[BC2014] [Web]

Week One in Review: HTML & CSS

Anatomy of a Website

Your Content
+
HTML (Structure)
+
CSS (Presentation)
=
Your Website

What is HTML?

HTML stands for HyperText Markup Language.

Structure of a web page

```
<html>
 <body>
  <p>Indicates a paragraph</p>
 </body>
</html>
```

This hierarchy structure is called the **DOM:The Document Object Model**

Anatomy of an HTML Tag

Each tag has a "start tag", "end tag", some content in between, and optional attributes.

```
<tagname attribute="value">
  content
</tagname>
```

Think of a tag as a "command" to the browser and of the attributes as modifiers of that command.

<!DOCTYPE html>

```
<!DOCTYPE html>
<html></html>
```

The doctype isn't an actual tag, but it needs to be at start at every HTML page to tell browser which version of HTML you're using.
The <html> tag is always the first, root tag in the page.

<head>

```
<!DOCTYPE html>
<html>
 <head>  <meta charset="utf-8">
  <title>Title of your page goes here</title>
 </head></html>
```

The <head> contains "meta" information about the page, information that the browser needs before rendering.

<body>

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Title of your page goes here</title>
 </head>
 <body>  Bulk of your content here.
 </body>
</html>
```

The <body> contains the actual content of the page.

HTML Readability

- Use quotes around attributes
- Indent to represent nesting
- Use same capitalization

Bad :(

```
<ul><li><a href=http://www.google.com>Google</a></li>
<LI>Yahoo</LI></UL>
```

Good :D

```
<ul>
  <li><a href="http://www.google.com">Google</a></li>
  <li>Yahoo</li>
 </ul>
```

<h1> Headline Tags </h1>

```
<body>
<h1>Header 1</h1>
…
<h6>Header 6</h6>
</body>
```

Header 1

…

Header 6

Div Tag

```
<body>
<div style=”color: #0000FF”>
   <h1>This is a Main Heading</h1>
   <h2>This is a Sub-Heading</h2></
div>
</body>
```

Paragraph Tags

```
<body>
<p>Paragraph 1</p>
<p>Paragraph 2</p>
<p>Paragraph 3</p>
</body>
```

Paragraph 1
Paragraph 2
Paragraph 3

Line Break Tags

```
<body>
<p>
Imagine there's no Heaven <br>
It's easy if you try <br>
No hell below us  <br>
</p>
</body>
```

Imagine there's no Heaven
It's easy if you try
No hell below us

e closed, since it doesn't encapsulate anything.

 Unordered List Tags

```
<body>
<ul> Unordered List
  <li>Item 1</li>  <li>Item 2</li>
  ...</ul>
</body>
```

Unordered List

- Item 1
- Item 2

...

Ordered List Tags

```
<body>
<ol> Ordered List
  <li>Item 1</li>  <li>Item 2</li>
  ...</ol>
</body>
```

Ordered List

1. Item 1
2. Item 2

...

<nav>Nav element </nav>

```
<body><nav>
<ul>
  <li><a href=”/”>Home</a></li>
  <li><a href=”/about”>About</li>
  <li><a href=”/works”>Works</li>
</ul>
</nav>
</body>
```

Navigation

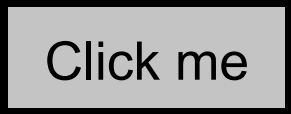
Home

About

Works

<button>button tag </button>

```
<body>
<button>Click me</button>
</body>
```



<!-- HTML Comments -->

```
<!-- I can comment for humans here. --><!--
I can write a comment that
spans multiple lines too.-->
```

<img src = “Images”>

```
<img src="http://www.google.com/images/srpr/
logo1w.png"
  alt="Google">
```


```
<img src="http://www.google.com/images/srpr/logo1w.png"
  alt="Google" height="50">
```

<a href= “Links”>

```
<a href="http://www.google.com">Google</a>
```

Google

```
<a href="http://www.google.com">
  <img src="http://www.google.com/images/srpr/logo1w.png"
    alt="Google"></a>
```

(in new window)

```
<a href="http://www.google.com" target="_blank">Google
(in new window)</a>
```

<id= “id’s”>

```
<h1 id="chapter1">Chapter 1</h1>

<p>Most exciting story in the world. To be continued...</p>
```

Every HTML element can carry the id attribute. It is used to uniquely identify that element from other elements on the page. Its value should start with a letter or an underscore (not a number or any other character).

<class= “classes”>

```
<h1 class ="warning">Warning!!!</h1>

<p>Most exciting story in the world. To be continued...</p>
```

Every HTML element can also carry a class attribute.
Sometimes, rather than uniquely identifying one element within a document, you will want a way to identify several elements as being different from the other elements on the page.

<a href= “#InternalLinks”>

```
<p>Jump to <a href="#chapter1">the first chapter!</a> </p><h1
id="chapter1">Chapter 1</h1>

<p>Most exciting story in the world. To be continued...</p>

<p>Read more on<a href="http://en.wikipedia.org/wiki/
Fox#Diet">Wikipedia</a></p>
```

CSS

What Is CSS?

CSS = Cascading Style Sheets
CSS is a "style sheet language" that lets you style the elements on your page.
CSS is embedded inside HTML, but it is ***not*** **HTML** itself

<style> CSS In HTML </style>

CSS can be embedded in HTML in several ways. One way is to include all CSS in a <style> tag, usually inside the head tag:

```
<html>
<head><style>body {  color:
yellow;
  background-color: black;}
</style></head>
```

<link rel =”External CSS File”

CSS can also be defined in a separate file.
Linked stylesheet:

```
<head><link rel="stylesheet" type="text/css"
href="main.css">
</head>
```

<link rel =”Google Fonts”

To add google fonts all you have to do is add a special stylesheet link to your HTML document, then refer to the font in a CSS style.
Linked stylesheet:

```
<head><link rel="stylesheet" type="text/css"
href="http://fonts.googleapis.com/css?family=Font
+Name">
</head>
```

<link rel =”Google Fonts”

Linked stylesheet:

```
<head><link rel="stylesheet" type="text/css"
href="http://fonts.googleapis.com/css?
family=Tangerine">
</head>
```

Assigned font style:

```
<style>
    body {
      font-family: 'Tangerine', serif;      font-size:
48px;
    }</style>
```

Anatomy of CSS

CSS consists of "style rules". Each style rule consists of a "selector" and "declarations" of property-value pairs:

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

```
body {
  color: yellow;
  background-color: black;
}
```

/*Comments*/

```
/* Comment here */p
{
  margin: 1em; /* Comment here */
  padding: 2em;
  /*   color: white; */
  background-color: blue;
}/*    multi-line
  comment here*/
```

/*Comments*/ will be ignored by the browser and are useful for documenting your styles to other humans or commenting out rules.

Selectors

The Selector

The selector is used to select which elements in the HTML page will be given the styles inside the curly braces.

```
selector {
  property: values;
}
```

Selector Type: Element

Selects all p elem

```
p {
}
```

Selector Type: #id

```
#header {
}
```

Selects any element with the id “header", e.g. The "#" is how you tell CSS "this is an id.”

Element ids are uni

```
<p id="header"></p>
```

Selector Type: .class

```
.warning {
  color: red;
}
```

Selects any element with the class name “warning”, e.g. The "." is how you tell CSS "this is a class name."

Multiple elements c

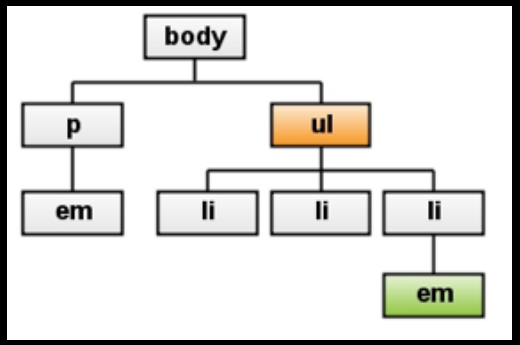
```
<p class="warning"></p>
```

Selector Type: Position In Doc

```
ul em {
  color: yellow;
}
```

Selects any **em** element that's a descendant of a **ul** element.
The " " (space) is how you say "find a descendant."

```
<body>
 <ul>
  <li><em>Hi</em></li>
 </ul>
 <p><em>Bye</em></p>
</body>
```



Selector Type: #id + position

```
#related-brands li {
  color: gray;
}
```

Selects any **<li>** element that is a descendant of any element with an id that equals **"related-brands."**

```
<ul id="related-brands">
  <li>Rice Krispies
  <li>NutriGrain
</ul>
```

Selector Type: element + .class

```
li.special {
  color: yellow;
}
```

Selects any **<li>** element with a class attribute that contains the word **"special".**

Warning: If you plac *cendants of* ***<li>*** *with class of* ***"spec***

```
<ul>
  <li>Rice Krispies
  <li class="special">NutriGrain
</ul>
```

Selector Type: pseudo classes

A set of "pseudo classes" can style anchor elements depending on their state.

```
a:link { /* unvisited link */
  color: red;
}a:visited { /* visited link */
  color: blue;
} a:hover { /* moused over link */
  color: green;
}  a:active { /* current link */
  color: purple;
} a:focus {  /* focused link */
  color: purple;
}
```

Grouping Selectors

You can group selectors to apply the same style to all of the selectors by separating them with commas:

```
a:hover, a:active, a:focus {
  background-color: yellow;
}
```

Cascade Rules

Generally:

- id is more specific than a class, class is more specific than element.
- the longer the selector, the more specific it is
- If style rules are equally specific, the last one wins!

Example:

```
#a #b h1 { color: red; } /* winner! */#a h1 { color: blue; }
```

Properties

Property Value Pairs

Each property can have one or more comma separated values.

```
font: italic 12px sans-serif;
color: #333;
background-color: red;
font-family: Arial, sans-serif;
```

Property: color

The **"color"** property changes the text color. Colors can be specified either by name, for the most common colors, or by hexadecimal value.

```
color: red;color: #ff0000;color: rgb(255, 0, 0);
```

This property is inherited, which *dant elements but can be overridden* *all the body text red unless specified otherwise:*

```
body {
  color: red;
}
```

Property: background-color

The **"background-color"** property changes the background color. Besides the BODY element, all elements default to a transparent background.

```
background-color:
black;background-color:
#000000;background-color:
rgb(0,0,0);
```

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

table {
  background-color: #FFCC00;
}
```

Property: font-family

The **"font-family"** property specifies the font family (or "font face") of the text. You can specify either a specific font name or a generic family name (serif, sans-serif, monospace, cursive).

```
font-family: sans-serif;
```

A comma separated list of font families can be specified if you want the browser to prefer one but use the others as backup options.

```
font-family:  "Times New Roman", serif;font-family: "Arial", sans-serif;font-family: Courier, monospace;
```

Property: font-size

The **"font-size"** property specifies the size of a font. It can be specified as a fixed size in various units, a percentage, or as a predefined keyword.

```
font-size: 1.5em;font-size: 12px;font-size:
100%;font-size: larger;
```

Property: font-style

The **"font-style"** property specifies the font style of the text, either **"normal"** by default or **"italic"**.

```
font-style: italic;
```

Property: font-weight

The **"font-weight"** property specifies the thickness of the font. The default is **"normal"** and the typical override is "bold". You can also specify or a number from 100 to 900.

```
font-weight: bold;
```

“Shorthand” Properties

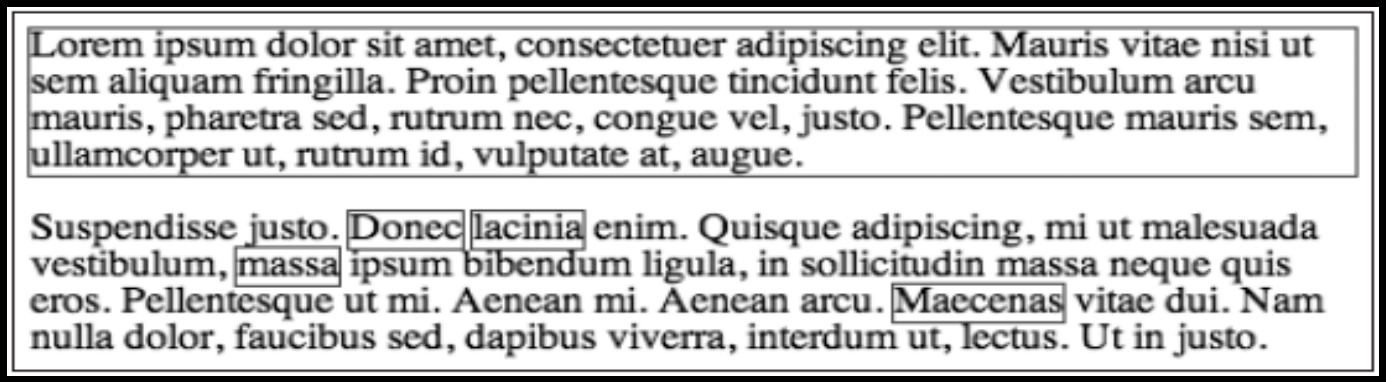
A "shorthand" property in CSS lets you specify multiple properties in one property, for conciseness purposes. Instead of specifying each **"font-"** property separately, you can bundle them up in one **"font"** property.

Those four rul

```
table.geeky {
  font-weight: bold;  font-style: italic;
  font-size: 10px;
  font-family: sans-serif;
}
```

```
table {
  font: italic bold 10px sans-serif;
}
```

Block & Inline Elements



A block-level box, such as a div, a paragraph, or a heading, begins rendering on a new line.

An inline-level box, such as a an , begins rendering wherever you place it within the document and does not force any line breaks.

Block & Inline Elements

```
<ul id="maintabs">
<li><a href="#">PersonalFinance</a></li>
<li><a href="#">BillPay</a></li>
<li><a href="#">FundsTransfer</a></li>
<li><a href="#">FinancialCalendar</a></li>
<li><a href="#">Customer Care</a></li>
```

Using CSS property, display, you can make an inline element display like a block-level element or vice-versa.

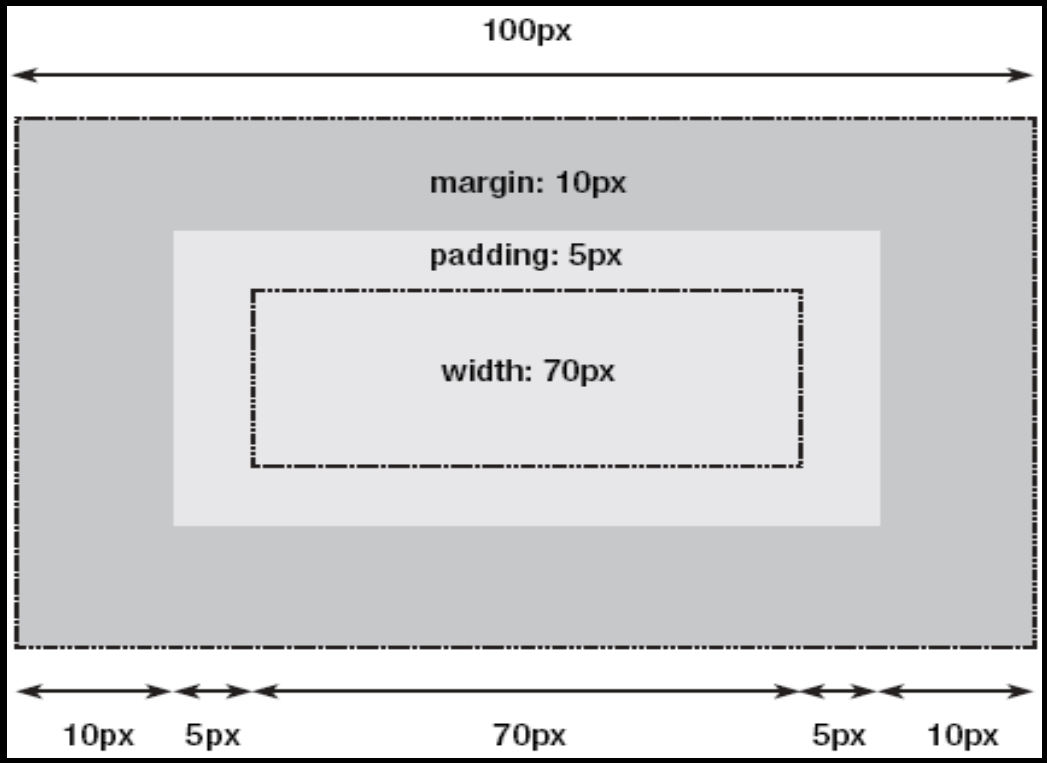
Block & Inline Elements

```
#maintabs li {
display:inline;
list-style-type:none;
float:left;
background-color:#dce2c7;
border:1px solid #c5d199;
padding:5px 10px;
```

Using CSS property display:inline, you can display a block element <li> as an inline element and can show the list items <li> as tabs.

Box Model

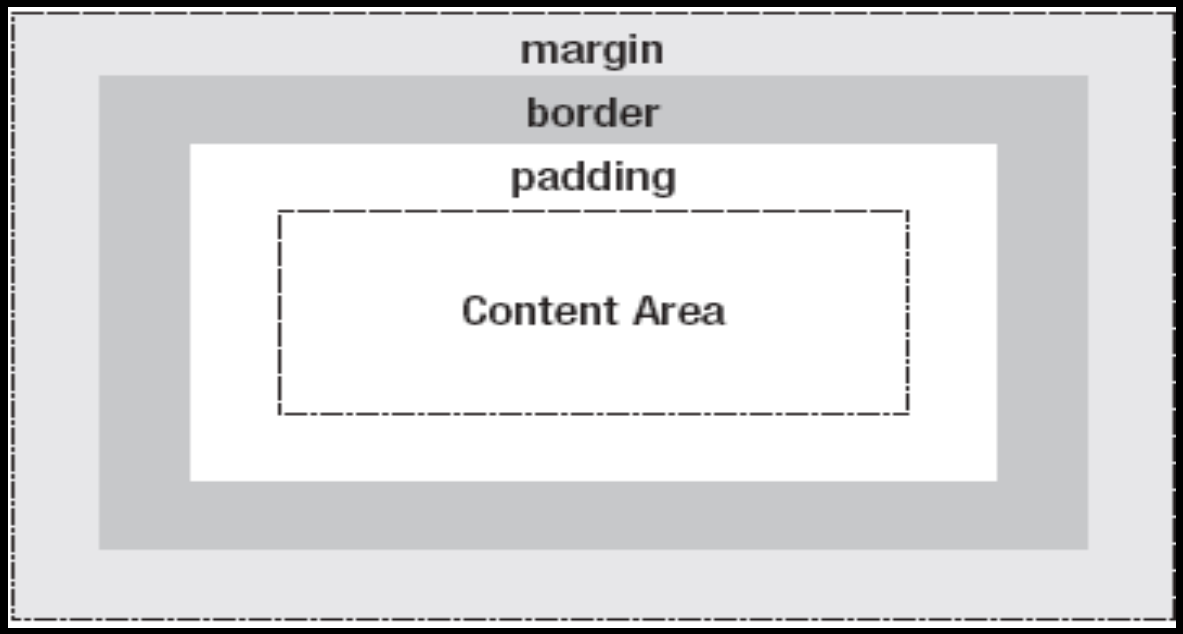
```
#myBox {
margin:10px;
padding:5px;
width:70px;
```



Positioning Elements

Padding is applied around the content area.
If you add a background to an element, it will be applied to the area formed by the content and padding.
Border applies a line to the outside of the padded area.
Outside the border is a margin.

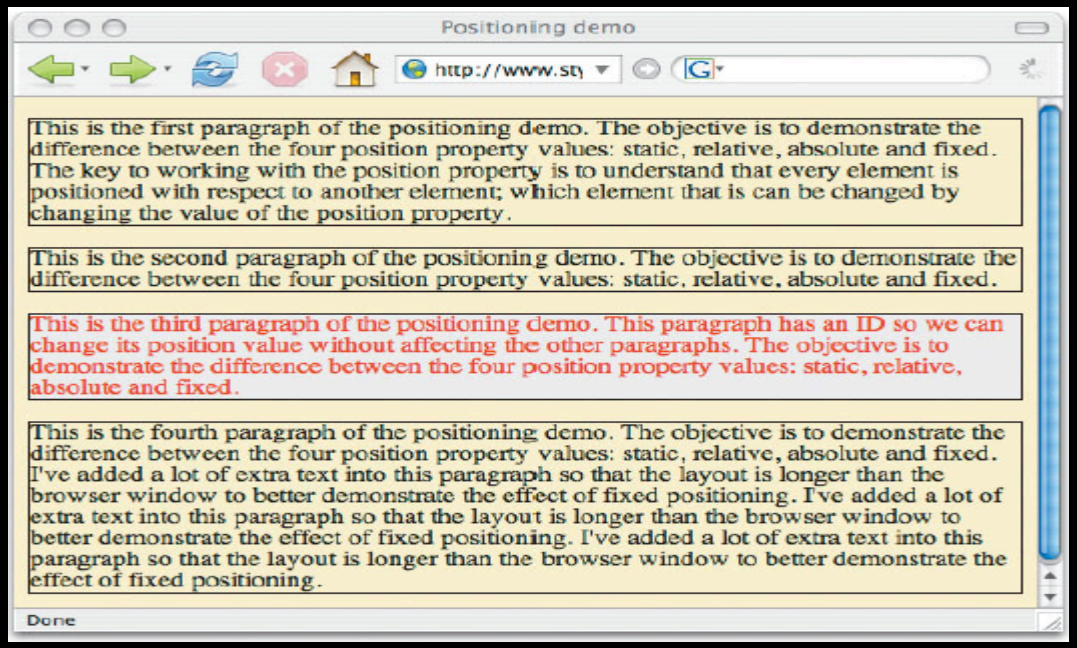
*Margins are transparent and cannot be seen.
** They are generally used to control the spacing between elements.



Every element on the page is considered to be a rectangular box made up of the element's content, padding, border, and margin.

Static Positioning

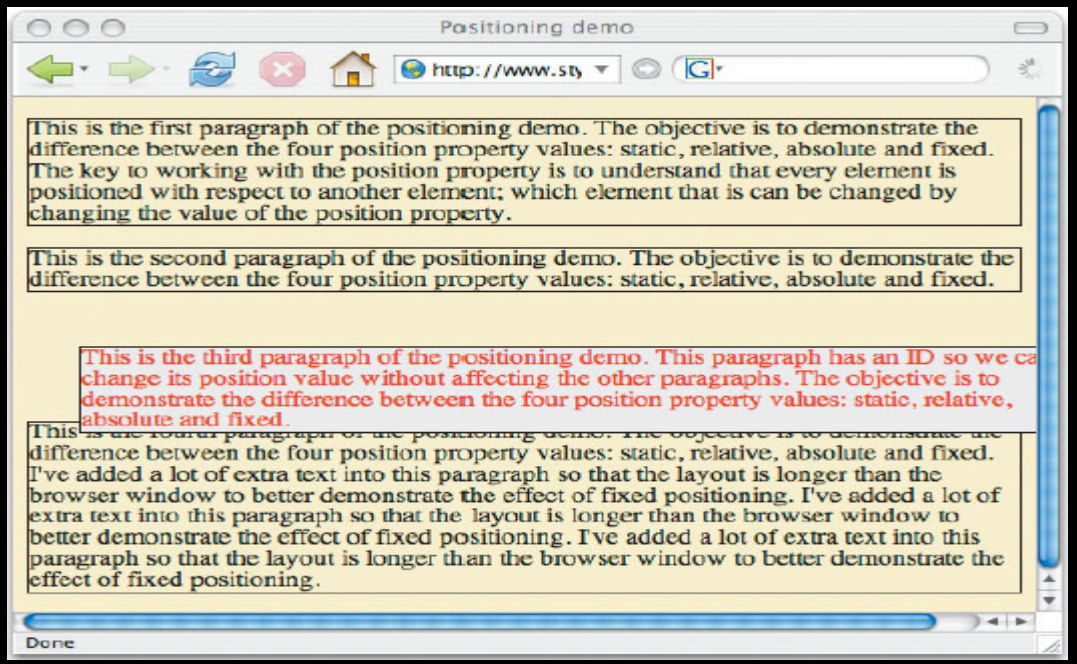
With the four paragraphs each displayed with the default position property value, static, they stack one above the other, as normal document flow dictates.



Relative Positioning

You can then move this paragraph with respect to its default position using the properties top, left,

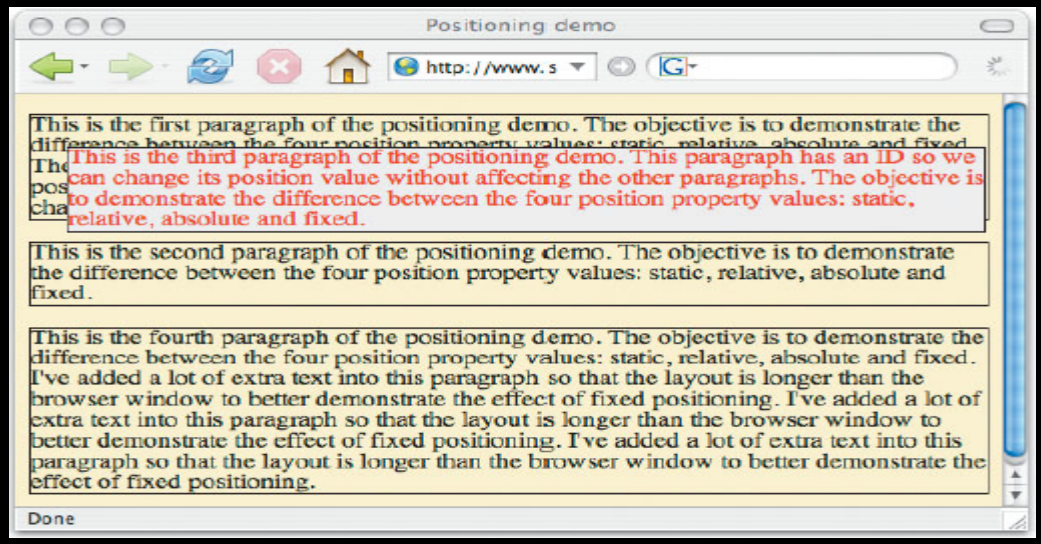
```
p#specialpara {
      position:relative;
      top:30px;
      left:20px;
```



Absolute Positioning

This type of positioning takes an element entirely out of the flow of the document.
Let's modify the code you used for relative positioning by changing relative to absolute

```
p#specialpara {
        position:absolute;
        top:30px; left:20px;
}
```



Fixed Positioning

Fixed positioning is similar to absolute positioning, except that the element's positioning context is the viewport (the browser window or the screen of a handheld device, for example).

So the element does

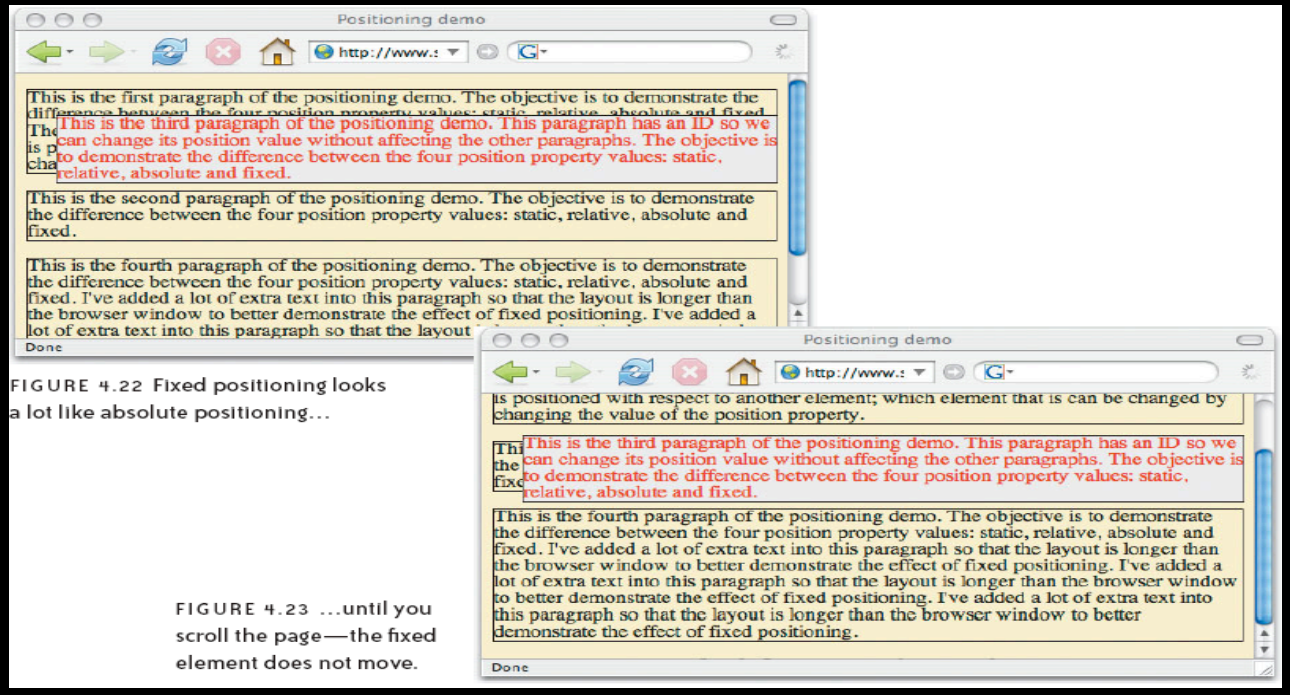
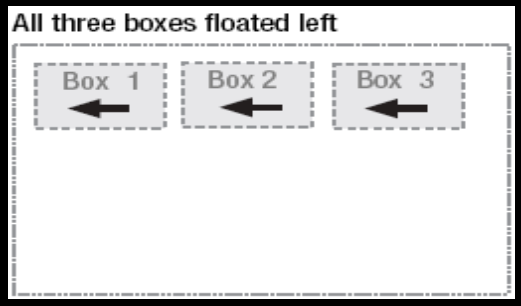


FIGURE 4.22 Fixed positioning looks a lot like absolute positioning...

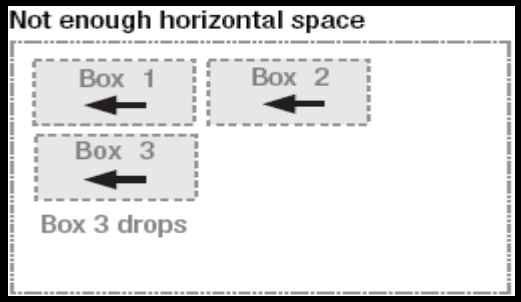
FIGURE 4.23 ...until you scroll the page—the fixed element does not move.

Floating

If you float all three boxes to the left, Box 1 is shifted left until it touches its containing box, and the other two boxes are shifted left until they touch the preceding floated box.



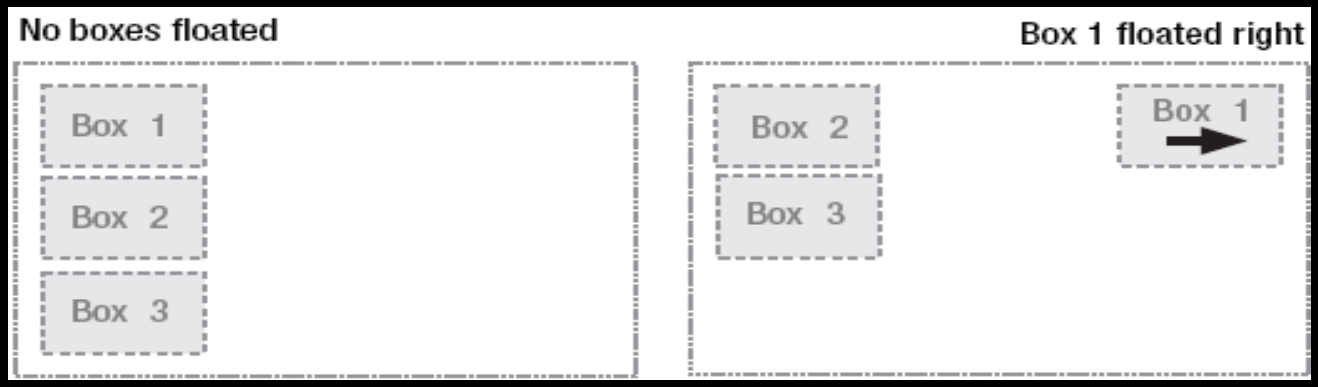
If the containing block is too narrow for all of the floated elements to fit horizontally, the remaining floats will drop down until there is sufficient space.



Floating

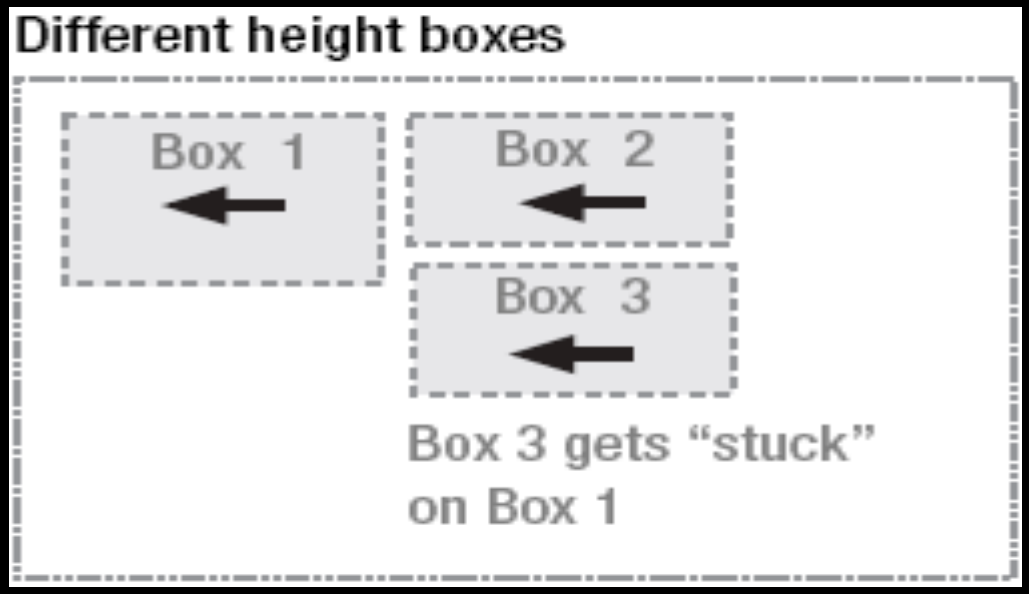
A floated box can either be shifted to the left or the right until its outer edge touches the edge of its containing box, or another floated box.

Because floated boxes aren't in the normal flow of the document, block boxes in the regular flow of the document behave as if the floated box wasn't there.



Floating

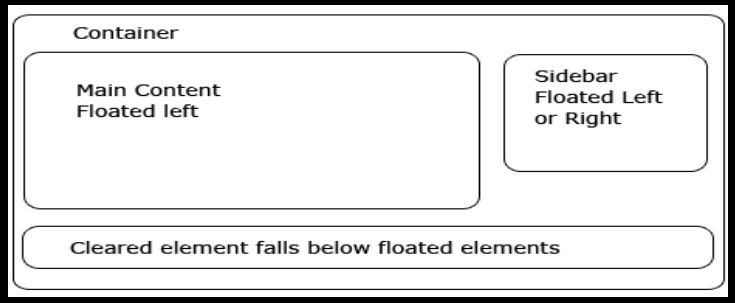
If the floated elements have different heights, it is possible for floats to get “stuck” on other floats when they drop down.

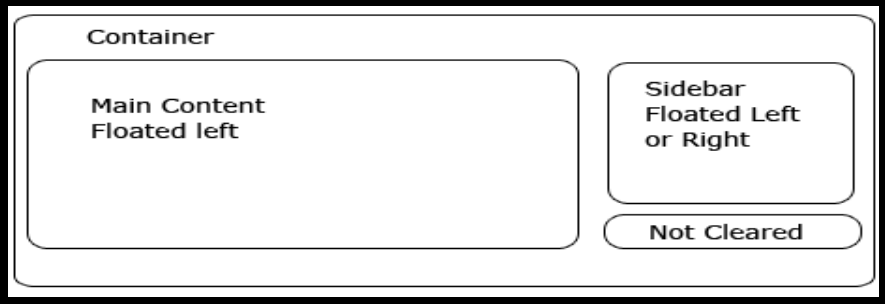


Clearing The Floats

Clearing Floats

Clear takes the values **left**, **right**, **both**, **none** (default), and **inherit**. In practice you'll only use the first 3 and most of the time you'll use clear: both. Perhaps the most common use is to clear your footer div so it sits below your 2 or 3 floated columns.





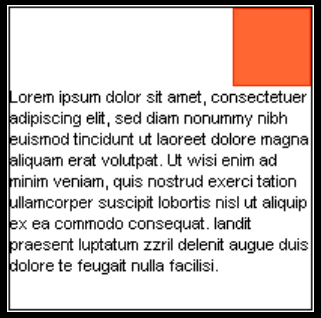
Clearing The Floats

clear: left

The element is moved below the bottom outer edge of any left-floating boxes that resulted from elements earlier in the source document.

clear: right

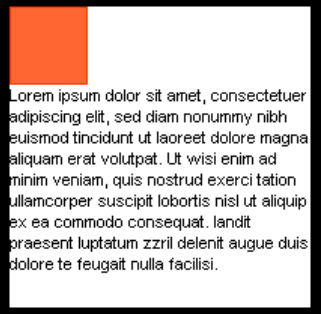
The element is moved below the bottom outer edge of any right-floating boxes that resulted from elements earlier in the source document.



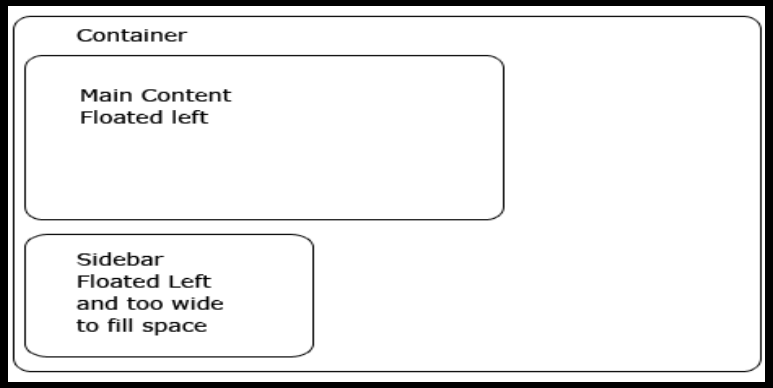
Clearing The Floats

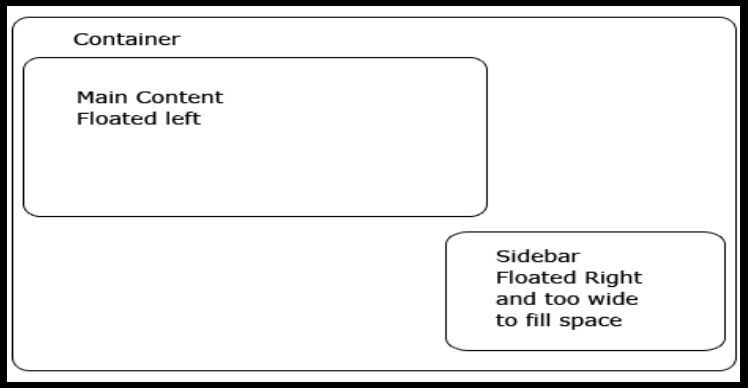
clear: both

The element is moved below all floating boxes of earlier elements in the source document.

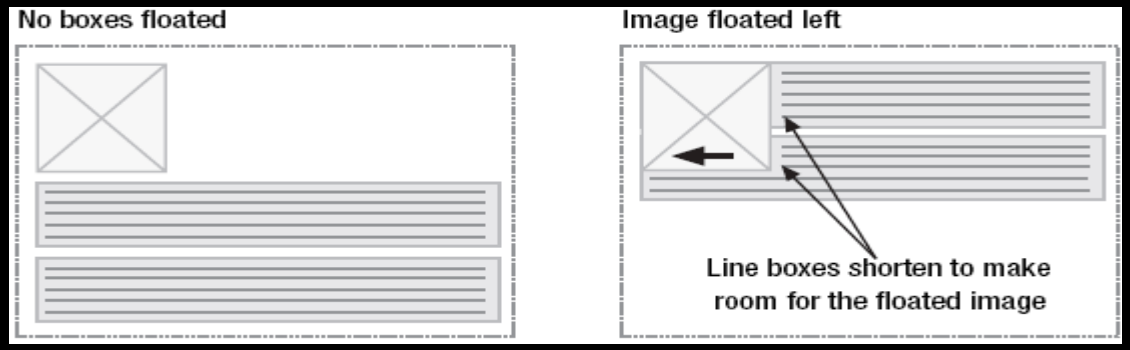


Containing Floats





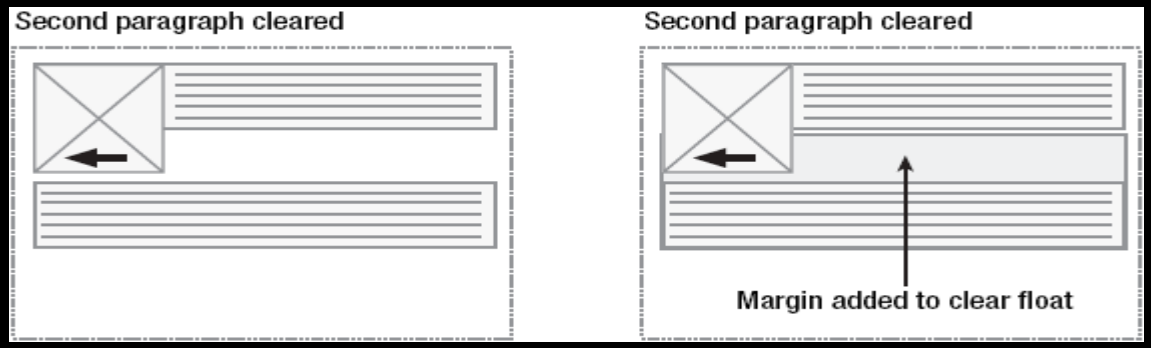
Line Boxes And Clearing



Line boxes (text boxes) next to a floated box are shortened to make room for the floated box, and flow around the float.

In fact, floats were created to allow text to flow around images

Line Boxes And Clearing



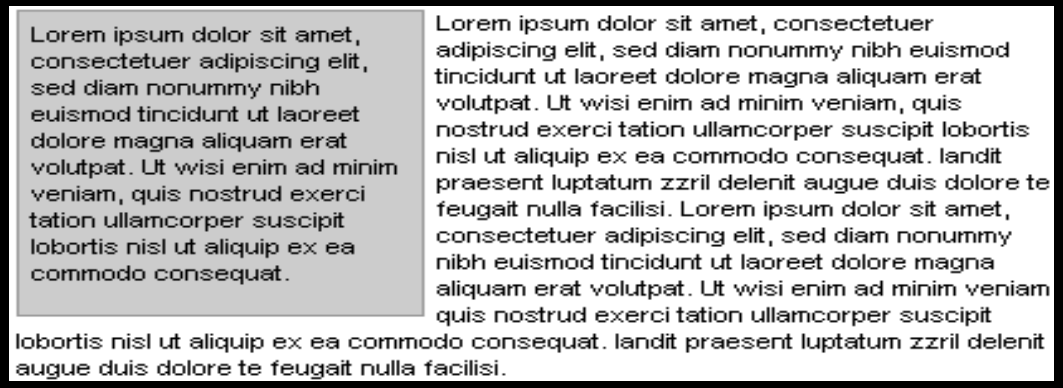
To stop line boxes flowing around the outside of a floated box, you need to apply a clear to that box.

The clear property can be left, right, both, or none, and indicates which side of the box should not be next to a floated box. To accomplish this, enough space is added above the cleared element's top margin to push the element's top border edge vertically down past the float

Overflow Property

```
<body>
    <div class="myBox">This is some text...
    </div>
This is some text...
```

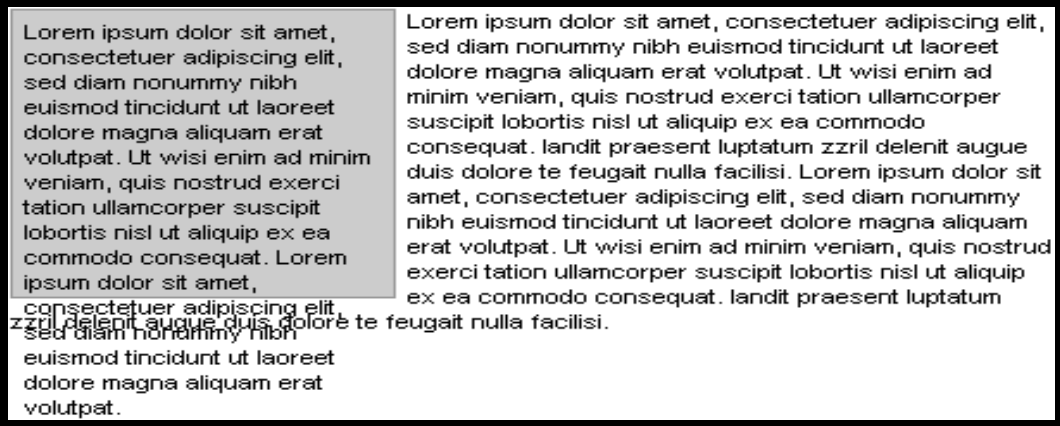
```
.myBox {
float: left;
width:150px;
height:150px;
background-color:#ccc;
border: 1px solid #999;
margin-right:5px;
```



Overflow: Visible

Visible is the default value. No scrollbars will be added, and your content will just flow.

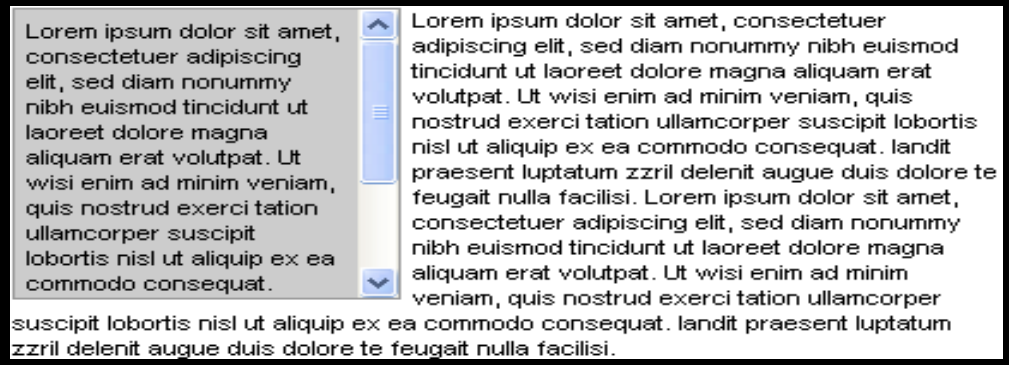
```
.myBox {
    overflow:visible;
}
```



Overflow: Auto

You should use this value when you want to let the browser decide what's right. Scroll down? Scroll right? No scrolling at all? The browser makes this decision. This value usually is the best choice.

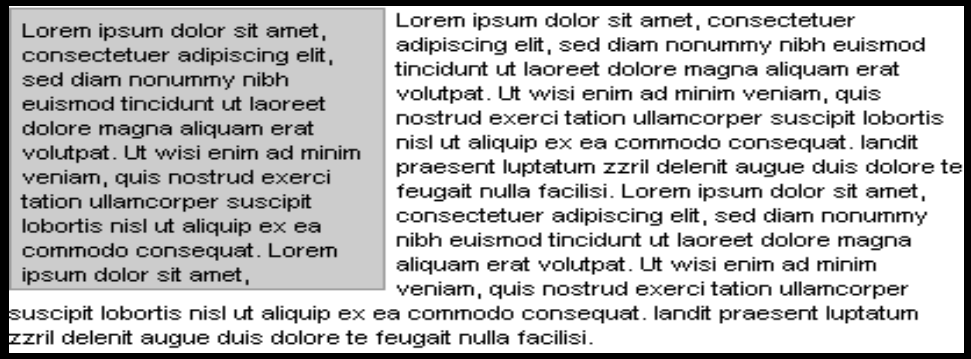
```
.myBox {
      overflow:auto;
}
```



Overflow: hidden

This value will not add any scrollbars or will not display more text then needed. When the content crosses the ‘height’ given to the container, it simply don’t display that content.

```
.myBox {
        overflow:hidden;
}
```



Overflow: scroll

An overflow of scroll means that the browser should place scrollbars on the element whether or not the contents of the element have overflowed.

Ensure there is always enough content in this box, because otherwise there are ugly scrollbars for nothing!

```
.myBox {
        overflow:scroll;
}
```