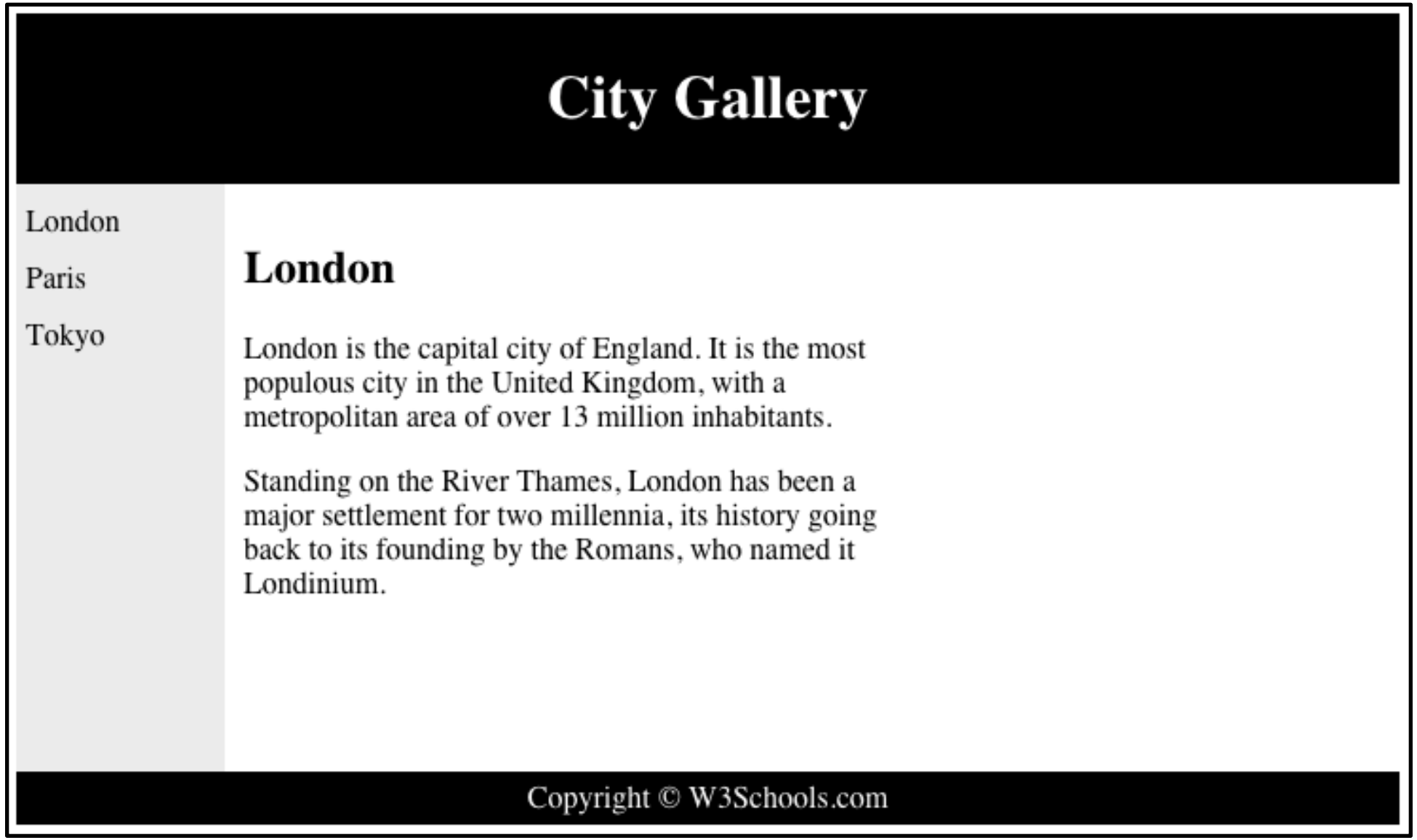
COMP30680
Web Application Development

Layouts with HTML and CSS.

HTML5

David Coyle
d.coyle@ucd.ie

A simple layout example



See basic_layout.html

CSS id type selectors are created and used with div elements in a HTML page

```
<style>
#header {
    background-color:black;
    color:white;
    text-align:center;
    padding:5px;
}
#nav {
    line-height:30px;
    background-color:#eeeeee;
    height:300px;
    width:100px;
    float:left;
    padding:5px;
}
#section {
    width:350px;
    float:left;
    padding:10px;
}
#footer {
    background-color:black;
    color:white;
    clear:both;
    text-align:center;
   padding:5px;
}
</style>
```

```
<body>

<div id="header">
<h1>City Gallery</h1>
</div>

<div id="nav">
London<br>
Paris<br>
Tokyo
</div>

<div id="section">
<h2>London</h2>
<p>London is the capital city of England. It is the most populous city in the Un
with a metropolitan area of over 13 million inhabitants.</p>
<p>Standing on the River Thames, London has been a major settlement for two mill
its history going back to its founding by the Romans, who named it Londinium.</p
</div>

<div id="footer">
Copyright © W3Schools.com
</div>

</body>
```

See basic_layout.html

CSS id type selectors are created and used with div elements in a HTML page

```
<style>
#header {
    background-color:black;
    color:white;
    text-align:center;
    padding:5px;
}
#nav {
    line-height:30px;
    background-color:#eeeeee;
    height:300px;
    width:100px;
    float:left;
    padding:5px;
}
#section {
    width:350px;
    float:left;
    padding:10px;
}
#footer {
    background-color:black;
    color:white;
    clear:both;
    text-align:center;
   padding:5px;
}
</style>
```

```
<body>

<div id="header">
<h1>City Gallery</h1>
</div>

<div id="nav">
London<br>
Paris<br>
Tokyo
</div>

<div id="section">
<h2>London</h2>
<p>London is the capital city of England. It is the most populous city in the Ur
with a metropolitan area of over 13 million inhabitants.</p>
<p>Standing on the River Thames, London has been a major settlement for two mill
its history going back to its founding by the Romans, who named it Londinium.</p
</div>

<div id="footer">
Copyright © W3Schools.com
</div>

</body>
```

See basic_layout.html

```css
<style>
#header {
    background-color:black;
    color:white;
    text-align:center;
    padding:5px;
}
#nav {
    line-height:30px;
    background-color:#eeeeee;
    height:300px;
    width:100px;
    float:left;
    padding:5px;
}
#section {
    width:350px;
    float:left;
    padding:10px;
}
#footer {
    background-color:black;
    color:white;
    clear:both;
    text-align:center;
    padding:5px;
}
</style>
```

The properties of the CSS rules are then used to style different elements.

```html
<body>

<div id="header">
<h1>City Gallery</h1>
</div>

<div id="nav">
London<br>
Paris<br>
Tokyo
</div>

<div id="section">
<h2>London</h2>
<p>London is the capital city of England. It is the most populous city in the Ur
with a metropolitan area of over 13 million inhabitants.</p>
<p>Standing on the River Thames, London has been a major settlement for two mill
its history going back to its founding by the Romans, who named it Londinium.</p
</div>

<div id="footer">
Copyright © W3Schools.com
</div>

</body>
```

See basic_layout.html

Key positioning properties

- Display: http://www.w3schools.com/css/css_display_visibility.asp
- Position: http://www.w3schools.com/css/css_positioning.asp
- Float: http://www.w3schools.com/css/css_float.asp

Also useful:

- Inline block: http://www.w3schools.com/css/css_inline-block.asp
- CSS alignment techniques: http://www.w3schools.com/css/css_align.asp
- Height and width: http://www.w3schools.com/css/css_dimension.asp
- Max-width: http://www.w3schools.com/css/css_max-width.asp

CSS Display property

The display property is the most important CSS property for controlling layout.
It specifies if/how an element is displayed.

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is block or inline.

- Examples of block-level elements: , <h1> - <h6>, <p>, <form>, <header>, <footer>, <section>,

- Examples of inline-level elements: , <a>, <img>

Display : none

- Used with dropdown menus
- Also commonly used with JavaScript to hide and show elements without deleting and recreating them. See js.display.html

Note: you can override the default, e.g. making inline <li> elements for horizontal menus.

CSS Position

The **position** property specifies the type of positioning method used for an element.

There are four different position values:

- static
- relative
- fixed
- absolute

Elements are then positioned using the **top, bottom, left, and right** properties.

Note:

- Top, bottom, left and right will not work unless the position property is set first.

- They also work differently depending on the position value.

Position - static

HTML elements are positioned ***static by default***.

Static positioned elements are not affected by the top, bottom, left, and right properties.
The element is not positioned in any special way; it is always positioned according to the normal flow of the page.

Position - relative

An element with position: relative; is positioned ***relative to its normal position***.

Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position.

Other content will not be adjusted to fit into any gap left by the element.

See **static_relative.html.**

Position - fixed

An element with position: fixed; is positioned ***relative to the viewport***, which means it always stays in the same place even if the page is scrolled.

The top, right, bottom, and left properties are used to position the element.

A fixed element does not leave a gap in the page where it would normally have been located.

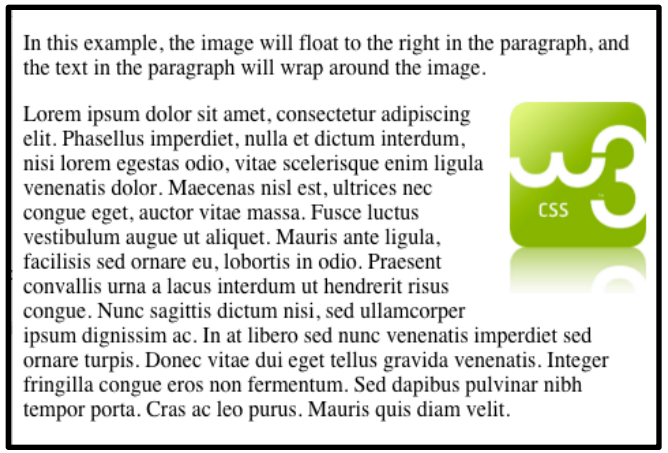
Position - absolute

An element with position: absolute; is positioned ***relative to the nearest positioned ancestor*** (instead of positioned relative to the viewport, like fixed).

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

See **fixed_absolute.html.**

CSS Float


In this example, the image will float to the right in the paragraph, and the text in the paragraph will wrap around the image.

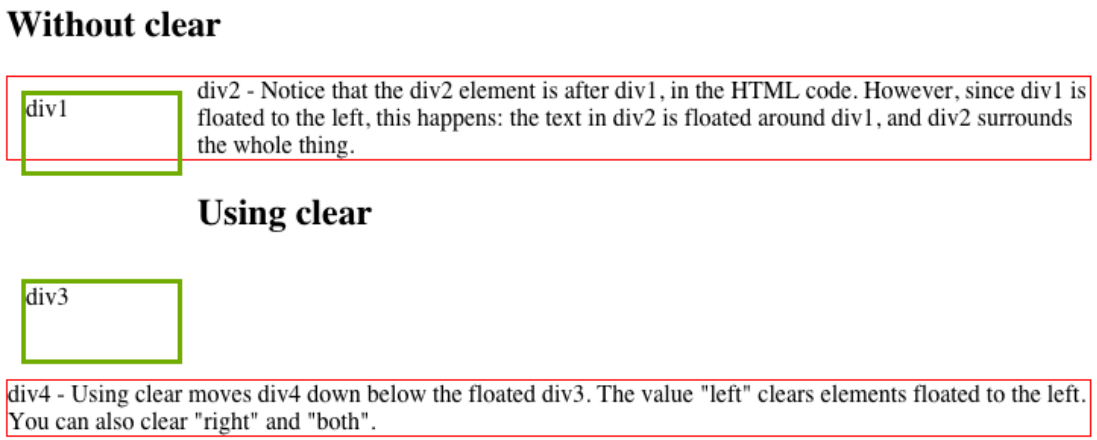
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac. In at libero sed nunc venenatis imperdiet sed ornare turpis. Donec vitae dui eget tellus gravida venenatis. Integer fringilla congue eros non fermentum. Sed dapibus pulvinar nibh tempor porta. Cras ac leo purus. Mauris quis diam velit.

The **float** property specifies whether or not an element should float.

See **image_float.html**

Elements after a floating element will ‘float’ around it. This is very useful for layouts, but you need to be careful about overlapping.

One good option is to use the **clear** property is used to control the behavior of floating elements. It can be used to specify the sides of an element floating ***on which other elements are not allowed to float***.



See **float_clear.html.**

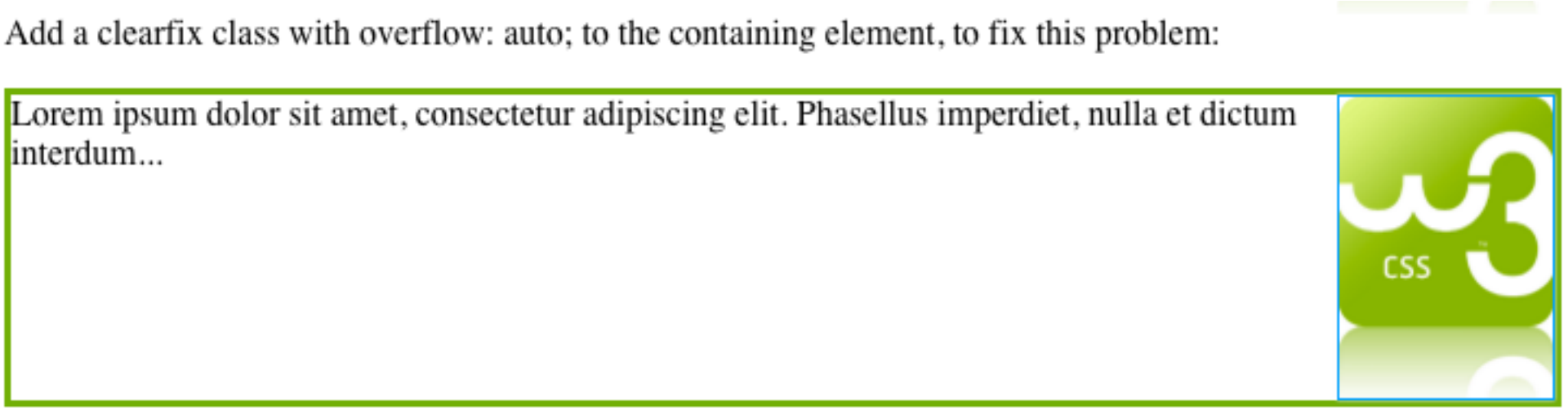
CSS Float - overflow: auto

The **overflow: auto** property is also useful.

E.g. if an element is taller than the element containing it, and it is floated, it will overflow outside of its container.

Then we can add overflow: auto; to the containing element to fix this problem

See **overflow_auto.html.**



CSS Inline Block

It has been possible for a long time to create a grid of boxes that fills the browser width and wraps nicely (when the browser is resized), by using the float property.

The **inline-block value** of the **display property** makes this even easier.

Inline-block elements are like inline elements but they can have a width and a height.

Compare the two files:

- **inline_box_old.html**
- **Inline_box.new.html**

Alignment

In CSS, several properties can be used to align elements.

Centre Align – using margin

Setting the **width** of a block-level element will prevent it from stretching out to the edges of its container. Combine this with **margin: auto;**, to horizontally center an element within its container.

The element will then take up the specified width, and the remaining space will be split equally between the two margins.

See **centre_margin_align.html**

Alignment

Left and right align – using position

Note: Absolute positioned elements are removed from the normal flow, and can overlap elements.

See **left_right_position_align.html**

Left and right align – using float

See **left_right_position_float.html**

For more examples see: http://www.w3schools.com/css/css_align.asp

WS3 schools tip: When aligning elements with float, always define margin and padding for the <body> element. This is to avoid visual differences in different browsers.

Back to our original example: basic_layout.html

CSS id type selectors are created and used with div elements in a HTML page

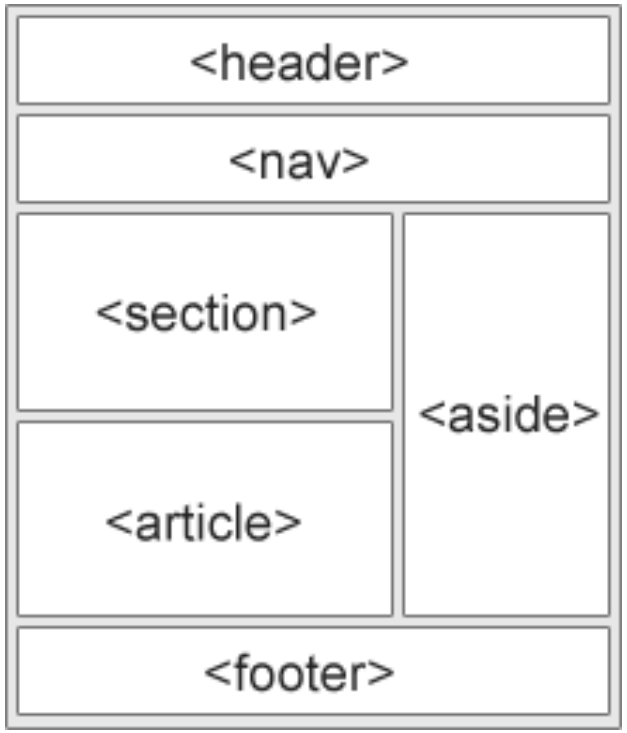
```
<style>
#header {
    background-color:black;
    color:white;
    text-align:center;
    padding:5px;
}
#nav {
    line-height:30px;
    background-color:#eeeeee;
    height:300px;
    width:100px;
    float:left;
    padding:5px;
}
#section {
    width:350px;
    float:left;
    padding:10px;
}
#footer {
    background-color:black;
    color:white;
    clear:both;
    text-align:center;
   padding:5px;
}
```

```
<body>

<div id="header">
<h1>City Gallery</h1>
</div>

<div id="nav">
London<br>
Paris<br>
Tokyo
</div>

<div id="section">
<h2>London</h2>
<p>London is the capital city of England. It is the most populous city in the Ur
with a metropolitan area of over 13 million inhabitants.</p>
<p>Standing on the River Thames, London has been a major settlement for two mill
its history going back to its founding by the Romans, who named it Londinium.</p
</div>

<div id="footer">
Copyright © W3Schools.com
</div>

</body>
```

See basic_layout.html

Website Layout Using HTML5

HTML5 offers new semantic elements that define different parts of a web page:



<header>	- Defines a header for a document or a section
<nav>	- Defines a container for navigation links
<section>	- Defines a section in a document
<article>	- Defines an independent self-contained article
<aside>	- Defines content aside from the content (like a sidebar)
<footer>	- Defines a footer for a document or a section

See **html5_layout.html**, which creates the sale layout as **basic_layout.html**, but uses semantic elements.

CSS is used to create styles for HTML5 semantic tags.

```
<style>
header {
    background-color:black;
    color:white;
    text-align:center;
    padding:5px;
}
nav {
    line-height:30px;
    background-color:#eeeeee;
    height:300px;
    width:100px;
    float:left;
    padding:5px;
}
section {
    width:350px;
    float:left;
    padding:10px;
}
footer {
    background-color:black;
    color:white;
    clear:both;
    text-align:center;
    padding:5px;
}
</style>
```

```
<body>

<header>
<h1>City Gallery</h1>
</header>

<nav>
London<br>
Paris<br>
Tokyo
</nav>

<section>
<h1>London</h1>
<p>London is the capital city of England. It is the most populou
with a metropolitan area of over 13 million inhabitants.</p>
<p>Standing on the River Thames, London has been a major settler
its history going back to its founding by the Romans, who named
</section>

<footer>
Copyright © W3Schools.com
</footer>

</body>
```

See **html5_layout.html**

CSS is used to create styles for HTML5 semantic tags.

```css
<style>
header {
    background-color:black;
    color:white;
    text-align:center;
    padding:5px;
}
nav {
    line-height:30px;
    background-color:#eeeeee;
    height:300px;
    width:100px;
    float:left;
    padding:5px;
}
section {
    width:350px;
    float:left;
    padding:10px;
}
footer {
    background-color:black;
    color:white;
    clear:both;
    text-align:center;
    padding:5px;
}
</style>
```

```html
<body>

<header>
<h1>City Gallery</h1>
</header>

<nav>
London<br>
Paris<br>
Tokyo
</nav>

<section>
<h1>London</h1>
<p>London is the capital city of England. It is the most populou
with a metropolitan area of over 13 million inhabitants.</p>
<p>Standing on the River Thames, London has been a major settler
its history going back to its founding by the Romans, who named
</section>

<footer>
Copyright © W3Schools.com
</footer>

</body>
```

See **html5_layout.html**

HTML5 semantic elements

Tag	Description
<article>	Defines an article
<aside>	Defines content aside from the page content
<details>	Defines additional details that the user can view or hide
<figcaption>	Defines a caption for a <figure> element
<figure>	Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.
<footer>	Defines a footer for a document or section
<header>	Specifies a header for a document or section
<main>	Specifies the main content of a document
<mark>	Defines marked/highlighted text
<nav>	Defines navigation links
<section>	Defines a section in a document
<summary>	Defines a visible heading for a <details> element
<time>	Defines a date/time

HTML5 semantic elements

Tag	Description
<article>	Defines an article
<aside>	Defines content aside from the
<details>	Defines additional details that t
<figcaption>	Defines a caption for a <figure
<figure>	Specifies self-contained conten
<footer>	Defines a footer for a documen
<header>	Specifies a header for a docum
<main>	Specifies the main content of a
<mark>	Defines marked/highlighted tex
<nav>	Defines navigation links
<section>	Defines a section in a documen
<summary>	Defines a visible heading for a <details> element
<time>	Defines a date/time

Example:

The <article> element specifies independent, self-contained content.

An article should make sense on its own, and it should be possible to read it independently from the rest of the web site.

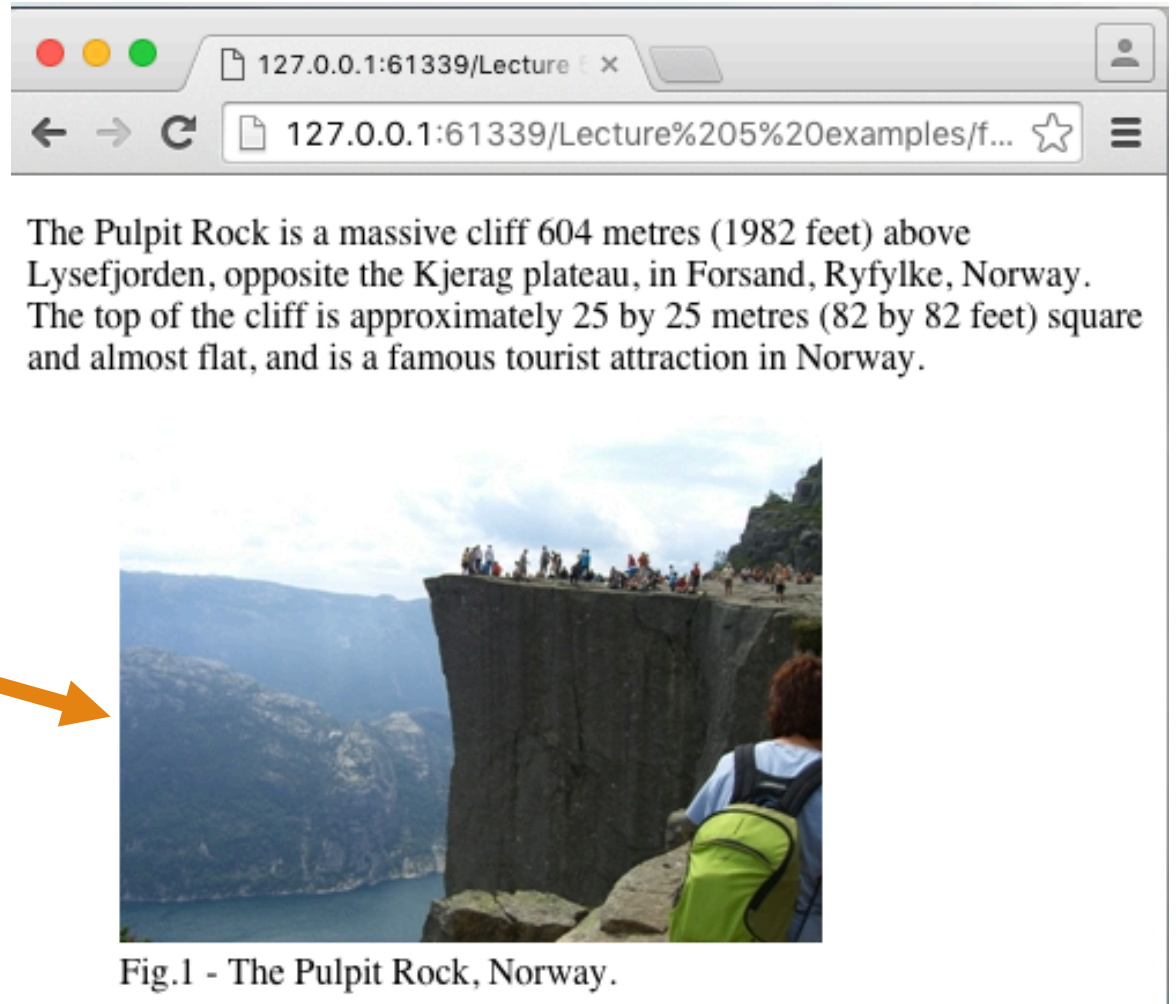
Examples of where an <article> element can be used:

- Forum post
- Blog post
- Newspaper article

For additional descriptions see:
http://www.w3schools.com/html/html5_semantic_elements.asp

HTML5 semantic elements

```
1   <!DOCTYPE html>
2   <html>
3   <body>
4
5   <p>The Pulpit Rock is a massive cliff 604 metres
    (1982 feet) above Lysefjorden, opposite the Kjerag
    plateau, in Forsand, Ryfylke, Norway. The top of
    the cliff is approximately 25 by 25 metres (82 by
    82 feet) square and almost flat, and is a famous
    tourist attraction in Norway.</p>
6
7   <figure>
8     <img src="img_pulpit.jpg" alt="The Pulpit Rock"
      width="304" height="228">
9     <figcaption>Fig.1 - The Pulpit Rock, Norway.
      </figcaption>
10  </figure>
11
12  </body>
13  </html>
14
```



Provides some nice layout short cuts, e.g. adding a caption to a figure.

Migrating to HTML5

Typical HTML4	Typical HTML5
	<header>
	<nav>
	<section>
	<article>
	<footer>

For a more complete tutorial and further examples of page layout using HTML5 see:
http://www.w3schools.com/html/html5_migration.asp

Why used Semantic Elements?

With HTML4, developers used their own favorite attribute names to style page elements:
header, top, bottom, footer, menu, navigation, main, container, content, article, sidebar, topnav, ...

This made it impossible for search engines to identify the correct web page content.

With HTML5 elements like: <header> <footer> <nav> <section> <article>, this will become easier.

According to the W3C, a Semantic Web:

"Allows data to be shared and reused across applications, enterprises, and communities."

Questions, Suggestions?

Next class:

Advanced HTML elements