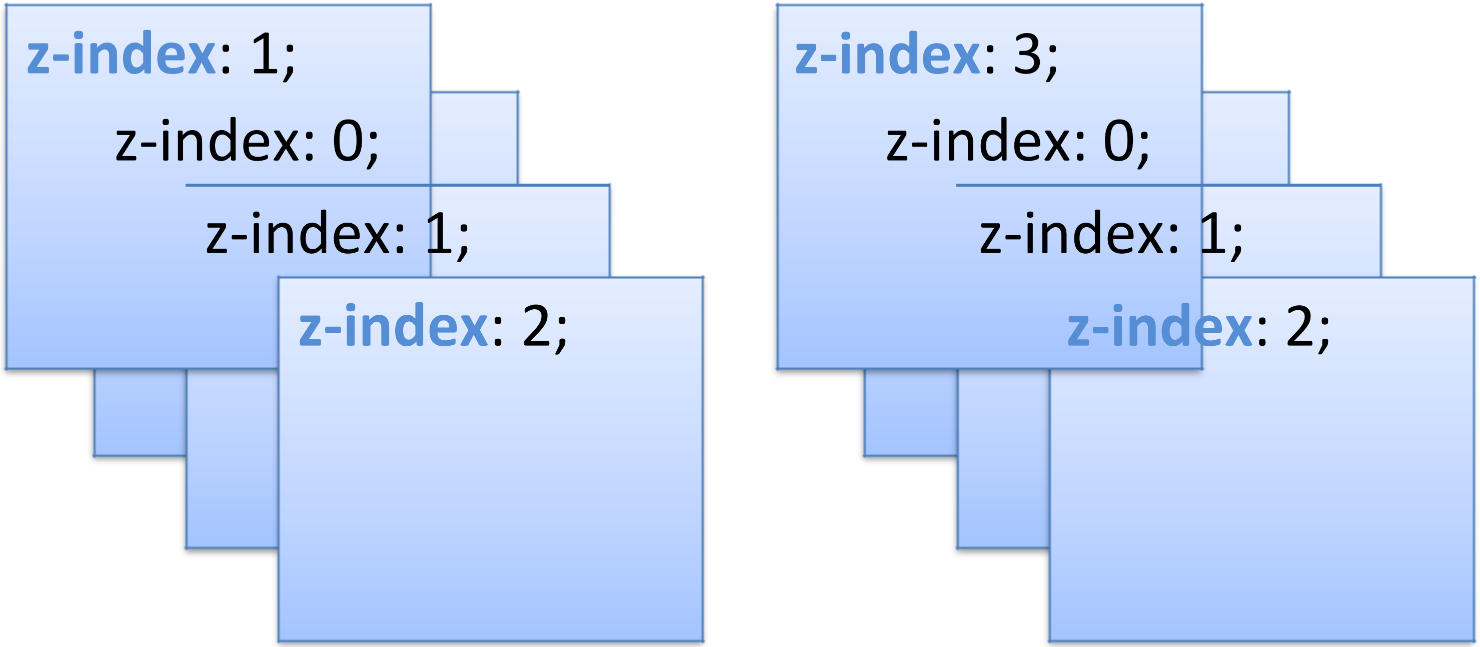
z-index: 1;

z-index: 2;

DEMO!

Page Layout: z-index

- Stacking order of elements with the same z-level value is based on the order in the HTML text



NEXT LECTURE:

CODING FOR OTHER DEVICES
USABILITY, ACCESSIBILITY