CSY1018
Web Development
Topic 2

Tom Butler
thomas.butler@northampton.ac.uk

Topic 2 - CSS

- What is CSS?
- How to attach a CSS file to a HTML document
- Targeting elements
- Example properties
- Useful resources

HTML

- A basic valid HTML file looks like this:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>My Web Page!</title>
    <meta charset="UTF-8" />
  </head>

  <body>
    <h1>Page heading</h1>
    <p>Page content</p>
  </body>
</html>
```

HTML

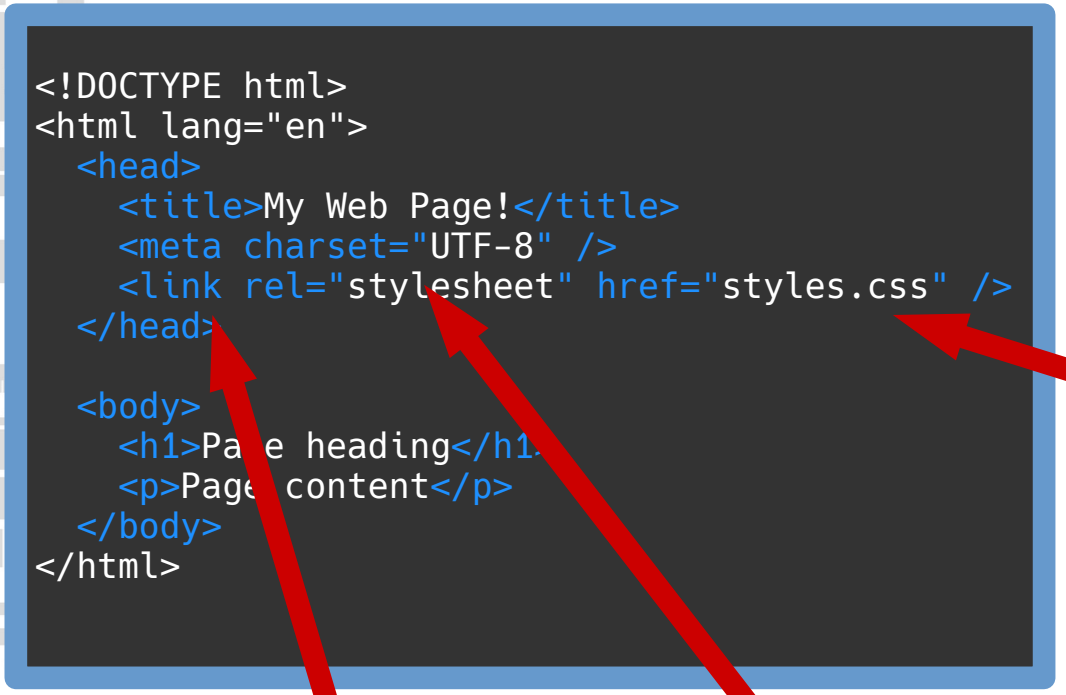
- When you put something in an HTML tag, the browser interprets that tag to mean a specific thing
  - Headings, paragraphs, lists, etc
- The browser makes some assumptions about how these should look
  - E.g. headings are larger and bold, text is black, the page background is white, etc

CSS

- CSS can be used to override this behaviour and describe how these elements should look
- CSS stands for *cascading style sheets*
- CSS is a list of *rules* which are applied to elements on the page
- In english a sample rule would be:
  - *All headings should have red text*

CSS

- CSS code goes in its own file
- CSS code looks very different from HTML. You can tell a CSS file apart from an HTML file just by looking at the structure of the code
- You have to specify the CSS file to use in the HTML document using the *link* tag in the <head> section of the page



Name of the CSS file
(should have a .css extension*)

The “rel” (relationship) tells
The browser what type of
File is being referenced

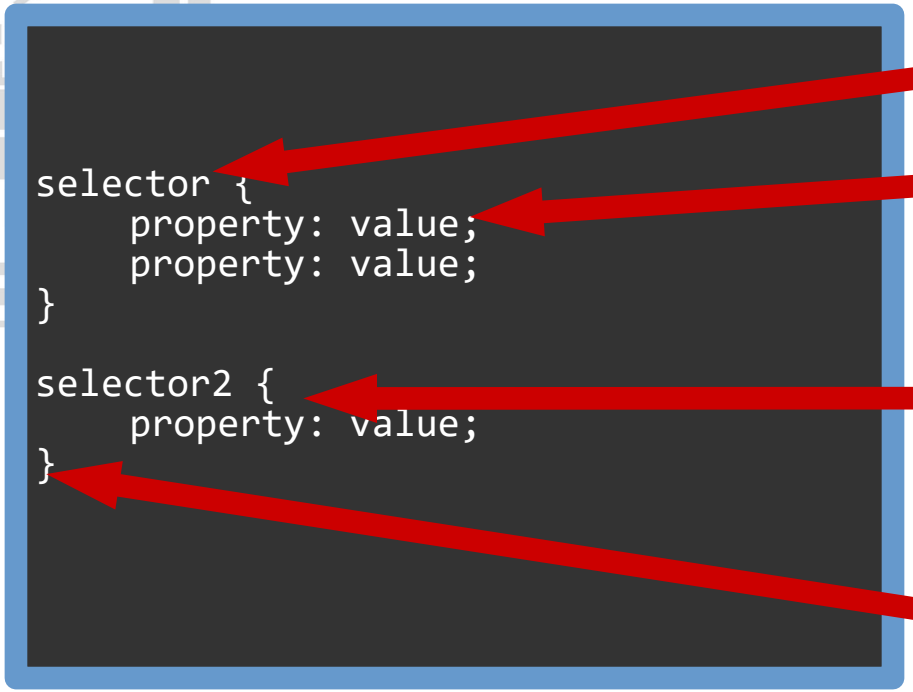
Link tag

CSS file

- A css file should be placed in the same folder as the HTML file and be given a .css extension
- Use your editor (Atom, VS Code, Sublime – see last week’s notes) to create the CSS file

CSS code

- CSS code uses this format:



Selector is used to find one or more HTML elements

One or more properties are applied to the selected elements

The CSS file can contain as many blocks as you like
Each block is a selector and properties

Braces are important:
Every block must have an opening and closing brace!

CSS real example

- A selector can be a tag name
- One property is “color” (Note the American spelling!)
- To set H1 elements to red and paragraphs to blue:

```
h1 {
    color: red;
}

p {
    color: blue;
}
```

CSS properties

- When you target a tag it will affect any tag of that type:

```
h1 {
    color: red;
}

p {
    color: blue;
}
```

Page heading

Paragraph 1

Paragraph 2

```
<body>
  <h1>Page heading</h1>
  <p>Paragraph 1</p>
  <p>Paragraph 2</p>
</body>
```

Class name selector

- Any element can be given a *class* this is an *HTML attribute* (see last week)
- The class name selector works in the format
  - .name
  - A dot followed by the name of the class you want to match
- This will match any element that has *class=”name”*
- **Note that the attribute does not include the dot in the HTML!**

Class name selector

- The class name selector is a name prefixed with a dot and selects any element with a class attribute set to that value

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <h1 class="myclass">Page heading</h1>
    <p class="myclass">Paragraph one</p>
    <p>Paragraph two</p>
  </body>
</html>
```

```
.myclass {
        color: green;
}
```

Notice that both elements with the class “myclass” are green

Class name selector

- The class name selector is useful whenever you want to be able to target specific elements on the page without affecting other elements by accident

ID Selector

- The ID selector works very similarly to the class name selector
  - Uses a # prefix
  - #myid
  - Matches any element with id=”myid” set as an attribute
- Note that the HTML does not contain the # symbol!
- IDs are different to class names. You can only use an ID once: Each ID should apply to a single element on the page

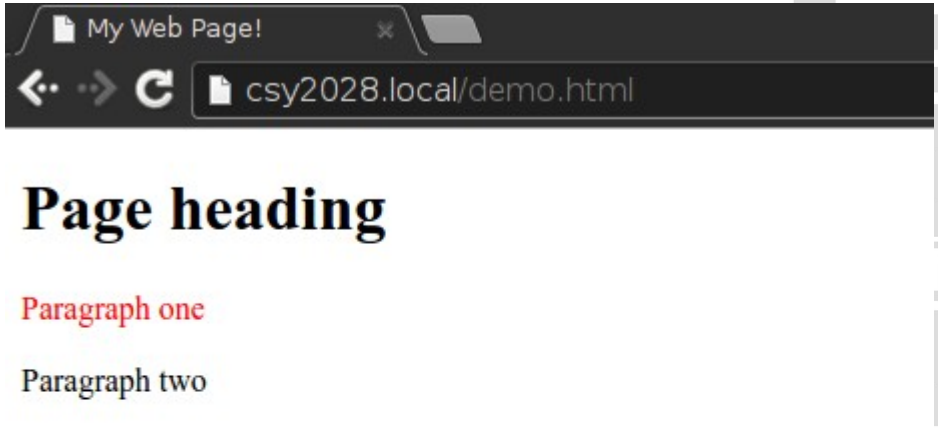
ID selector

- The ID name selector is a name prefixed with a hash and selects the element with that ID

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
    <h1>Page heading</h1>
    <p id="myid">Paragraph one</p>
    <p>Paragraph two</p>
  </body>
</html>
```

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

You may only have one element with each ID



Which should I use?

- In a lot of cases you can use a class name, or an element name or an ID to achieve exactly the same result!
- **You should avoid IDs** because you cannot easily reuse the style (if you want to make two elements red, you need to add two css rules and two IDs!)
- You should use element selectors when you want to affect all elements of that type
- If you want a specific style on a single part of the page, you should use class names
- **Don’t use IDs there is no benefit to doing so**

Combining Selectors

- HTML is a *nested* notation
  - (Some) Elements can exist inside (some) other elements
- You can use css to target nested elements. By combining a selector with a space you can target specific elements in the HTML DOM Tree
  - **article h1 {font-weight: bold}** – Sets any <h1> element inside an <article> element to bold
  - **.myclass span {font-style: italic}** – Sets any element inside an element with *class=”myclass”* to italic
- This will target elements any *depth!*

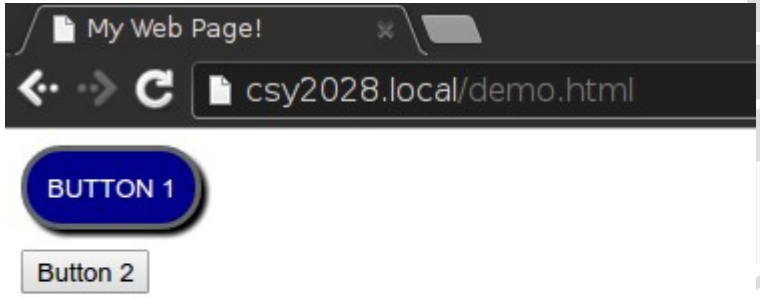
Combining selectors

- You can combine selectors to be more specific

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
    <div class="myform">
      <input type="button" value="Button 1" />
    </div>

    <input type="button" value="Button 2" />
  </body>
</html>
```

```
.myform input {
  border-radius: 20px;
  border: 3px solid #666;
  box-shadow: 3px 3px 3px #000;
  margin-bottom: 10px;
  background-color: darkblue;
  padding: 10px;
  text-transform: uppercase;
  color: white;
}
```



Notice that only the button inside the element with the class *myform* has been styled

Combining selectors

- This will target elements of any *depth*

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
    <div class="myform">
       <input type="button" value="Button 1" />
      <section>
        <form>
          <input type="button" value="Button 2" />
        </form>
      </section>
    </div>

    <input type="button" value="Button 3" />
  </body>
</html>
```

```
.myform input {
  border-radius: 20px;
  border: 3px solid #666;
  box-shadow: 3px 3px 3px #000;
  margin-bottom: 10px;
  background-color: darkblue;
  padding: 10px;
  text-transform: uppercase;
  color: white;
}
```

Notice that only the button inside the element with the class *myform*

Combining selectors

- You can use the *direct decedent* selector to filter to only elements that are only one level down
- The direct decedent selector uses the greater than symbol >

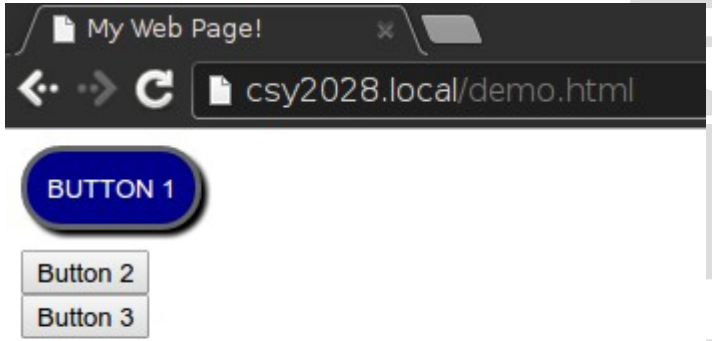
Combining selectors

- This will target elements only one level down

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
    <div class="myform">
       <input type="button" value="Button 1" />
      <section>
        <form>
          <input type="button" value="Button 2" />
        </form>
      </section>
    </div>

    <input type="button" value="Button 3" />
  </body>
</html>
```

```
.myform > input {
  border-radius: 20px;
  border: 3px solid #666;
  box-shadow: 3px 3px 3px #000;
  margin-bottom: 10px;
  background-color: darkblue;
  padding: 10px;
  text-transform: uppercase;
  color: white;
}
```



Notice that only the button inside the element with the class *myform*

Combining selectors

- You can combine selectors into very complex expressions e.g.:
- **`#myelement > .myclass article section header h1`**
- However, this is generally discouraged
- We call this *tightly coupled* to the HTML: A slight change to the HTML will stop the h1 tag being styled
- As a rule of thumb you shouldn't need more than three levels of selector

Block and inline elements

- Every element in HTML defaults to one of two *display* properties:
  - inline
  - block

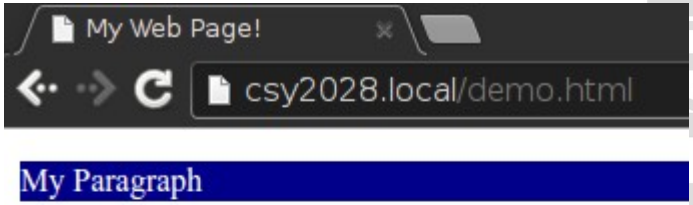
  These are treated very differently by the browser and determine how elements interact with one another.
- Block level elements extend all the way across the page and nothing can appear on the same line as them
- Inline elements are only as long as their content (e.g. the text inside them) and can appear on the same line

- By default:
  - Paragraphs, headings, lists, etc are *block* elements
  - Buttons, images, textboxes and elements are *inline*
- But this is only a default CSS can be used to override the display property in CSS can be set to inline or block to override the default behaviour

Block elements

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>My Paragraph</p>
  </body>
</html>
```

```
p {
  background-color: darkblue;
  color: white;
  display: block;
}
```

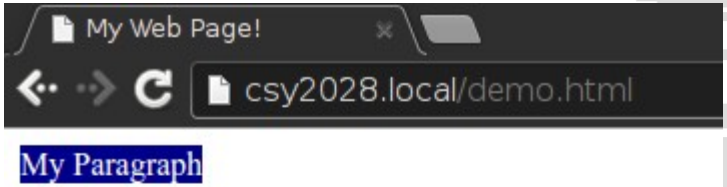


Notice that the blue background goes all the way across the page

Inline elements

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>My Paragraph</p>
  </body>
</html>
```

```
p {
  background-color: darkblue;
  color: white;
  display: inline;
}
```



Notice that the blue background only extends to the width of the text

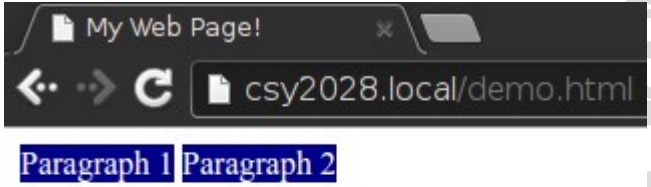
Block vs inline

- Block and inline affects how elements interact with each other
- Block elements will always be on their own line
- Inline elements will appear on the same line if there is enough space

Inline elements

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
  </body>
</html>
```

```
p {
  background-color: darkblue;
  color: white;
  display: inline;
}
```

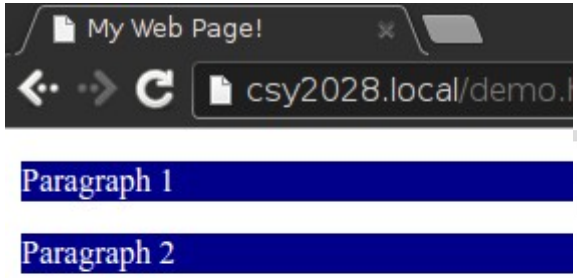


The elements appear on the same line

Block elements

```html
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
  </body>
</html>
```

```css
p {
  background-color: darkblue;
  color: white;
  display: block;
}
```



The elements appear on the same line

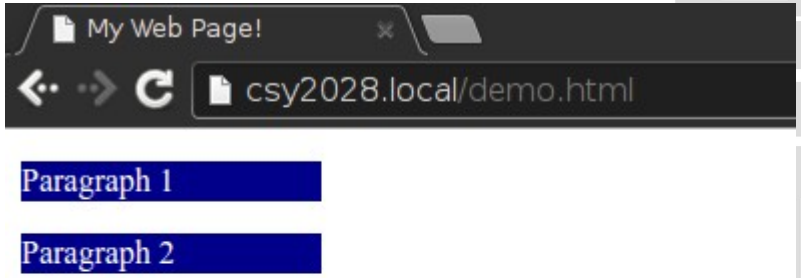
Block vs inline

- Block elements can have a width and height
- Setting a width and a height on inline elements will have no effect

Block elements

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
  </body>
</html>
```

```
p {
  background-color: darkblue;
  color: white;
  display: block;
  width: 150px;
}
```

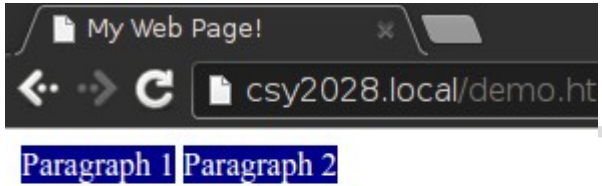


The elements appear on their own lines but now don't span the entire width

Inline elements

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
  </body>
</html>
```

```
p {
  background-color: darkblue;
  color: white;
  display: inline;
  width: 150px;
}
```



Width has no effect, they both stretch only as wide as the content

Float

- How do you put elements with a fixed with/height on the same line?
- *Float* can be applied to *block* elements and have them appear on the same line but with fixed dimensions
- Float has three options:
  - None (default, no float)
  - Left
  - Right

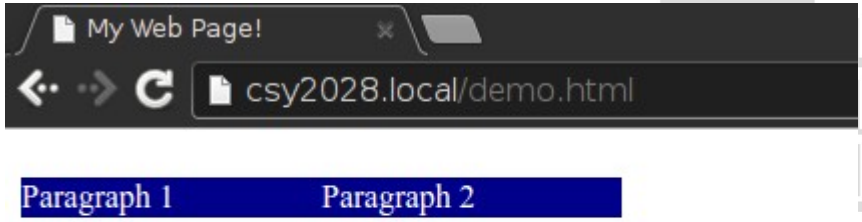
Float: left

- Float left will have the affect of stacking block elements left to right. When elements get too wide for the browser they will wrap to the next line

Float: left

```html
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
  </body>
</html>
```

```css
p {
  background-color: darkblue;
  color: white;
  display: block;
  width: 150px;
  float: left;
}
```



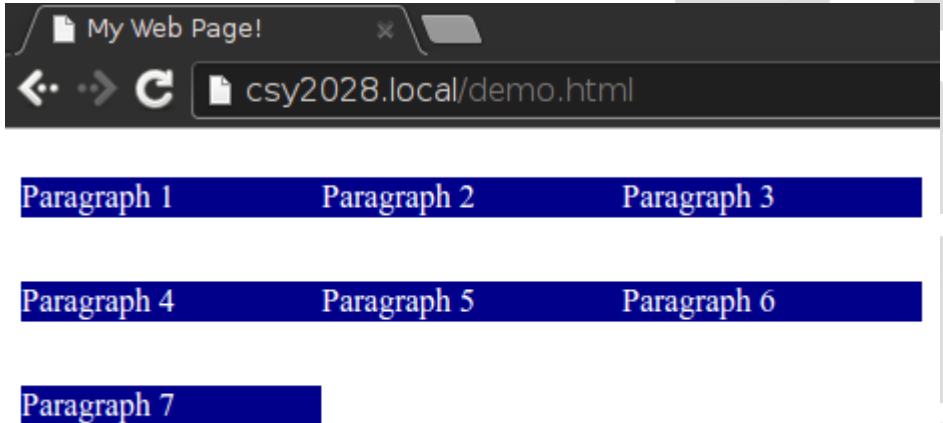
Float: left

- As more elements are added, they will be stacked left to right until there is no more room on the line

Float: left

```html
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
     <p>Paragraph 3</p>
     <p>Paragraph 4</p>
     <p>Paragraph 5</p>
     <p>Paragraph 6</p>
     <p>Paragraph 7</p>
  </body>
</html>
```

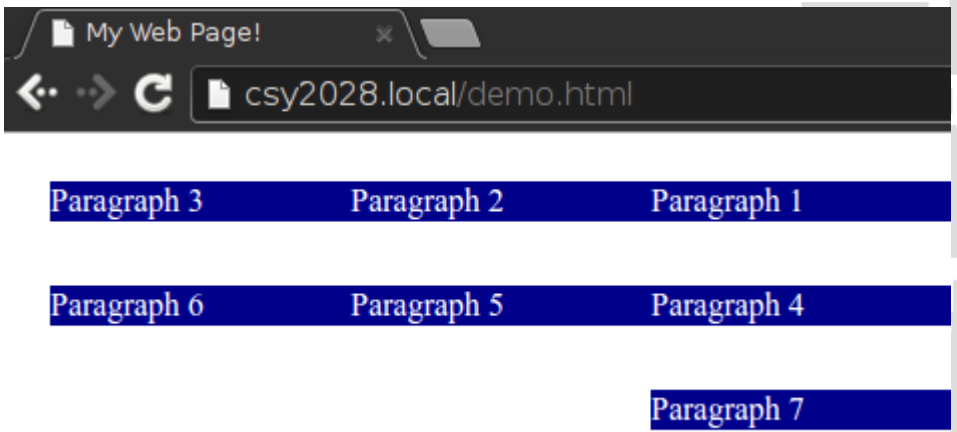
```css
p {
  background-color: darkblue;
  color: white;
  display: block;
  width: 150px;
  float: left;
}
```



Float: right

```html
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
     <p>Paragraph 3</p>
     <p>Paragraph 4</p>
     <p>Paragraph 5</p>
     <p>Paragraph 6</p>
     <p>Paragraph 7</p>
  </body>
</html>
```

```css
p {
  background-color: darkblue;
  color: white;
  display: block;
  width: 150px;
  float: right;
}
```

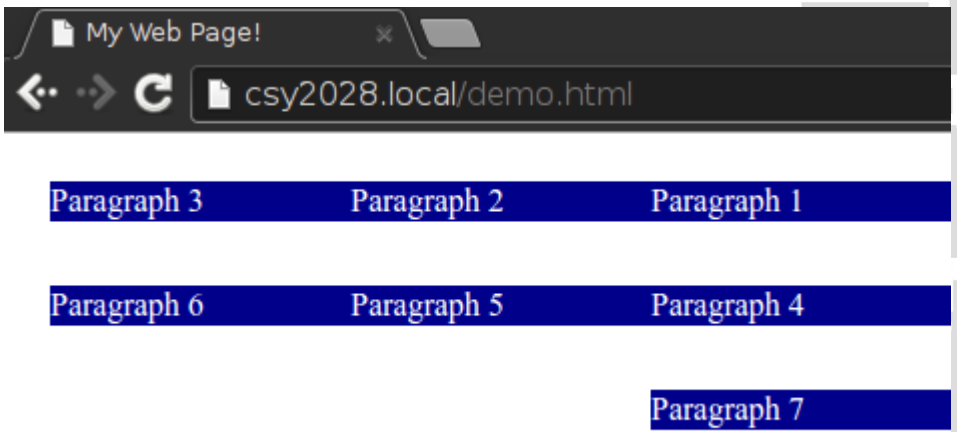


Float: right stacks the elements from right to left

Float: right

```html
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
     <p>Paragraph 3</p>
     <p>Paragraph 4</p>
     <p>Paragraph 5</p>
     <p>Paragraph 6</p>
     <p>Paragraph 7</p>
  </body>
</html>
```

```css
p {
  background-color: darkblue;
  color: white;
  display: block;
  width: 150px;
  float: right;
}
```



Float: right stacks the elements from right to left

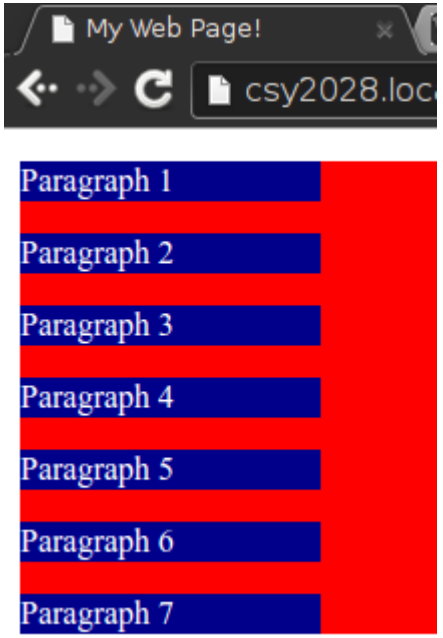
Float

```html
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
    <div class="red">
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
     <p>Paragraph 3</p>
     <p>Paragraph 4</p>
     <p>Paragraph 5</p>
     <p>Paragraph 6</p>
     <p>Paragraph 7</p>
    </div>
  </body>
</html>
```

```css
p {
  background-color: darkblue;
  color: white;
  display: block;
  width: 150px;

}

.red {
  background-color: red;
}
```



As you expect, the paragraphs sit in a container with a red background

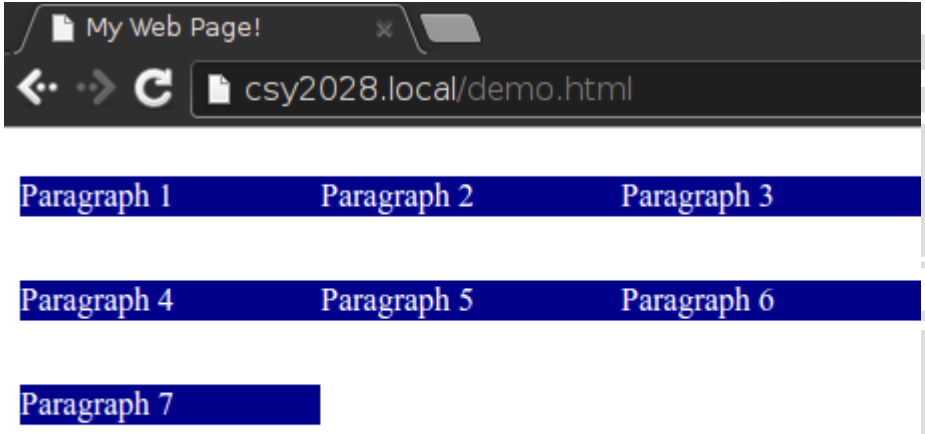
Float

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
    <div class="red">
     <p>Paragraph 1</p>
     <p>Paragraph 2</p>
     <p>Paragraph 3</p>
     <p>Paragraph 4</p>
     <p>Paragraph 5</p>
     <p>Paragraph 6</p>
     <p>Paragraph 7</p>
    </div>
  </body>
</html>
```

```
p {
  background-color: darkblue;
  color: white;
  display: block;
  width: 150px;
  float: left;

}

.red {
  background-color: red;
}
```



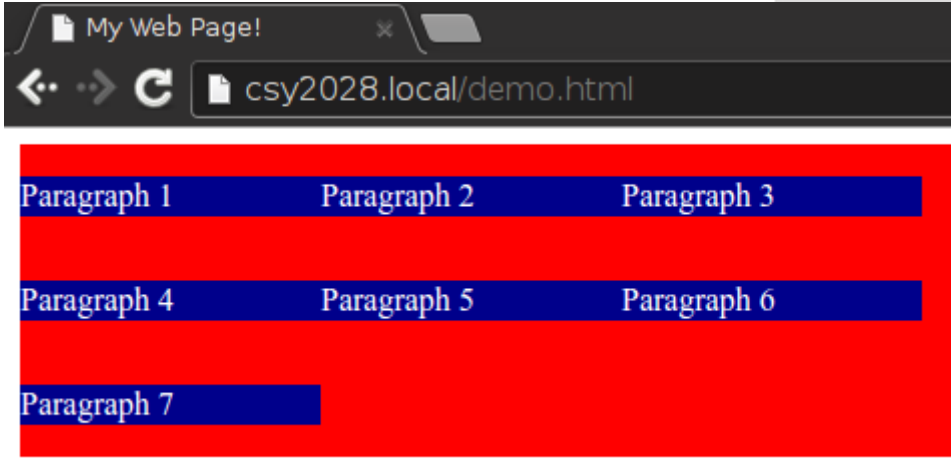
But when float: left is added,
The background disappears!

Float

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Web Page!</title>
    <link rel="stylesheet" href="demo.css" />
  </head>
  <body>
    <div class="red">
      <p>Paragraph 1</p>
      <p>Paragraph 2</p>
      <p>Paragraph 3</p>
      <p>Paragraph 4</p>
      <p>Paragraph 5</p>
      <p>Paragraph 6</p>
      <p>Paragraph 7</p>
    </div>
  </body>
</html>
```

```
p {
  background-color: darkblue;
  color: white;
  display: block;
  width: 150px;
  float: left;
}

.red {
  background-color: red;
  overflow: auto;
}
```



By adding overflow: auto,
The background appears

Exercises

- 1. Download exercise1.html
  - Create the css file and create rules that make the various elements
  - **Do not change the HTML code!**
  - **Hint: Be sure to look at the structure of the HTML!**
  - Necessary CSS properties:
    - Color (sets the colour of the font)
    - Background-color (sets an element’s background colour)

Exercise 2

1. Download Exercise2.html

   - Create the css file

   - Make it so each module appears in its own column with its own background colour

Hint: You can use 50% widths on the Elements!

CSY1018 - Web Development

Lecturer: Tom Butler

This purpose of this (Level 4) module is to give students an understanding of client side web technologies. This module provides students with: the essential knowledge and practical skills to design develop and implement a Web site to contenporary web standard; an overview of the Internet technologies the overall software architecture of the Internet including servers clients browsers is covered leading to the use of a Web server to install maintain and publish Web Pages to achieve Web presence; standard client side dynamic web development environment such as HTML Cascading Style Sheets (CSS) and JavaScript are covered in detail.

CSY1019 - Software Engineering 1

Lecturer: Mark Johnson

The module will introduce students to the Software Engineering lifecycle. Focusing on: investigating a problem domain eliciting software requirements preparing a requirement specification document performing system design and presenting them to clients; introduce students to the skills principles and concepts necessary to implement solutions; use of a high level programming language to implement algorithms; a late-objects approach will be adopted to teach programming.

Exercise 3

- Extend exercise 2 to include information about 4 modules in a 4x4 grid
- Extend exercise 2 to include information about 6 modules in a 3x2 grid
- Module information can be found here: https://www.northampton.ac.uk/awards/modules/COMPUTER-SYSTEMS/4/

Exercise 4

- Use your pages from Topic 1 that include module information
- Style them so that the navigation appears vertically across the page
- Use font-family: arial; to give the pages a nicer font
  - (or choose a different font if you prefer)
- Select background colours and font colours to make the page look nicer
- Select a colour for your links
- Play around with the margins and paddings to make the page look nice
- **Hint: You can link to the same css file from each of your**
- Hint: You can hide the bullet points from a list using the property:
  - list-style:none

Exercise 5 (Difficult!)

- Try creating a page with 3 columns and a header and footer
  - Note: This is difficult, there are multiple solutions and we will cover this in detail next week