# COS30043
# Interface Design and Development

Lecture 2 - Layout and Grid System

1

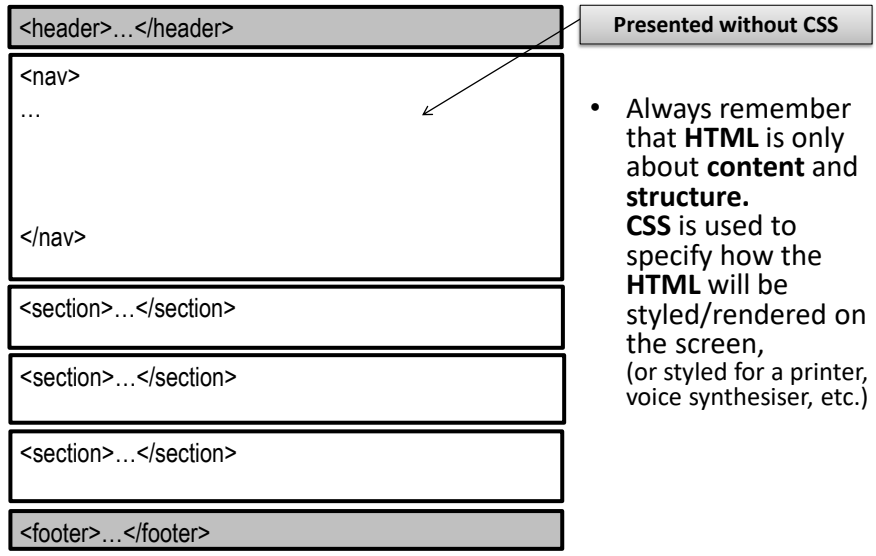
## Topics

- Layout: Elements
- Layout: Page
- Bootstrap – Introduction and How to Use It
- Bootstrap - Grid System



2

# HTML Structure




- Always remember that **HTML** is only about **content** and **structure.**
  **CSS** is used to specify how the **HTML** will be styled/rendered on the screen,
  (or styled for a printer, voice synthesiser, etc.)

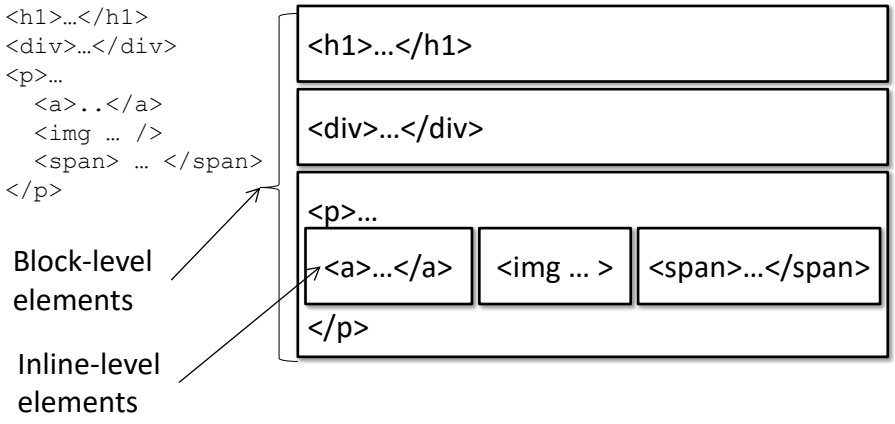


# Understanding Layout

- Visual formatting model describes how the element content boxes should be displayed
  - **Block-level elements** appear as blocks *such as paragraphs*
  - **Inline-level elements** are contained within block-level elements, *such as anchors*
- ***Box model*** describes the rectangular boxes that contain content on a web page

# Understanding Layout (Flow)

- Flow is from top to bottom left to right according to how the elements are ordered



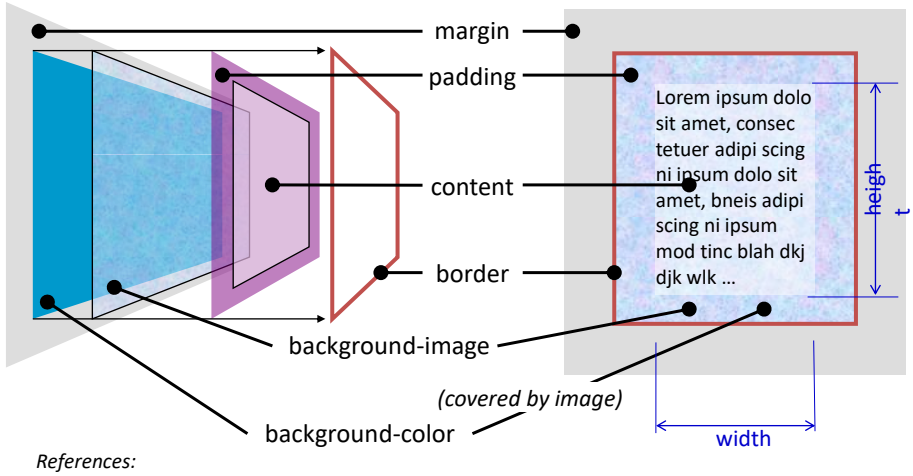



# Understanding Layout (Display)

- display : inline | block | list-item | inline-block | table | inline-table | table-row-group | table-header-group | table-footer-group | table-row | table-column-group | table-column | table-cell | table-caption | none
  - display: block
    used to change an inline element to a block level element,
  - display: inline
    used to change a block level element to an inline element
  - display: table values
    used to create table-like displays using CSS
    (HTML tables are only for tabular data)
  - display: none value hides the element from display

http://css-tricks.com/almanac/properties/d/display/

# Understanding Layout (Box Model)




*References:*

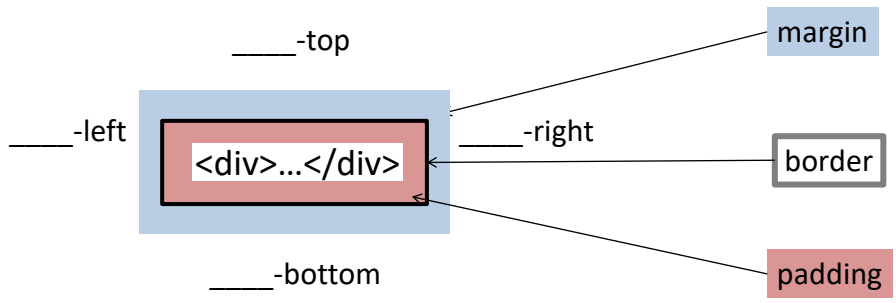
- CSS2.1 Box Model http://www.w3.org/TR/CSS2/box.html
- CSS Backgrounds and Borders Module Level 3 http://www.w3.org/TR/css3-background/





# Understanding Layout (Box Model)

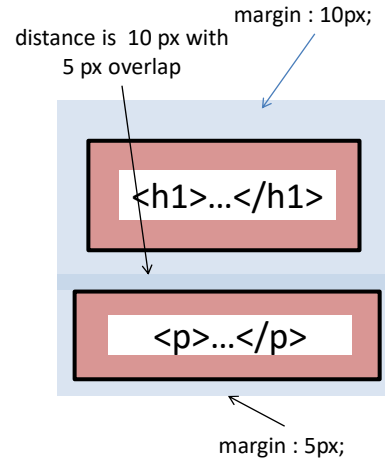
- Spaces are on the margin, padding and border
- Settings are done using shorthand notation  individual notation






# Understanding Layout (Box Model)

- Margin is the space outside the element's border.
- Collapsing margin is where the minimum separation distance is between element's border
- Margin do not collapse on
  - float,
  - absolutely positioned elements,
  - inline-block elements,
  - elements with overflow set to anything other than visible, and
  - root element

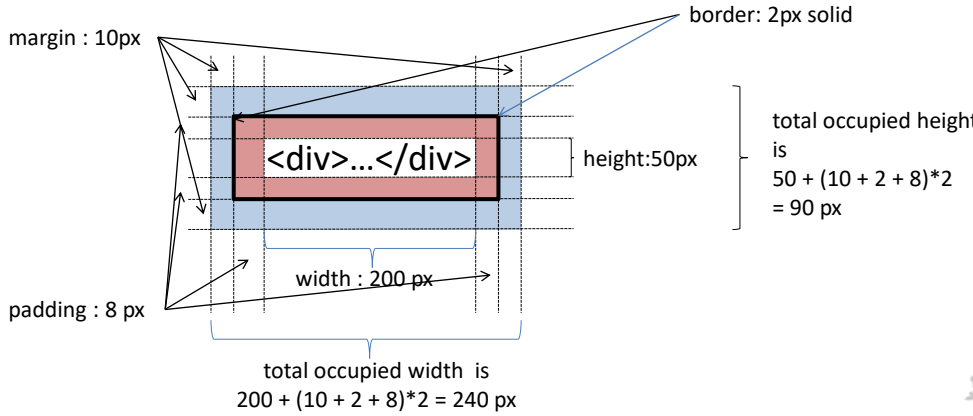







# Understanding Layout (Box Model)

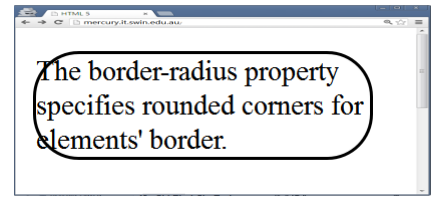
- Calculating occupied width and height
  - Factor in the border, margin and padding sizes



# Understanding Layout (Box Model)

- Remember to adjust the padding accordingly, when using rounded corners

```
<style type="text/css">
p {
 border : 1px solid;
 border-radius:25px;
}
</style>

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








# Understanding Layout (Float&Clear)

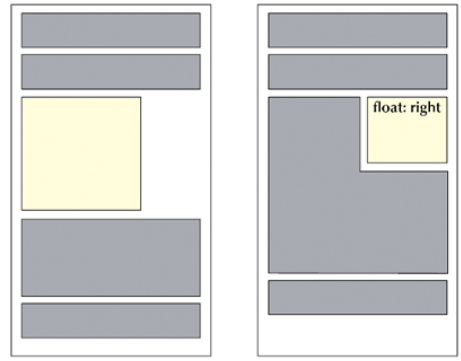
- **`float:`**
  `left, right, none`
  - Set an element to **`float`** against the parent border. Other block positions are unaffected, but block contents (eg. text) will flow around the floated element.
- **`clear:`**
  `left, right, both, none`
  - The **`clear`** property lets you position elements "clear" from other "floated" elements.

    Example: Make sure that the next "intro" paragraph is clear, both left and right, from any floated images:

```
p.intro {
  clear: both;
}
```

# CSS Element Layout Example

- Float example, clear example



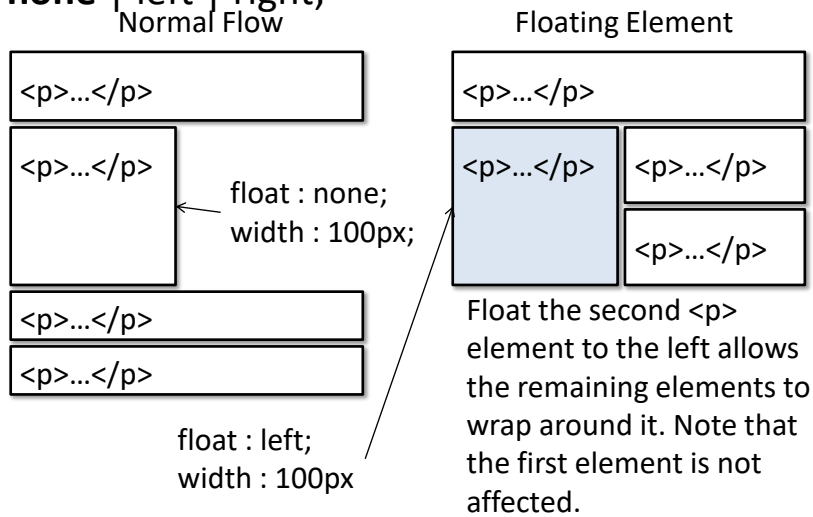

***See also CSS Page Layout notes. eg.'float' div blocks into columns***
http://css.maxdesign.com.au/floatutorial/





# Understanding Layout (Float)

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




Float the second <p> element to the left allows the remaining elements to wrap around it. Note that the first element is not affected.

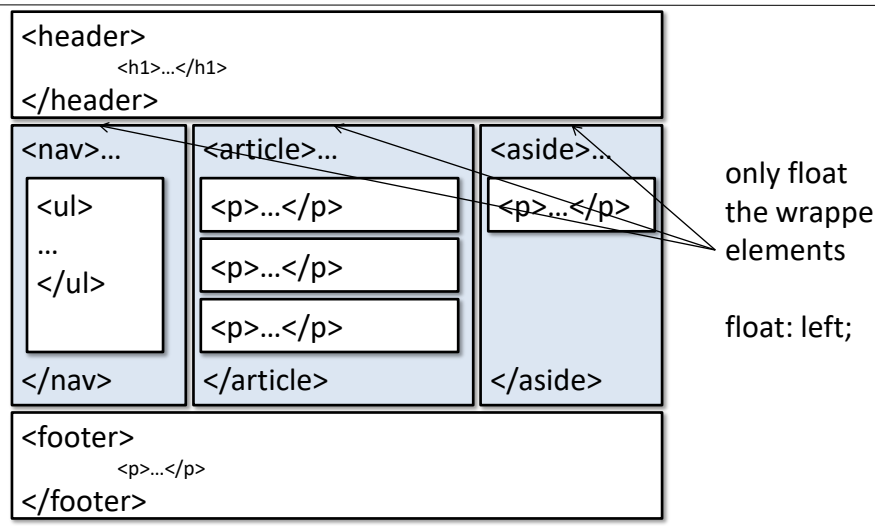


# Understanding Layout (Wrapper)

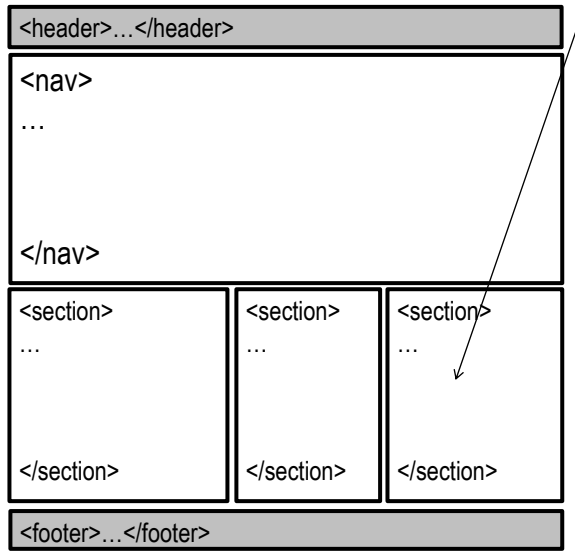
- Wrapper elements are used to hold page pieces together
- Wrapper elements can be . With HTML5, these are <header>, <nav>, <article>, <main>, <section>, <aside> and <footer>
- If elements inside the wrapper element are floated,
  - margins are used to set the gutters between elements
  - the height of the wrapper is based on the maximum height of a non-floating element



# Understanding Layout (Wrapper)




# Layout Example #1

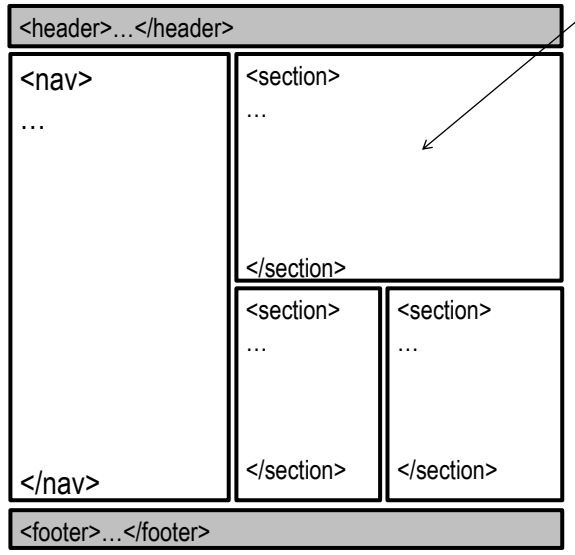



Float all?

- How can you have this layout?



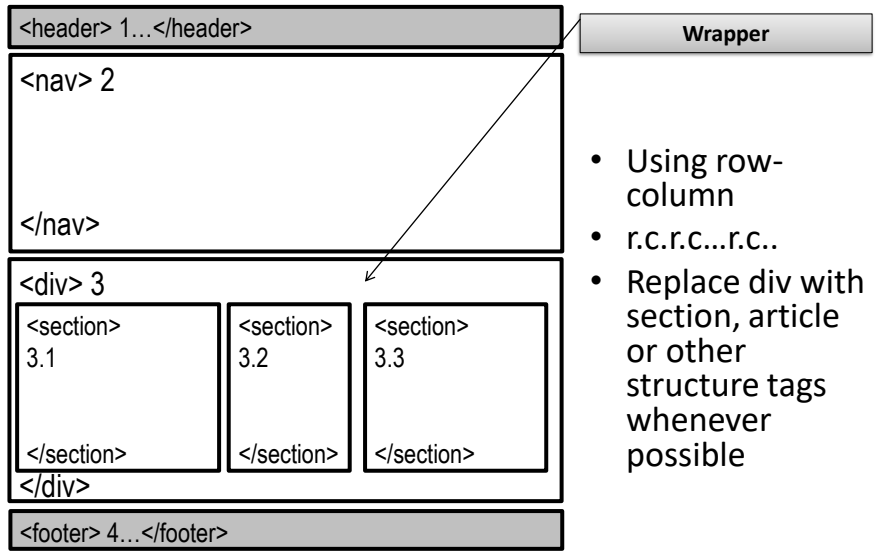
# Layout Example #2




Which do we float?

- How can you have this layout?

# Layout Example #1 – Improved

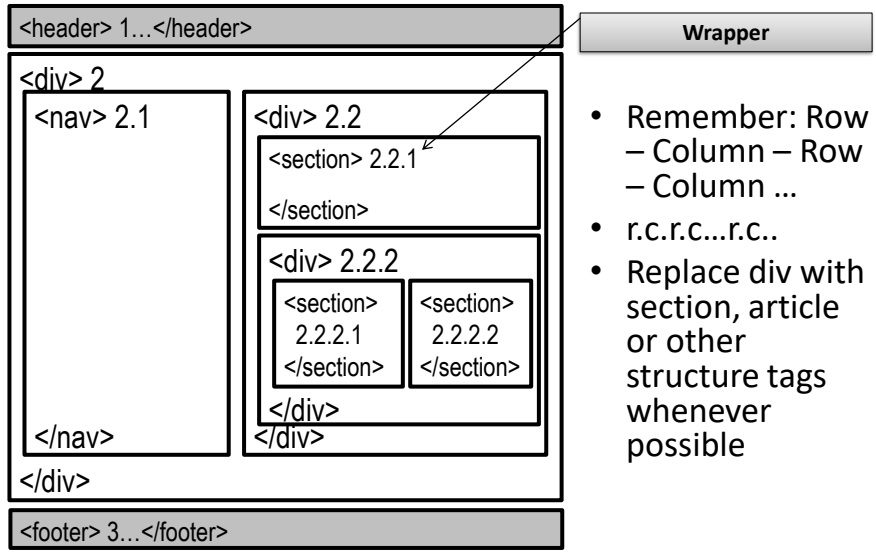



- Using row-column
- r.c.r.c…r.c..
- Replace div with section, article or other structure tags whenever possible





# Layout Example #2 – Improved




- Remember: Row – Column – Row – Column …
- r.c.r.c…r.c..
- Replace div with section, article or other structure tags whenever possible

# Layout Using CSS

- Separate content from presentation
- Easier to maintain large projects
- Provides more control than just HTML
- Supports different user needs
- Supports different presentation alternatives
- Supports device independence
- Supports device specific styles





# Layout Using CSS (Selectors)

| Selector | Description | Example |
|---|---|---|
| `element` | Applies the style rule to *all elements* that match the **element name.**<br>*Also called "**tag style**"* | `h1 {color: green;}` |
| `#id` | Applies the rule *only for the **single** element* that has this **id value.**<br>eg. <tag id="info"><br>*Also called "**id style**"* | `#info { background-color: red; }` |
| `.class` | Applies the rule to *any elements* that have the matching **class value.**<br>eg. <<tag class="note"><br>*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 | `p.note { border: 1px solid blue;}` |

## Example: Selector (id)

- Apply to an HTML **id** attribute

```
<p id = "copyright" >…</p>
```

- Select by using the id name prefixed with a hash "#"

```
#copyright { color : red; }
```

- An `id` ***must be unique*** in a webpage, so this selector will apply style rules to **only one** element in the page.



## Example: Selectors (class)

- Apply an HTML **class** attribute

```
<p class = "story" >…</p>
<h2 class="story">HTML Introduction</h2>
```

- Select by using the class name prefixed with a dot "."

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

- The `class` can be added to many elements, so one class style can be applied many times on a page.
- Can be made element specific by adding an element selector before the dot "."

```
p.story { color : blue; }
```

## Layout: Element Positioning

- In CSS 2.1, a box may be laid out according to three positioning schemes: *Reference: http://www.w3.org/TR/CSS21/visuren.html*
  - Normal flow
    Includes block formatting of block-level boxes, inline formatting of inline-level boxes, and relative positioning of block-level and inline-level boxes.
  - Floats
    In the float model, a box is first laid out according to the normal flow, then taken out of the flow and shifted to the left or right as far as possible. Content may flow along the side of a float.
  - Absolute positioning
    In the absolute positioning model, ***a box is removed from the normal flow entirely*** (it has no impact on later siblings) and assigned a position with respect to a containing block.





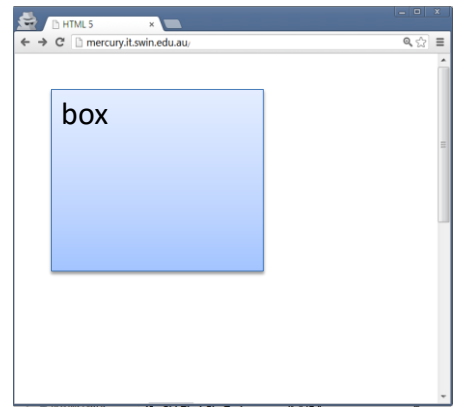
## Layout: Position, Top and Left

- position: **static** | absolute | fixed | relative;
  - static is the default positioning of the elements as they appear in the 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 and left property
- Avoid `position:` unless really needed

# Layout: Position, Top and Left

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

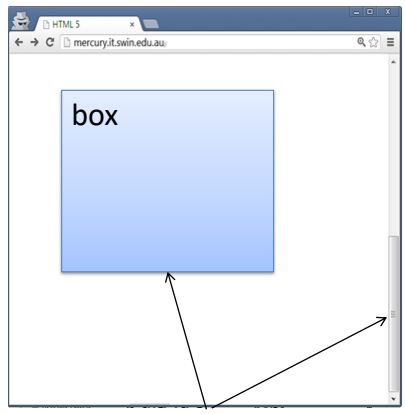
```
<div style="
width:100px;height:100px;
border:1px solid #black;
background-color:skyblue;

position:absolute;
top:100px;left:100px;">
box</div>
```



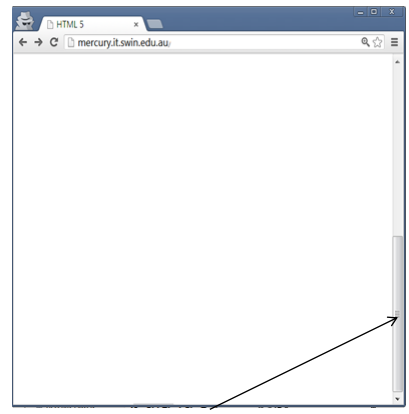



# Layout: Position, Top and Left

- fixed
- absolute



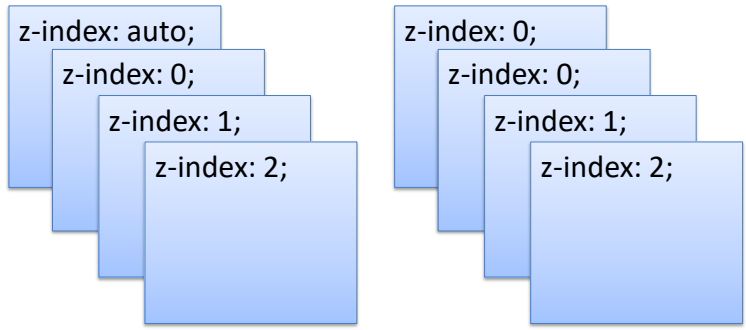

Relative to the window, stays on screen
Even if user scrolls down




Relative to the page, scrolls with the webpage

## Layout: z-index

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

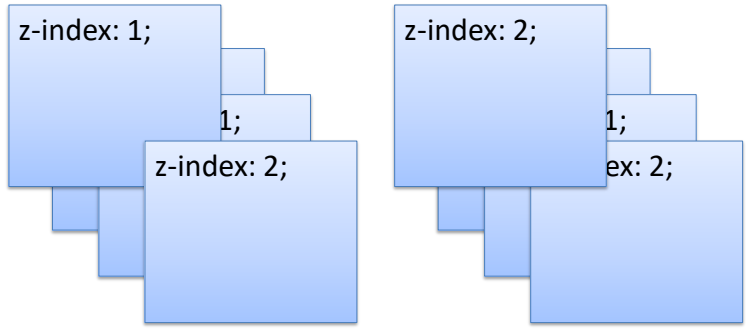







## Layout: z-index

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



## Topics

- Layout: Elements
- Layout: Page
- Bootstrap – Introduction and How to Use It
- Bootstrap - Grid System





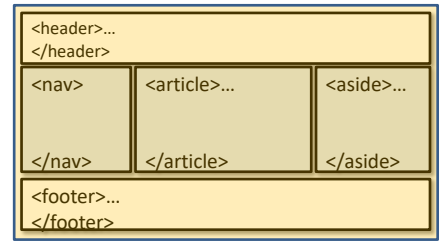
## Layout: Page Design

- **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.
  - 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

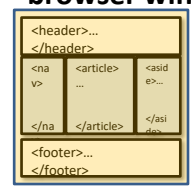
# Layout: Page Design - Fluid

```
<header >…
 </header>
<nav >…
 </nav>
<article>…
 </article>
<aside>…
 </aside>
<footer>…
 </footer>
```

```
header {width:100%;}
nav {width:25%;  float:left;}
article {width:50%; float:left;}
aside {width:20%; float:left;}
footer {width:100%; clear:both;}
```




**Adapts to the size of the browser window**






# Designing for different devices

- "Responsive design" layout adjust as user changes orientation of a mobile device, changes screen resolution, window size.
- "Mobile first"
  - Determine what is most important based on content
  - Design layout to start with smaller width first, using media queries
  - Adds elements as screen size increases

## Topics

- Layout: Elements
- Layout: Page
- Bootstrap – Introduction and How to Use It
- Bootstrap - Grid System





## What is Bootstrap?

- Bootstrap was developed by Mark Otto and Jacob Thornton at Twitter, and released as an open source product in August 2011.
- Bootstrap is a free front-end framework for faster and easier web development
- Bootstrap contains CSS- and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components.
- Bootstrap also gives you the ability to easily create responsive designs

## Bootstrap

- Supports a responsive mobile first fluid system
- Scales up to 12 columns as the device or viewport size increases
- Provides predefined layout (grid) classes
- Uses rows to create horizontal groups of columns
- The current version is Version 5



## Advantages of Bootstrap

- Advantages of Bootstrap:
  - **Easy to use:** Anybody with just basic knowledge of HTML and CSS can start using Bootstrap
  - **Responsive features:** Bootstrap's responsive CSS adjusts to phones, tablets, and desktops
  - **Mobile-first approach:** In Bootstrap 3, mobile-first styles are part of the core framework
  - **Browser compatibility:** Bootstrap is compatible with all modern browsers (Chrome, Firefox, Internet Explorer, Safari, and Opera)

# How to Use Bootstrap?

- There are two ways to start using Bootstrap on your own web site.
  - Use Bootstrap CDN (Content Delivery Network)
    - When you only need to include Bootstrap's compiled CSS or JS, you can use jsDelivr (A free CDN for Open Source).
  - Download Bootstrap from getbootstrap.com
    - If you want to download and host Bootstrap yourself, go to getbootstrap.com, and follow the instructions there.



Use Bootstrap Method 1

# Bootstrap CDN Example

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet"
        integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
        crossorigin="anonymous">
    <title>Hello, world!</title>
  </head>
  <body>
    <h1>Hello, world!</h1>
    Your content here.
    <!-- Optional JavaScript -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
        integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
        crossorigin="anonymous"></script>
  </body>
</html>
```

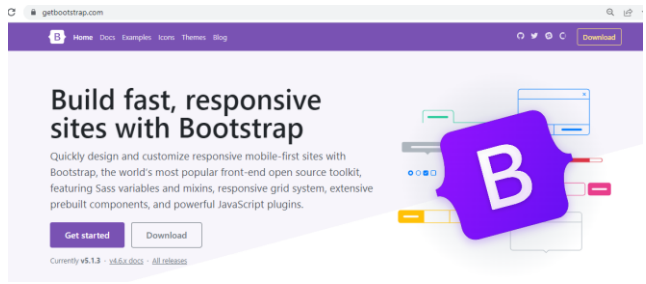
The width=device-width part sets the width of the page to follow the screen-width of the device (which will vary depending on the device).

The initial-scale=1 part sets the initial zoom level when the page is first loaded by the browser.

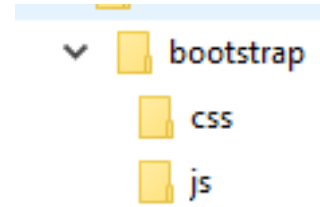
Use Bootstrap Method 2

# Download Bootstrap

- https://getbootstrap.com/docs/5.1/getting-started/introduction/




- Unzip the files to a proper folder locally, for example








Use Bootstrap Method 2

# Use the Downloaded Bootstrap Locally

```
<!DOCTYPE html>
<html lang="en">
   <head>
      <title>Template that uses Bootstrap</title>
      <meta charset="utf-8" />
      <meta name="viewport" content="width=device-width,
                initial-scale=1.0" />
      <!-- Bootstrap -->
      <link href="bootstrap/css/bootstrap.min.css" rel="stylesheet" />
   </head>
   <body>
      <h1>Hello world!</h1>

      Your content here.

      <!-- Bootstrap javascript plug-ins -->
      <script src="bootstrap/js/bootstrap.bundle.min.js"></script>
   </body>
</html>
```

# Bootstrap CSS Classes

**There are many Bootstrap CSS classes we can use. For example**

- **Text Colors**

```
<p class="text-primary">This text is important.</p>
<p class="text-success">This text indicates success.</p>
<p class="text-info">This text represents some information.</p>
<p class="text-warning">This text represents a warning.</p>
<p class="text-danger">This text represents danger.</p>
```

This text is important.

This text indicates success.

This text represents some information.

This text represents a warning.

This text represents danger.

- **Background Colors**

```
<p class="bg-success ">This text indicates success.</p>
<p class="bg-info ">This text represents some information.</p>
```

This text indicates success.

This text represents some information.

- **Buttons, Tables, and etc**





# Topics

- Layout: Elements
- Layout: Page
- Bootstrap – Introduction and How to Use It
- Bootstrap - Grid System

# Grid System

- Organise and structure content
- Enables easy scanning
- Reduces cognitive load on users



45

# Breakpoint

https://getbootstrap.com/docs/5.0/layout/breakpoints/

| Breakpoint | Class infix | Dimensions |
|---|---|---|
| X-Small | *None* | <576px |
| Small | `sm` | ≥576px |
| Medium | `md` | ≥768px |
| Large | `lg` | ≥992px |
| Extra large | `xl` | ≥1200px |
| Extra extra large | `xxl` | ≥1400px |



46

## Bootstrap: Grid Structure

```
<div class="container">
   <div class="row">
      <div class="col-*-*"></div>
      <div class="col-*-*"></div>
   </div>
   <div class="row">...</div>
</div>
<div class="container">....
```



## Container

Containers are used to contain, pad, and align the content within them. Bootstrap comes with three different containers:

- .container, which sets a max-width at each responsive breakpoint
- .container-fluid, which is width: 100% at all breakpoints
- .container-{breakpoint}, which is width: 100% until the specified breakpoint

# Container

https://getbootstrap.com/docs/5.0/layout/containers/

| | Extra small <576px | Small ≥576px | Medium ≥768px | Large ≥992px | X-Large ≥1200px | XX-Large ≥1400px |
|---|---|---|---|---|---|---|
| .container | 100% | 540px | 720px | 960px | 1140px | 1320px |
| .container-sm | 100% | 540px | 720px | 960px | 1140px | 1320px |
| .container-md | 100% | 100% | 720px | 960px | 1140px | 1320px |
| .container-lg | 100% | 100% | 100% | 960px | 1140px | 1320px |
| .container-xl | 100% | 100% | 100% | 100% | 1140px | 1320px |
| .container-xxl | 100% | 100% | 100% | 100% | 100% | 1320px |
| .container-fluid | 100% | 100% | 100% | 100% | 100% | 100% |





# Columns

Each row is divided into 12 columns.
For example, assuming all columns are of equal width

| To have | we write |
|---|---|
| 12 columns | col-xx-1 |
| 6 columns | col-xx-2 |
| 4 columns | col-xx-3 |
| 3 columns | col-xx-4 |
| 2 columns | col-xx-6 |
| 1 column | col-xx-12 |

xx can be sm, md, lg, xl, xxl.
For example, col-md-4

No breakpoint abbreviation for extra small. For example, col-4

Each class scales up, so if you wish to set the same widths for md and lg, you only need to specify md.

# Example #1: Stacked-to-horizontal

```
<div class="container">
  <div class="row">
    <div class="col-12">
       <h1>50-50 Split</h1>
    </div>
  </div>
  <div class="row">
      <div class="col-md-6">
         <p>Product A</p>
         <p>Photo of Product A</p>
      </div>
      <div class="col-md-6">
         <p>Product B</p>
         <p>Photo of Product B</p>
      </div>
   </div>
</div>
```



# Example #2: Stacked-to-horizontal

```
<div class="container">
  <div class="row">
    <h1>50-50 @ medium or 25-75 @ large</h1>
  </div>
  <div class="row">
    <div class="col-md-6 col-lg-3">
         <p>Author</p>
         <p>Photo of Author</p>
    </div>
    <div class="col-md-6 col-lg-9">
         <p>Paragraph 1 of article</p>
         <p>Paragraph 2 of article</p>
    </div>
   </div>
</div>
```

# Example #3:Responsive Column Resets

```
<div class="container">
   <div class="row">
      <div class="col-sm-6 col-md-3" >
         <p>Item 1</p>
      </div>
      <div class="col-sm-6 col-md-3" >
         <p>Item 2</p>
      </div>
      <div class="col-sm-6 col-md-3">
         <p>item 3</p>
      </div>
      <div class="col-sm-6 col-md-3">
         <p>Item 4</p>
      </div>
   </div>
</div>
```

53 - Interface Design and Development, © Swinburne

53

# Example #4: Nesting Columns

```
<div class="container">
   <div class="row">
      <div class="col-md-3">
         <h1>First Column</h1>
      </div>
      <div class="col-md-9">
        <div class="row">
            <h2>Second Column - contains 2 row,
                  with 2 columns each showing 4 boxes</h2>
        </div>
        <div class="row">
            <div class="col-md-6">
               <p>Product A</p>
            </div>
            <div class="col-md-6">
               <p>Photo of Product A</p>
            </div>
        </div>
        <div class="row">
            <div class="col-md-6">
               <p>Product B</p>
            </div>
            <div class="col-md-6">
               <p>Photo of Product B</p>
            </div>
        </div>
      </div>
   </div>
</div>
```

54 - Interface Design and Development, © Swinburne

54

## Example #5: Column Ordering

```
<div class="container">
  <div class="row">
      <!-- No Ordering -->
      <div class="col-md-4">
         <p>Left column</p>
      </div>
      <div class="col-md-8">
         <p>Right column</p>
      </div>
   </div>
   <div class="row">
      <!-- With Ordering -->
      <div class="col-md-4 col-md-push-8">
         <p>Left column displayed on the right</p>
      </div>
      <div class="col-md-8 col-md-pull-4">
         <p>Right column displayed on the left</p>
      </div>
   </div>
</div>
```



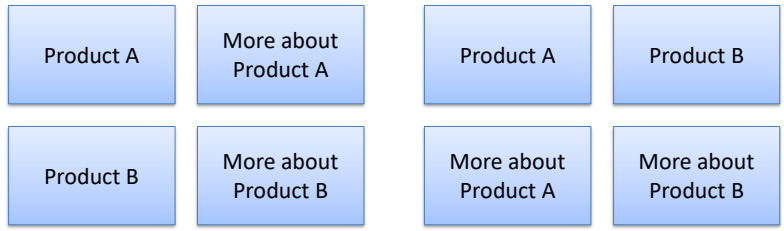
## Bootstrap: Row-Column Format

- Always observe the row-column format
- Remember to use "col-12" for all single column row

```
<div class="row">
  <div class="col-12">
    <h1>Hello World</h1>
  </div>
</div>
```

## Bootstrap: Context Grouping

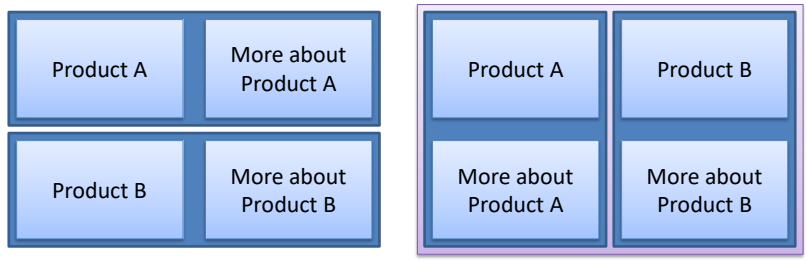
- Visually they look the same
- But contextually they are not






## Bootstrap: Context Grouping

- Design must be based on context not visual
- For example




2 rows with 2 columns

2 columns with 2 rows

# Bootstrap: Context Grouping

```
<div class="row">
  <div class="col-6">
    <p>Product A</p>
  </div>
  <div class="col-6">
    <p>More Product A</p>
  </div>
</div>
<div class="row">
  <div class="col-6">
    <p>Product B</p>
  </div>
  <div class="col-6">
    <p>More Product B</p>
 </div>
</div>
```

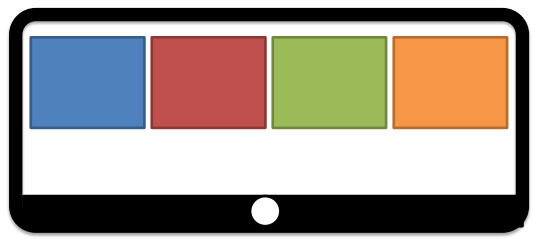
```
<div class="row">
 <div class="col-6">
   <div class="row">
     <div class="col-12">
       <p>Product A</p>
     </div>
   </div>
   <div class="row">
     <div class="col-12">
       <p>More Product A</p>
      </div>
   </div>
 </div>
 <div class="col-6">
   <div class="col-12">
     <div class="row">
       <p>Product B</p>
     </div>
   </div>
   <div class="row">
     <div class="col-12">
       <p>More Product B</p>
     </div>
   </div>
</div> </div>
```





# Bootstrap: Design for Multiple Devices

- md



- sm

SWINBURNE UNIVERSITY OF TECHNOLOGY

# WHAT'S NEXT?
## – VUEJS DATA BINDING, DIRECTIVES AND FILTERS

KNOWING

61