PAGE LAYOUT WITH CSS
Overview

- Styling Page Sections
- Introduction to Layout
- Floating Elements
- Sizing & Positioning
Why do we need page sections?

- Style individual elements, groups of elements, sections of text or of the page
- Create complex page layouts
Sections of a page <div>

```html
<div class="shout">
<h2>Coding Horror! Coding Horror!</h2>
<p class="special">See our special deal on Droids!</p>
<p>We'll beat any advertised price!</p>
</div>
```

HTML

Coding Horror! Coding Horror!

See our special deal on Droids!

We’ll beat any advertised price!

- Tag used to indicate a logical section or area of a page
- Has no appearance by default, but you can apply styles to it
Coding Horror! Coding Horror!

See our <span class="special">spectacular</span> deal on Droids!

We’ll beat <span class="shout">any advertised price</span>!

- has no onscreen appearance, but you can apply a style or ID to it, which will be applied to the text inside the span
CSS context selectors

```
selector1 selector2 { 
properties
}
```

- Applies the given properties to `selector2` only if it is inside a `selector1` on the page.

```
selector1 > selector2 { 
properties
}
```

- Applies the given properties to `selector2` only if it is directly inside a `selector1` on the page.
Eat at Greasy’s Burger…

- The **greasiest** burgers in town!
- Yummy and greasy at the same time!
Eat at Greasy’s Burger…

- The **greasiest** burgers in town!
- Yummy and **greasy at the same time**!
The CSS Box Model

- Every element composed of:
  - content
  - a border around the element
  - padding between the content and the border
  - a margin between the border and other content
The CSS Box Model (cont.)

width = content width + L/R padding + L/R border + L/R margin

height = content height + T/B padding + T/B border + T/B margin
Document Flow – block elements
Document flow - inline elements
Document flow - a larger example
CSS properties for borders

This is a heading.

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>border</td>
<td>thickness/style/size of border on all 4 sides</td>
</tr>
</tbody>
</table>

- **Thickness**: px, pt, em, or thin, medium, thick
- **Style**: none, hidden, dotted, dashed, double, groove, inset, outset, ridge, solid
- **color**
## More border properties

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>border-color, border-width, border-style</td>
<td>specific properties of border on all 4 sides</td>
</tr>
<tr>
<td>border-bottom, border-left, border-right, border-top</td>
<td>all properties of border on a particular side</td>
</tr>
<tr>
<td>border-bottom-color, border-bottom-style, border-bottom-width, border-left-color, border-left-style, border-left-width, border-right-color, border-right-style, border-right-width, border-top-color, border-top-style, border-top-width</td>
<td>properties of border on a particular side</td>
</tr>
</tbody>
</table>

[Complete list of border properties](http://www.w3schools.com/css/css_border.asp)
Another border example

```css
h2 {
  border-left: thick dotted #CC0088;
  border-bottom-color: rgb(0, 128, 128);
  border-bottom-style: double;
}

CSS
```

- This is a heading.
  - each side’s border properties can be set individually
  - if you omit some properties, they receive default
CSS properties for padding

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>padding</td>
<td>padding on all 4 sides</td>
</tr>
<tr>
<td>padding-bottom</td>
<td>padding on bottom side only</td>
</tr>
<tr>
<td>padding-left</td>
<td>padding on left side only</td>
</tr>
<tr>
<td>padding-right</td>
<td>padding on right side only</td>
</tr>
<tr>
<td>padding-top</td>
<td>padding on top side only</td>
</tr>
</tbody>
</table>

[Complete list of padding properties](http://www.w3schools.com/css/css_padding.asp)
Padding example 1

```css
p { padding: 20px; border: 3px solid black; }

h2 { padding: 0px; background-color: yellow; }
```

This is a first paragraph.

This is a second paragraph.

This is a heading
Padding example 2

```css
p {
  padding-left: 200px; padding-top: 30px;
  background-color: fuchsia;
}
```

This is a first paragraph

This is a second paragraph

- each side's padding can be set individually
- notice that padding shares the background color of the element
## CSS properties for margins

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin</td>
<td>margin on all 4 sides</td>
</tr>
<tr>
<td>margin-bottom</td>
<td>margin on bottom side only</td>
</tr>
<tr>
<td>margin-left</td>
<td>margin on left side only</td>
</tr>
<tr>
<td>margin-right</td>
<td>margin on right side only</td>
</tr>
<tr>
<td>margin-top</td>
<td>margin on top side only</td>
</tr>
</tbody>
</table>

*Complete list of margin properties*

[http://www.w3schools.com/css/css_margin.asp](http://www.w3schools.com/css/css_margin.asp)
Margin example 1

```css
p {
  margin: 50px;
  background-color: fuchsia;
}
```

- notice that margins are always transparent
Margin example 2

```
p {  
margin-left: 8em;
background-color: fuchsia;
}
```

- each side's margin can be set individually

This is a first paragraph

This is a second paragraph
CSS properties for dimensions

p { width: 350px; background-color: yellow; }
h2 { width: 50%; background-color: aqua; }

This paragraph uses the first style above

An h2 heading

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>width, height</td>
<td>how wide or tall to make this element (block elements only)</td>
</tr>
<tr>
<td>max-width, max-height,</td>
<td>max/min size of this element in given dimension</td>
</tr>
<tr>
<td>min-width, min-height</td>
<td></td>
</tr>
</tbody>
</table>
Floating Elements
The **CSS** `float` property

```css
img.headericon {
  float: right; width: 130px;
}
```

Ghostbusters is a 1984 American science fiction comedy film written by co-stars Dan Aykroyd and Harold Ramis about three eccentric New York City parapsychologists-turned-ghost capturers.

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>float</td>
<td>side to hover on; can be left, right, or none (default)</td>
</tr>
</tbody>
</table>

removed from normal document flow; underlying text wraps around as necessary
Floating elements diagram
Common **float bug**: missing width

- Often floating block elements must have a width property value
Mario is a fictional character in his video game series. Serving as Nintendo's mascot and the main protagonist of the series, Mario has appeared in over 200 video games since his creation.
The **clear** property (cont.)

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>clear</td>
<td>disallows floating elements from overlapping this element; can be left, right, or none (default)</td>
</tr>
</tbody>
</table>
Clear diagram

```css
div#sidebar { float: right; }
p { clear: right; }
```
Mario is a fictional character in his video game series. Serving as Nintendo's mascot and the main protagonist of the series, Mario has appeared in over 200 video games since his creation.
Mario is a fictional character in his video game series. Serving as Nintendo's mascot and the main protagonist of the series, Mario has appeared in over 200 video games since his creation.
### The `overflow` Property (cont.)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>overflow</td>
<td>specifies what to do if an element's content is too large; can be auto, visible, hidden, or scroll</td>
</tr>
</tbody>
</table>
Multi-column layouts

```html
<div>
  <p>first paragraph</p>
  <p>second paragraph</p>
  <p>third paragraph</p>
  Some other text that is important
</div>
```

```css
p { float: right; width: 25%; margin: 0.5em; border: 2px solid black; }
div { border: 3px dotted green; overflow: hidden; }
```

Some other text that is important

```
third paragraph  second paragraph  first paragraph
```

output
Sizing and Positioning
### The position property (examples)

```css
div#ad {
  position: fixed;
  right: 10%;
  top: 45%;
}
```

<table>
<thead>
<tr>
<th>property</th>
<th>value</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>static</td>
<td>default position</td>
</tr>
<tr>
<td></td>
<td>relative</td>
<td>offset from its normal static position</td>
</tr>
<tr>
<td></td>
<td>absolute</td>
<td>a fixed position within its containing element</td>
</tr>
<tr>
<td></td>
<td>fixed</td>
<td>a fixed position within the browser window</td>
</tr>
<tr>
<td>top, bottom, left, right</td>
<td>positions of box's corners</td>
<td></td>
</tr>
</tbody>
</table>
Absolute positioning

```css
#menubar {
    position: absolute;
    left: 400px;
    top: 50px;
}
```

- removed from normal flow
- positioned relative to the block element containing them
- actual position determined by `top`, `bottom`, `left`, `right`
- should often specify a `width` property as well
Relative positioning

```css
#area2 { position: relative; }
```

- absolute-positioned elements are normally positioned at an offset from the corner of the overall web page
- to make the absolute element to position itself relative to some other element's corner, wrap the absolute element in an element whose position is relative
Fixed positioning

- removed from normal flow
- positioned relative to the browser window even when the user scrolls the window, element will remain in the same place

```css
#menubar {
  position: fixed;
  left: 400px;
  top: 50px;
}
```
Alignment vs. float vs. position

- If possible, lay out an element by aligning its content
  - horizontal alignment: `text-align`
  - vertical alignment: `vertical-align`
- If alignment won't work, try floating the element
- If floating won't work, try positioning the element
  - absolute/fixed positioning are a last resort and should not be overused
Details about inline elements

- Size properties \( (width, \ height, \ min-width, \ etc.) \) are ignored
- \texttt{margin-top} and \texttt{margin-bottom} are ignored,
- \texttt{but margin-left} and \texttt{margin-right} are not ignored
Details about inline elements

- The containing block element's `text-align` property controls the horizontal position of inline elements within it.
  - `text-align` does not align block elements within the page.
- Each inline element's `vertical-align` property aligns it vertically within its block element.
The vertical-align property

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vertical-align</td>
<td>specifies where an inline element should be aligned vertically, with respect to other content on the same line within its block element's box</td>
</tr>
</tbody>
</table>

- can be top, middle, bottom, baseline (default), sub, super, text-top, text-bottom, or a length value or %
  (baseline means aligned with bottom of non-hanging letters)
The display property

```css
div {
  display: inline;
  background-color: yellow;
}
```

This is a heading This is another heading

<table>
<thead>
<tr>
<th>property</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>display</td>
<td>sets the type of CSS box model an element is displayed with</td>
</tr>
</tbody>
</table>

- **values:** none, inline, block, run-in, compact, ...
- use sparingly, because it can radically alter the page layout
The **display** property

```html
<ul id="topmenu">
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>
```

```css
#topmenu li {
    display: inline;
    border: 2px solid gray;
    margin-right: 1em;
}
```

- lists and other block elements can be displayed inline
- flow left-to-right on same line
- width is determined by content
The `visibility` property

```css
p.secret {
    visibility: hidden;
}
```

- hidden elements will still take up space onscreen, but will not be shown
  - to make it not take up any space, set `display` to `none` instead
- can be used to show/hide dynamic HTML content on the page in response to events