POLITECNICO DI TORINO

e-Lite

Style & Layout in the web: CSS and Bootstrap

Ambient intelligence: technology and design

Fulvio Corno

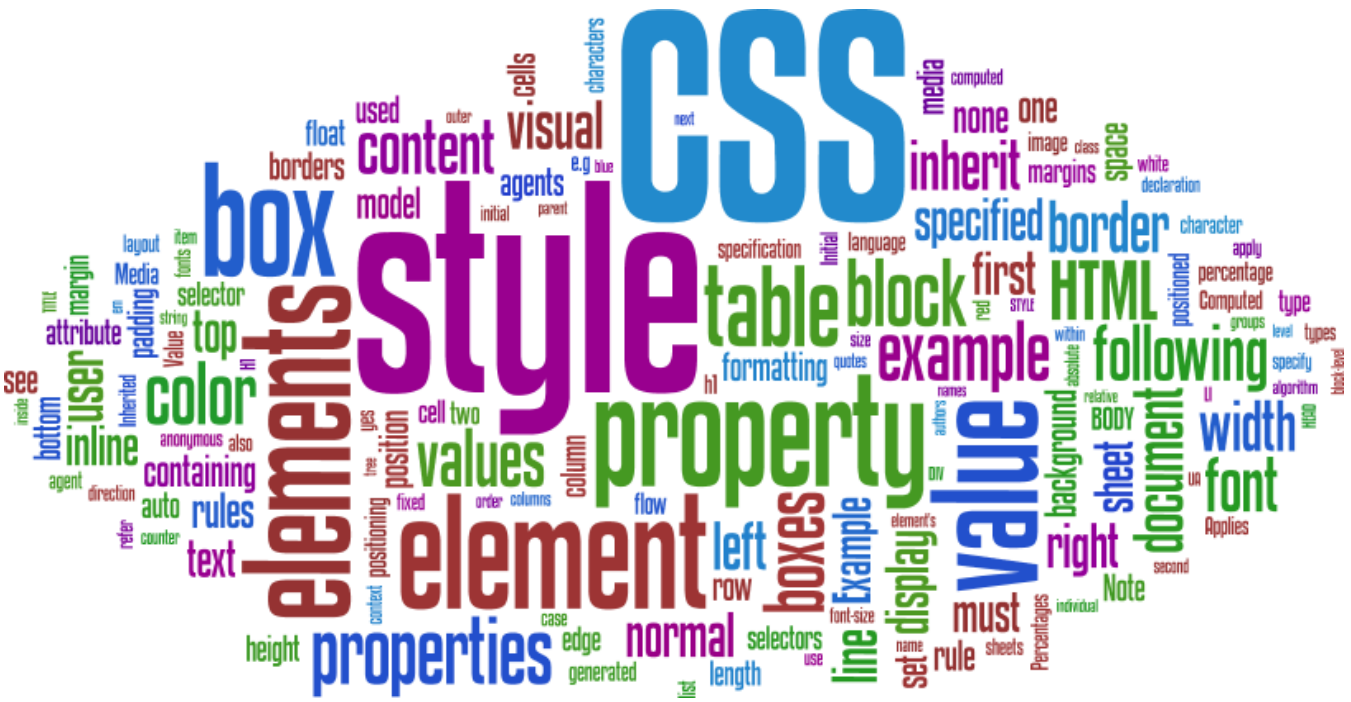
Politecnico di Torino, 2014/2015

Goal

- Styling web content
- Advanced layout in web pages
- Responsive layouts
- Libraries

Summary

- CSS – Cascading Style Sheets
- The Bootstrap framework



Style & Layout in the web

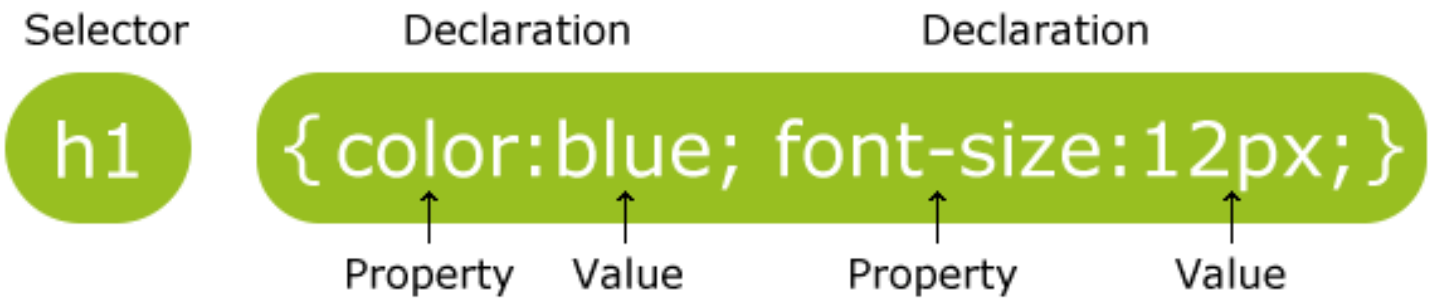
CSS – CASCADING STYLE SHEETS

Cascading Style Sheets

- CSS: Cascading Style Sheet
- CSS 1: W3C recommendation (17 Dec 1996)
- **CSS 2.1: W3C Recommendation (7 June 2011)**
- CSS 3: Working Draft
- Resources:
  - CSS 2.1 standard, http://www.w3.org/TR/CSS21/
  - W3C CSS Home: http://www.w3.org/Style/CSS/
  - W3C CSS Tutorial, http://www.w3.org/Style/Examples/011/firstcss

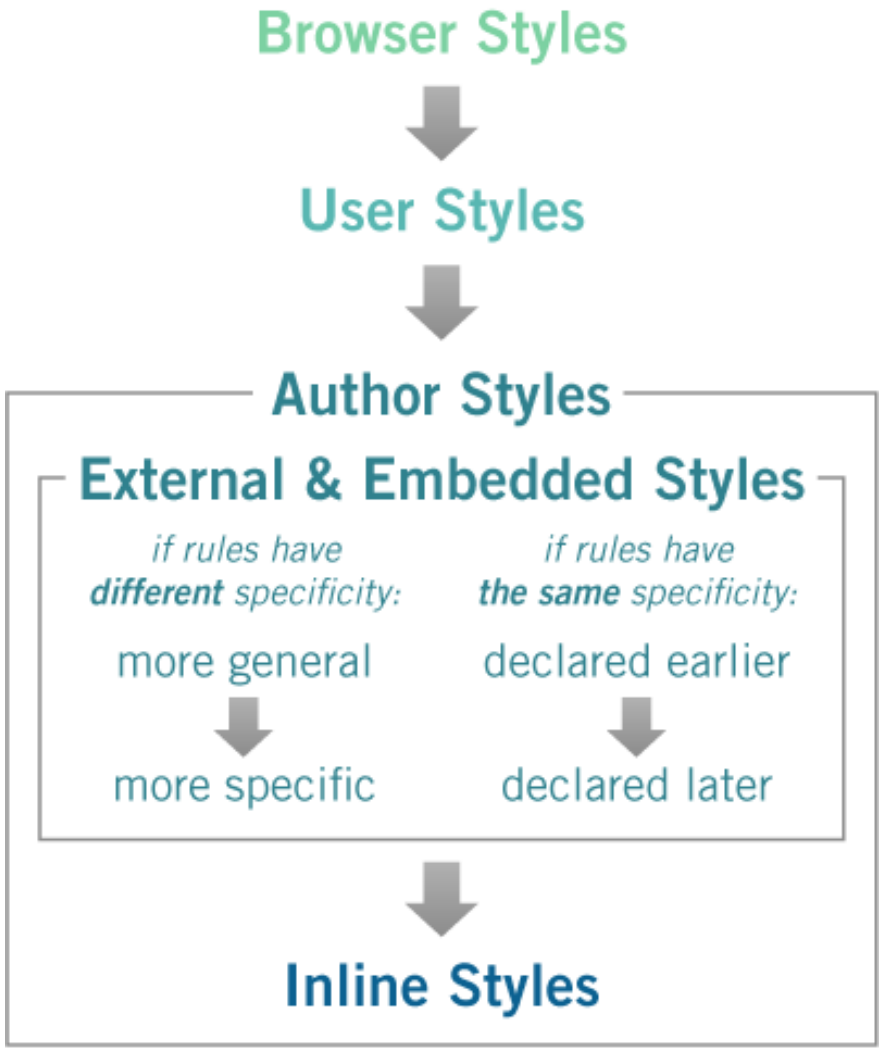
CSS Syntax

- CSS is based on rules
- A rule is a statement about one stylistic aspect of one or more XHTML elements
- A style sheet is a set of one or more rules that apply to an XHTML document



Cascading Style Sheets

- The term “cascading” means that a document can include more than one style sheet
- In this case, visualization follows **priority** rules



External style

- Link to an external style sheet using the <link> element

style.css

```
h1 { font-size:17px;
   font-family:verdana; color:green; }
h2 { font-size:18px;
   font-family:arial; color:red; }
```

```
<head>
<link rel=stylesheet type="text/css"
      href="style.css">
</head>
<body>
<h1>Green text on verdana 17 pixel font</h1>
<h2>Red text on arial a 18 pixel font</h2>
</body>
```

External style

- Alternative method
- @import directive in the <style> element

```
<head>
  <style>
    @import url(style.css);
  </style>
</head>
<body>
...
</body>
```

Internal style

- <style> element inside the document header

```
<head>
<style type="text/css">
h1 { font-size:17px; font-family:verdana;
color:green; }
h2 { font-size:18px; font-family:arial;
color:red; }
</style>
</head>
```

Inline style

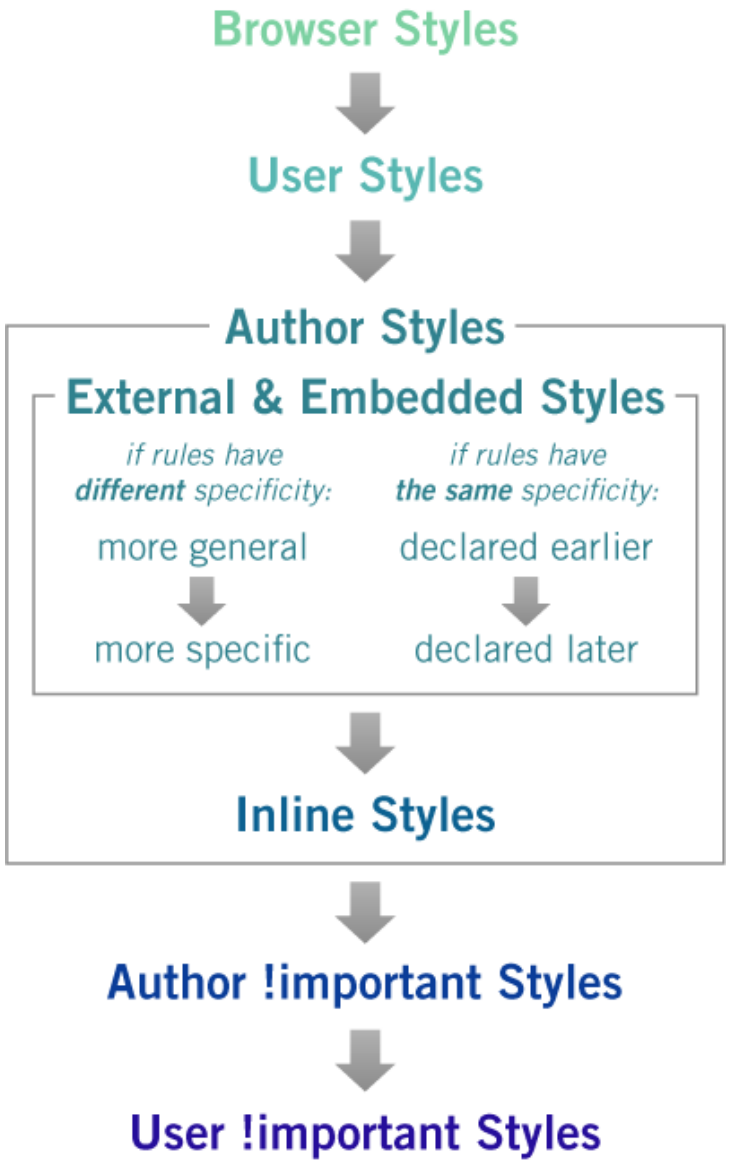
- <style> attribute within an XHTML element

```
<h1 style="font-size:17px;
  font-family:verdana; color:green; "> Green text
on verdana 17 pixel font </h1>
```

Priority rules

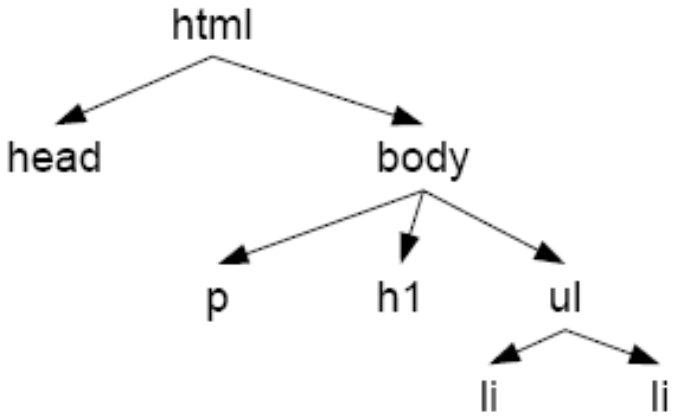
- Rules can be marked as “important”

```
h1 {
  color:red !important
}
```



Tree structure and inheritance

- XHTML documents are trees
- Styles are inherited along trees



- When two rules are in conflict the most specific wins
- Example
  - body {color: green}
  - h1 {color: red}

Main Selectors

Selector	E	.cc	#uu
HTML	<E>	<E class="cc">	<E id="uu">

Pattern	Meaning
*	Matches any element.
E	Matches any E element (i.e., an element of type E).
E F	Matches any F element that is a descendant of an E element.
E > F	Matches any F element that is a child of an element E.
E:first-child	Matches element E when E is the first child of its parent.
E:link E:visited	Matches element E if E is the source anchor of a hyperlink of which the target is not yet visited (:link) or already visited (:visited).
E:active E:hover E:focus	Matches E during certain user actions.
E:lang(c)	Matches element of type E if it is in (human) language c (the document language specifies how language is determined).
E + F	Matches any F element immediately preceded by a sibling element E.
E[foo]	Matches any E element with the "foo" attribute set (whatever the value).
E[foo="warning"]	Matches any E element whose "foo" attribute value is exactly equal to "warning".
E[foo~="warning"]	Matches any E element whose "foo" attribute value is a list of space-separated values, one of which is exactly equal to "warning".
E[lang|="en"]	Matches any E element whose "lang" attribute has a hyphen-separated list of values beginning (from the left) with "en".
DIV.warning	*Language specific.* (In HTML, the same as DIV[class~="warning"].)
E#myid	Matches any E element with ID equal to "myid".

http://www.w3.org/TR/css-2010/#selectors

Pseudo class selector

- Used to style an element based on something other than the structure of the document
  - E.g., the state of a form element or link

```
/* makes all unvisited links blue */
a:link {color:blue;}
/* makes all visited links green */
a:visited {color:green;}
/* makes links red when hovered or activated */
a:hover, a:active {color:red;}
/* makes table rows red when hovered over */
tr:hover {background-color: red;}
/* makes input elements yellow when focus is applied */
input:focus {background-color:yellow;}
```

Meaningful XHTML

- Meanginful elements
  - h1, h2, ...
  - ul, ol, and dl
  - strong and em
  - blockquote and cite
  - abbr, acronym, and code
  - fieldset, legend, and label
  - caption, thead, tbody, and tfoot
  - HTML5 adds many new “semantic” elements
- `id` and `class` names
  - Allow to give extra meaning
- `div` and `span`
  - Add structure to document

DIV element

- Stands for “division”
- Used to group block-level elements
  - Provides a way of dividing a document into meaningful areas
- Use only if necessary and not redundant

```
<div id="mainNav">
  <ul>
    <li>Home</li>
    <li>About Us</li>
    <li>Contact</li>
  </ul>
</div>
```

```
<ul id="mainNav">
  <li>Home</li>
  <li>About Us</li>
  <li>Contact</li>
</ul>
```

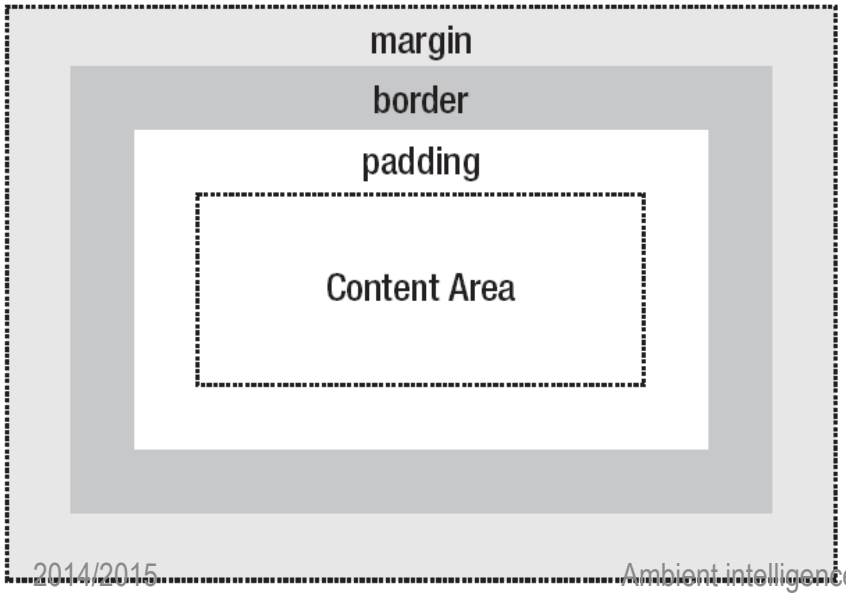
SPAN element

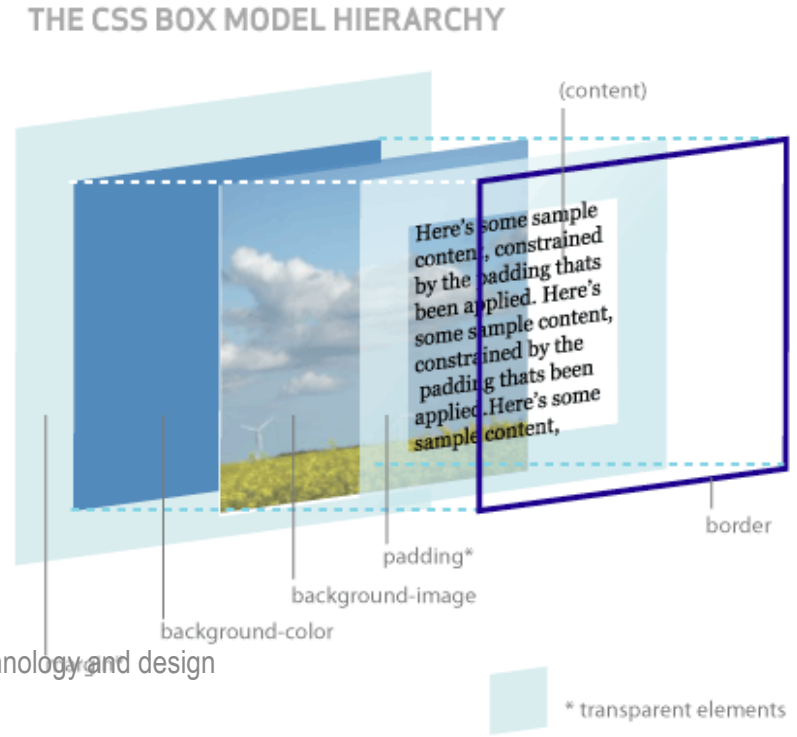
- Used to group or identify inline elements

```
<h2>Where's Durstan?</h2>
<p>Published on
     <span class="date">March 22nd, 2005</span>
by <span class="author">Andy Budd</span></p>
```

The box model

- One of the cornerstones of CSS
- Dictates how elements are displayed and, to a certain extent, how they interact with each other
- Every element on the page is considered to be a rectangular box





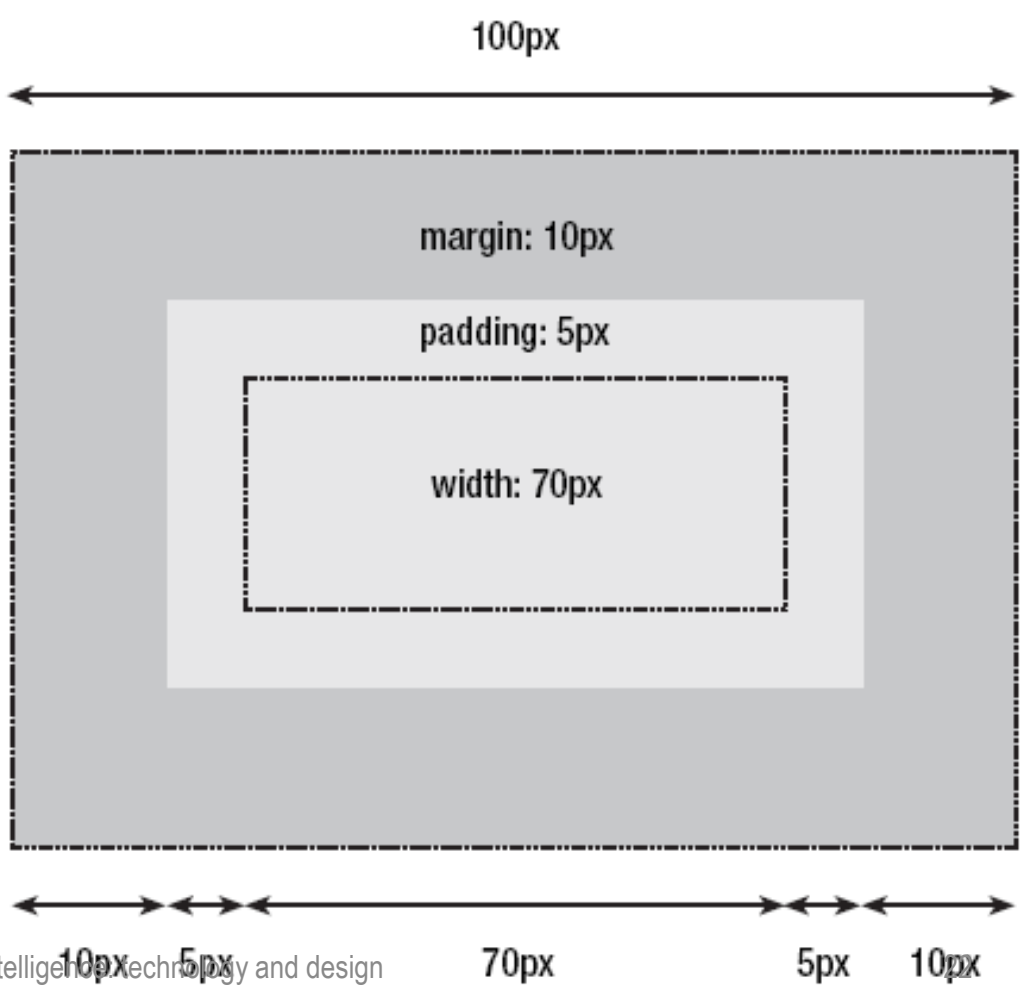
The box model

- Content
  - The content of the box, where text and images appear
- Padding
  - Clears an area around the content
  - The padding is affected by the background color of the box
- Border
  - A border that goes around the padding and content
  - The border is affected by the background color of the box
- Margin
  - Clears an area around the border
  - The margin does not have a background color, it is completely transparent

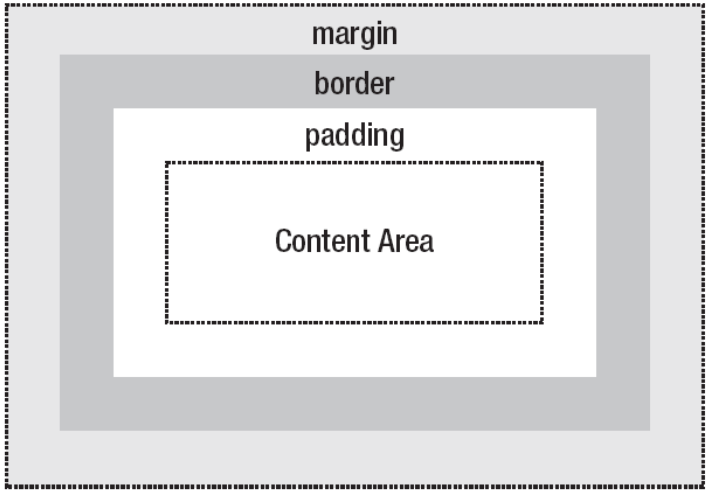
Example

- Padding, borders, and margins are optional and default to zero

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



The box model



- Total element width = width + left padding + right padding + left border + right border + left margin + right margin
- Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin
- Example
  - W3Schools.com
  - http://www.w3schools.com/Css/css_boxmodel.asp

Positioning schemes

- Three basic positioning schemes in CSS
  - Normal flow
  - Floats
  - Absolute positioning
- Unless specified, all boxes start life being positioned in the normal flow
  - The position of an element's box in the normal flow will be dictated by that element's position in the (X)HTML

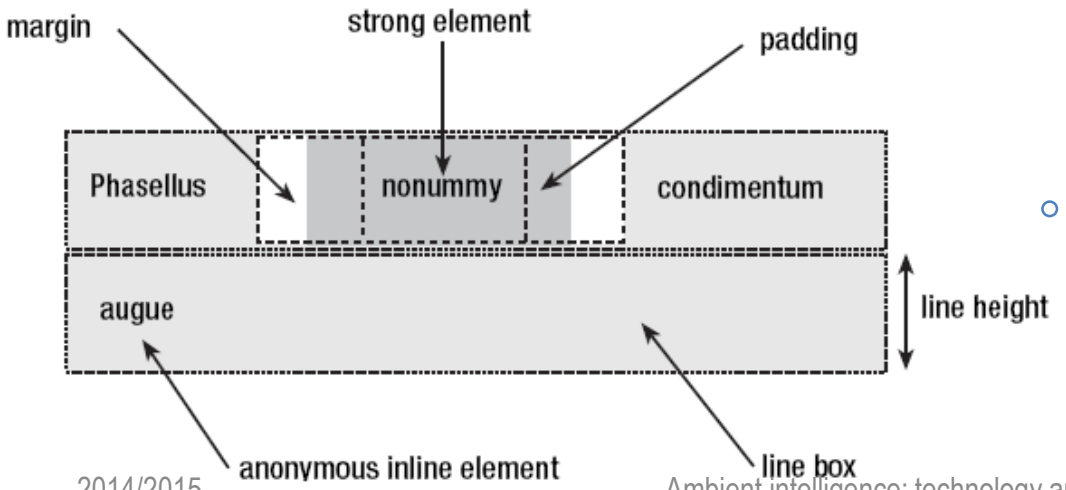
Normal flow

- **Block-level** boxes will appear vertically one after the other

```
<div> … </div>
```

  - The vertical distance between boxes is calculated by the boxes' vertical margins
- **Inline** boxes are laid out in a line horizontally

```
<span> … </span>
```



  - Their horizontal spacing can be adjusted using horizontal padding, borders, and margins
  - Vertical padding, borders, and margins will have no effect on the height of an inline box

Display property

- Allows to control element visualization (block or inline)
- Changing an inline element to a block element, or vice versa, can be useful for making the page look a specific way
- Example
  - W3Schools.com
  - http://www.w3schools.com/Css/css_display_visibility.asp

```
li {display:inline;}
```

```
span {display:block;}
```

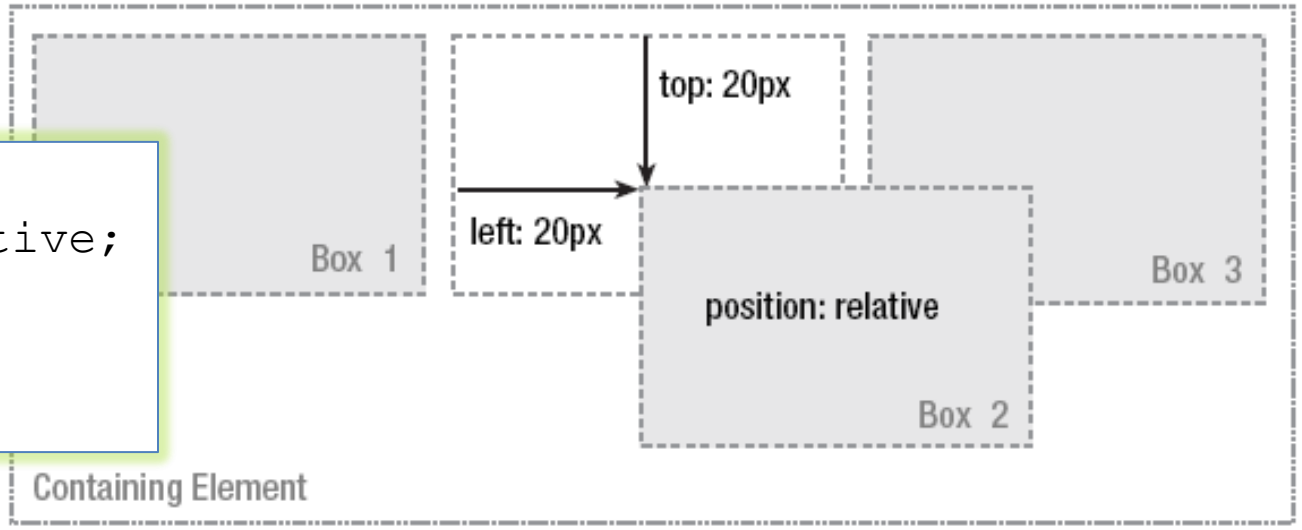
Box Positioning

- A block can be positioned in different ways to which correspond different positioning schemes
  - **Static**: normal block
  - **Relative**: the offset values are relative to the block position in the normal flow. If a relative block B follows a relative block A, the offset is respect to the position of A without the offset
  - **Absolute**: the box position is determined by the top, left, right, bottom properties and is relative to the containing block
  - **Fixed**: the box is fixed with respect to some reference (the viewport as an example)

Relative positioning

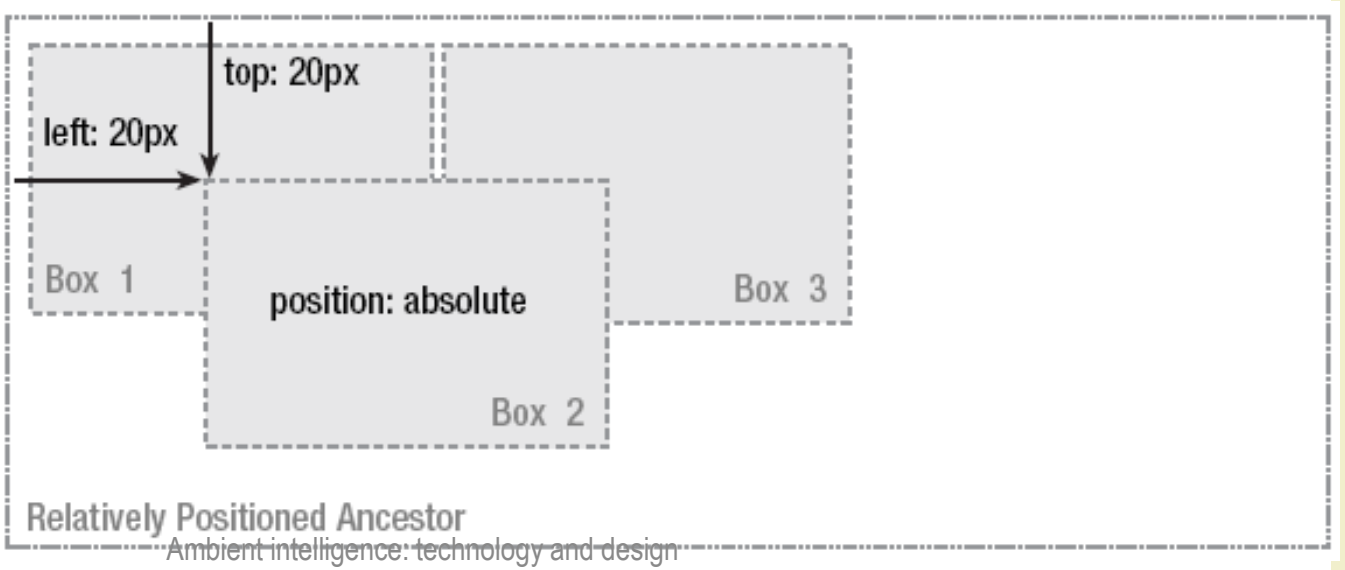
- It is possible to shift one element “relative” to its starting point by setting a vertical or horizontal position

```
#myBox {
  position: relative;
  left: 20px;
  top: 20px;
}
```



Absolute positioning

- Takes the element out of the flow of the document, thus taking up no space
- Other elements in the normal flow of the document will act as though the absolutely positioned element was never there



Fixed positioning

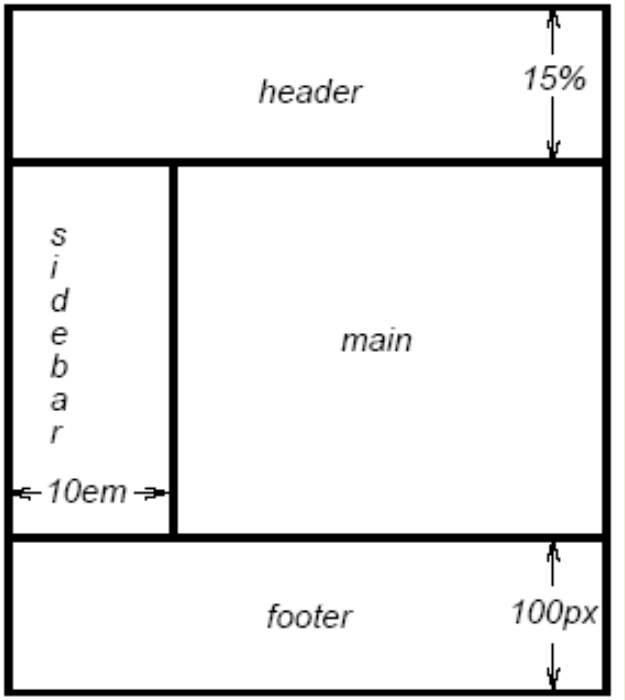
- A subcategory of absolute positioning
  - A fixed element's containing block is the viewport
- Allows to create elements that always stay at the same position in the window

- Note: in case of overlaps the z-index property specifies the stack order of an element (which element should be placed in front of, or behind, the others)

Fixed positioning

- Can be used to create complex frame-like presentations

```
#header { position: fixed; width: 100%;
height: 15%; top: 0; right: 0;
bottom: auto; left: 0; }
#sidebar { position: fixed; width: 10em;
height: auto; top: 15%; right: auto;
bottom: 100px; left: 0;}
#main {position: fixed; width: auto;
height: auto; top: 15%; right: 0;
bottom: 100px; left: 10em; }
#footer {position: fixed; width: 100%;
height: 100px; top: auto; right: 0;
bottom: 0; left: 0; }
```



Examples

- W3Schools.com
  - http://www.w3schools.com/Css/css_positioning.asp

The main problem people have with positioning is remembering which type of positioning is which. Relative positioning is "relative" to the element's initial position in the flow of the document, whereas absolute positioning is "relative" to nearest positioned ancestor or, if one doesn't exist, the initial container block.

A. Budd, C. Moll, S. Collison,
"CSS Mastery: Advanced Web Standards Solutions", FriendsOfED, 2006

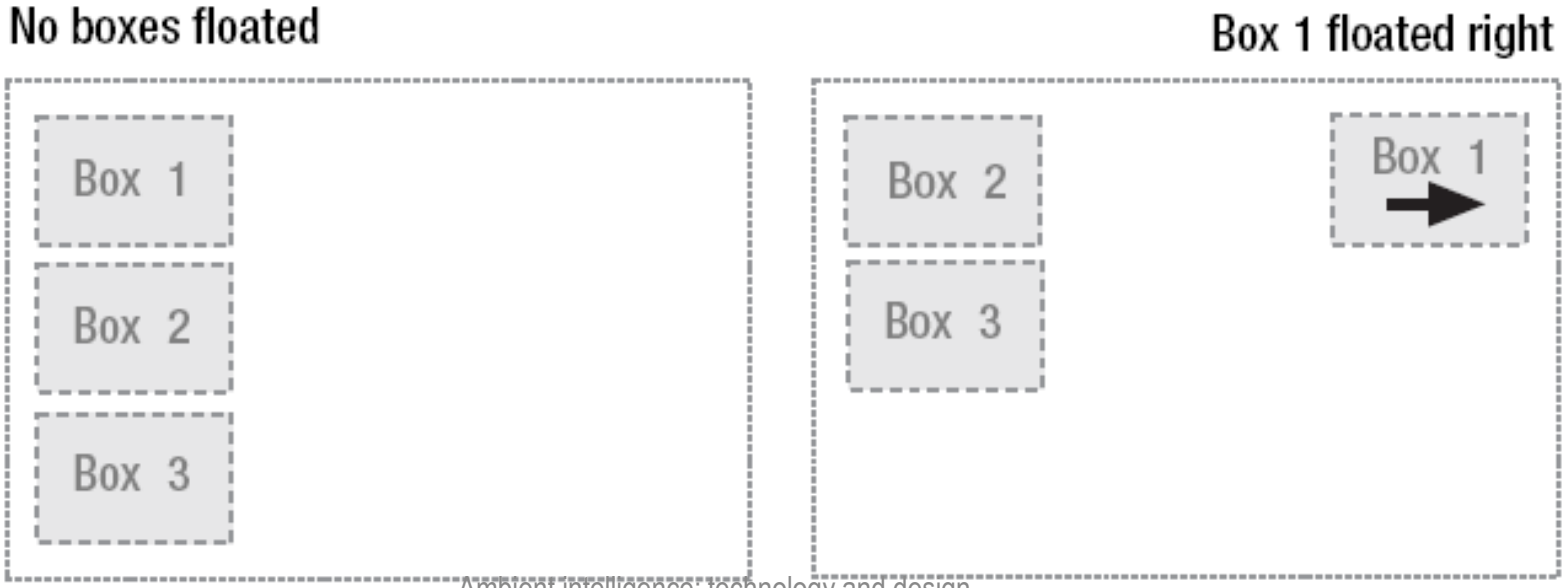
Floating

- 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
- Often used for images and when working with layouts
  - Example
  - http://www.w3schools.com/Css/css_float.asp

```
img
{
float:right;
}
```

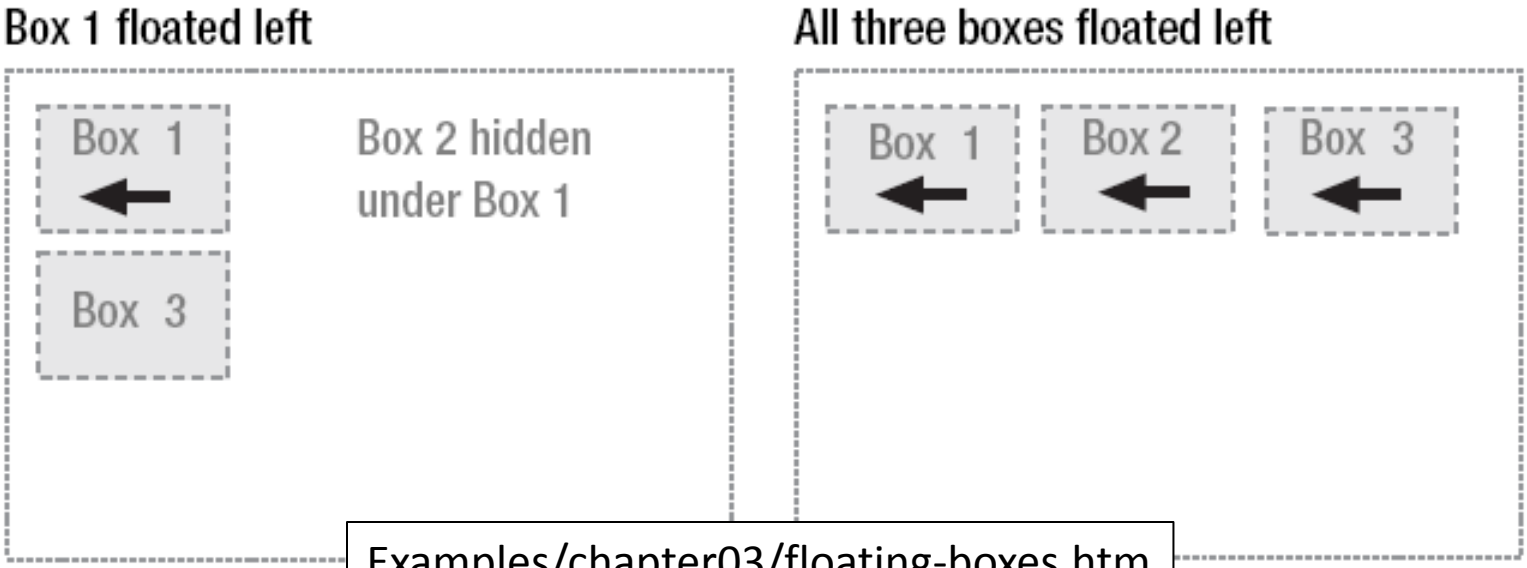
Floating

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



Floating

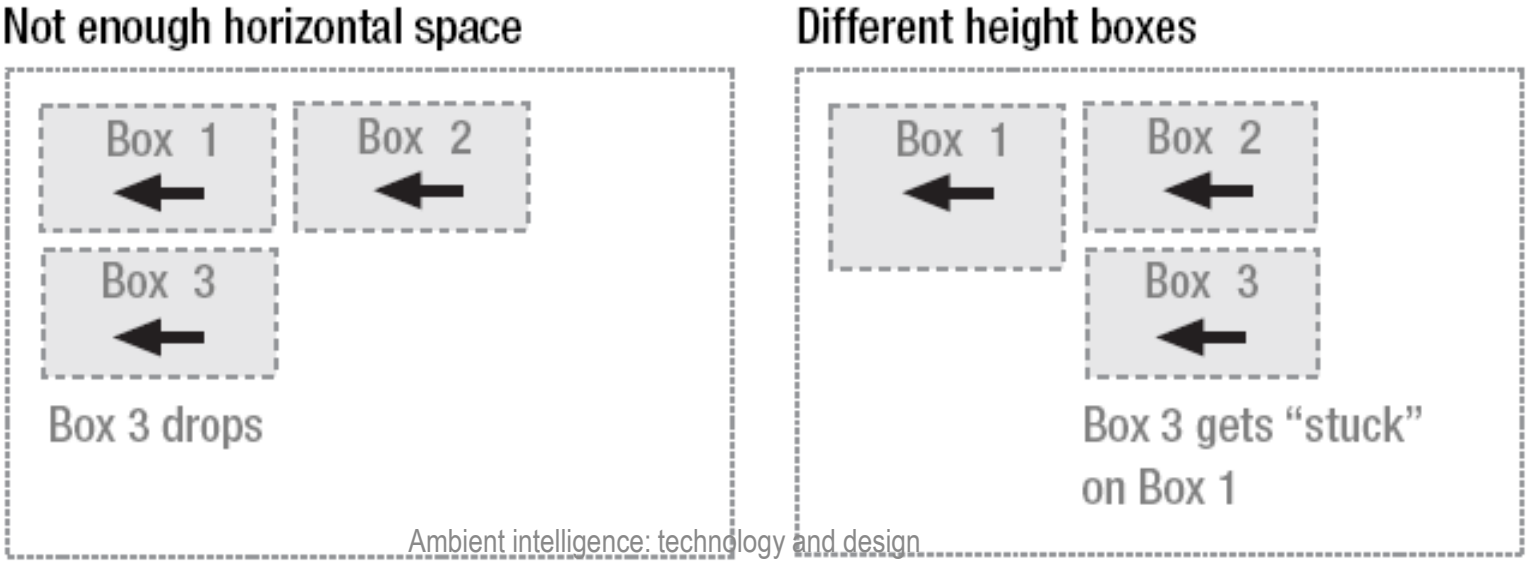
- If all three boxes are floated left
  - Box 1 is shifted left until it touches its containing box
  - Other two boxes are shifted left until they touch the preceding floated box



Examples/chapter03/floating-boxes.htm

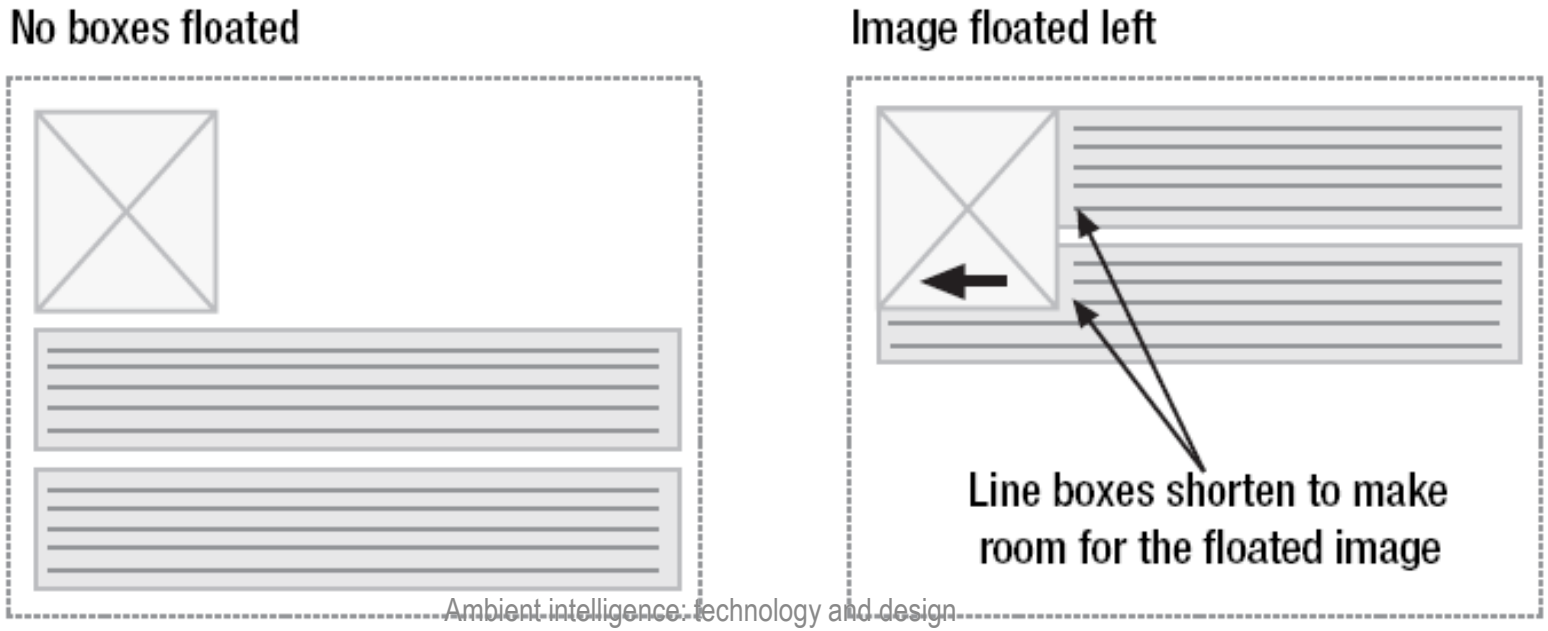
Floating

- 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
  - If the floated elements have different heights, it is possible for floats to get “stuck” on other



Line boxes and clearing

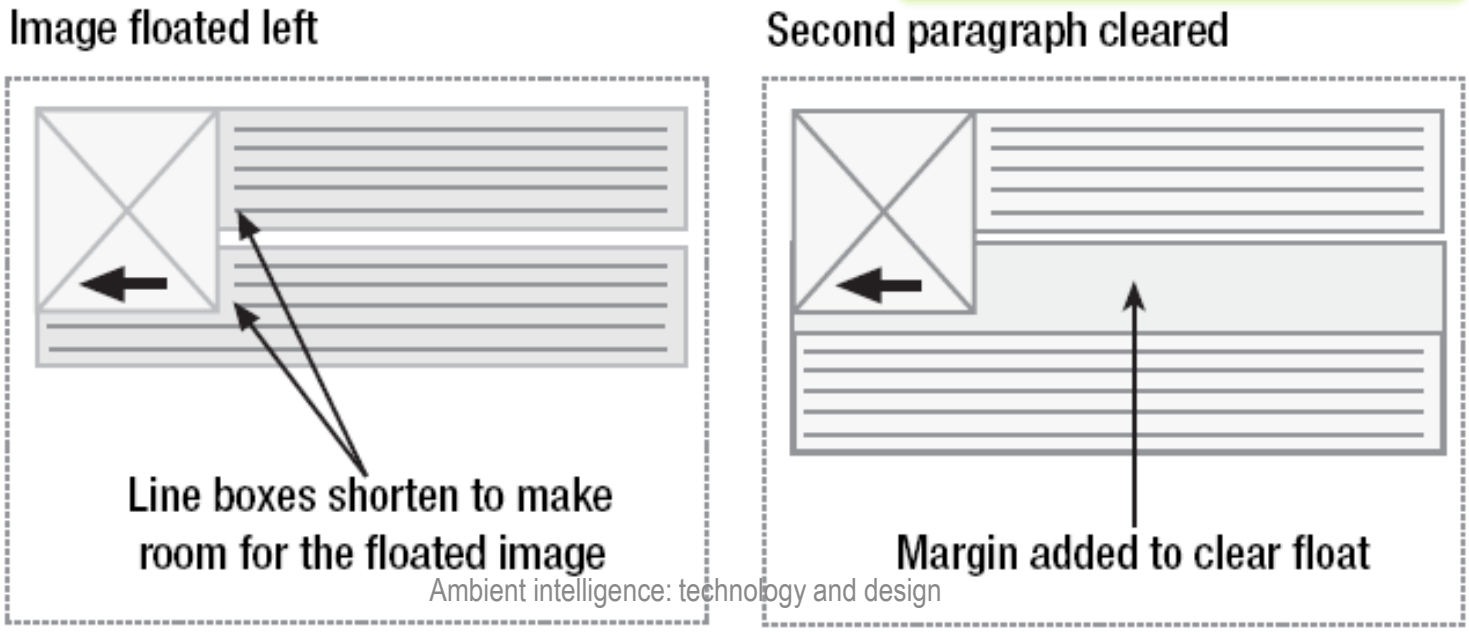
- Line boxes next to a floated box are shortened to make room for the floated box, and flow around the float
  - Floats were created to allow text to flow around images



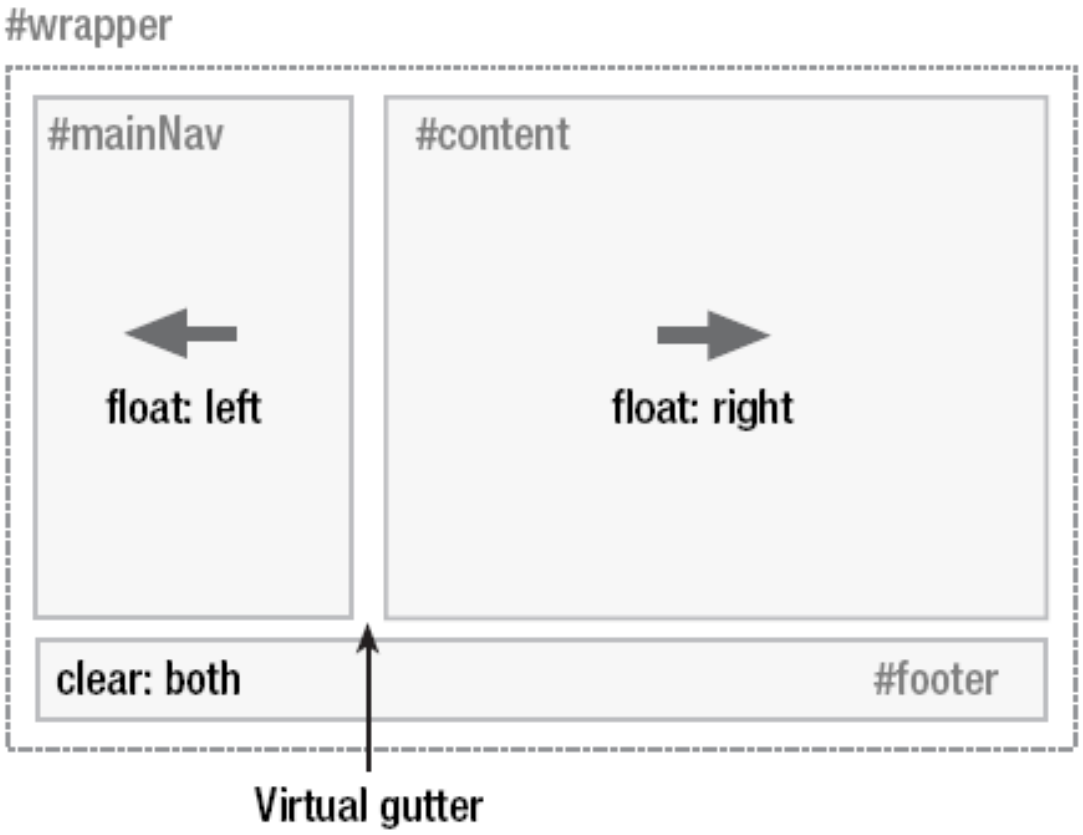
Line boxes and clearing

- 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

```
p { clear: left }
```



Two-column floated layout



Two-column floated layout

- Be al padding

```
#content {
width: 520px;
float: right;
}
#mainNav {
width: 180px;
float: left;
}
#footer {
clear: both;
}
```

```
#mainNav {
padding-top: 20px;
padding-bottom: 20px;
}
#mainNav li {
padding-left: 20px;
padding-right: 20px;
}
#content h1, #content h2,
     #content p {
padding-right: 20px;
}
```

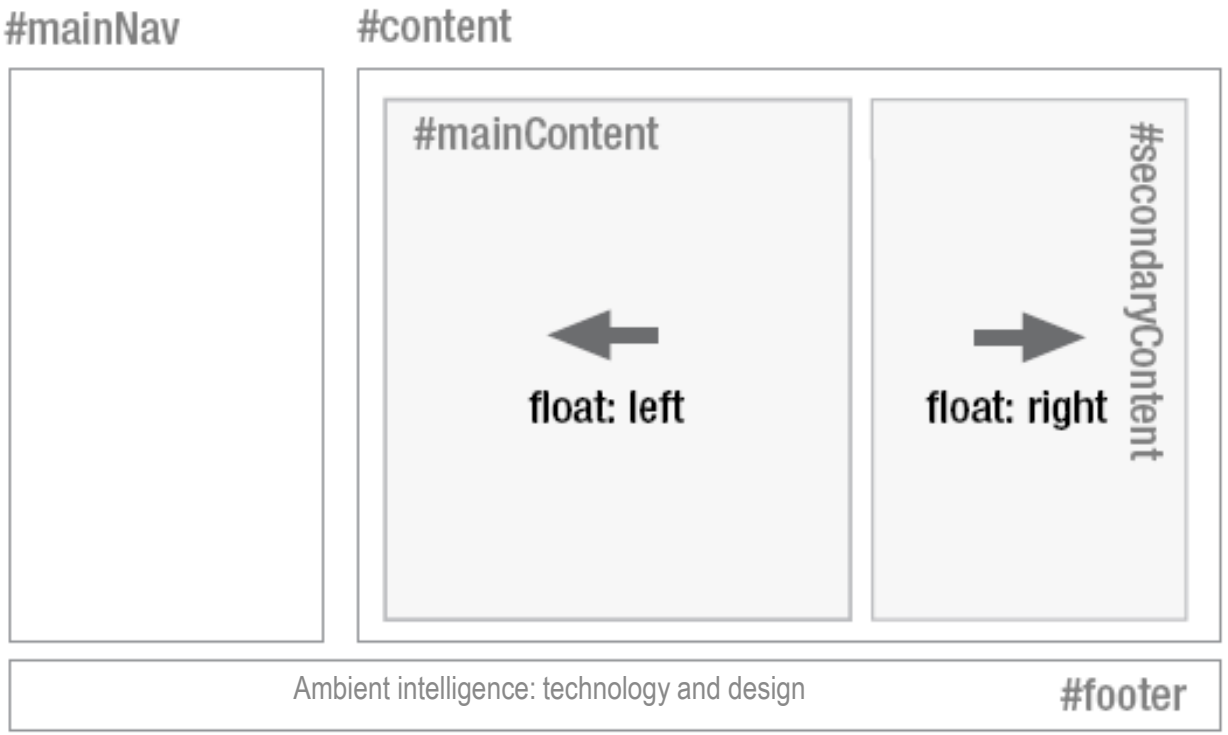
Three-column floated layout

- (X)HTML framework
  - similar to the two column layout, but two new divs inside the content div

```
<div id="content">
<div id="mainContent">
…
</div>
<div id="secondaryContent">
…
</div>
</div>
```

Three-column floated layout

- Float the main content left and the secondary content right, inside the already floated content div
  - Divides the second content column in two, creating a three-column effect



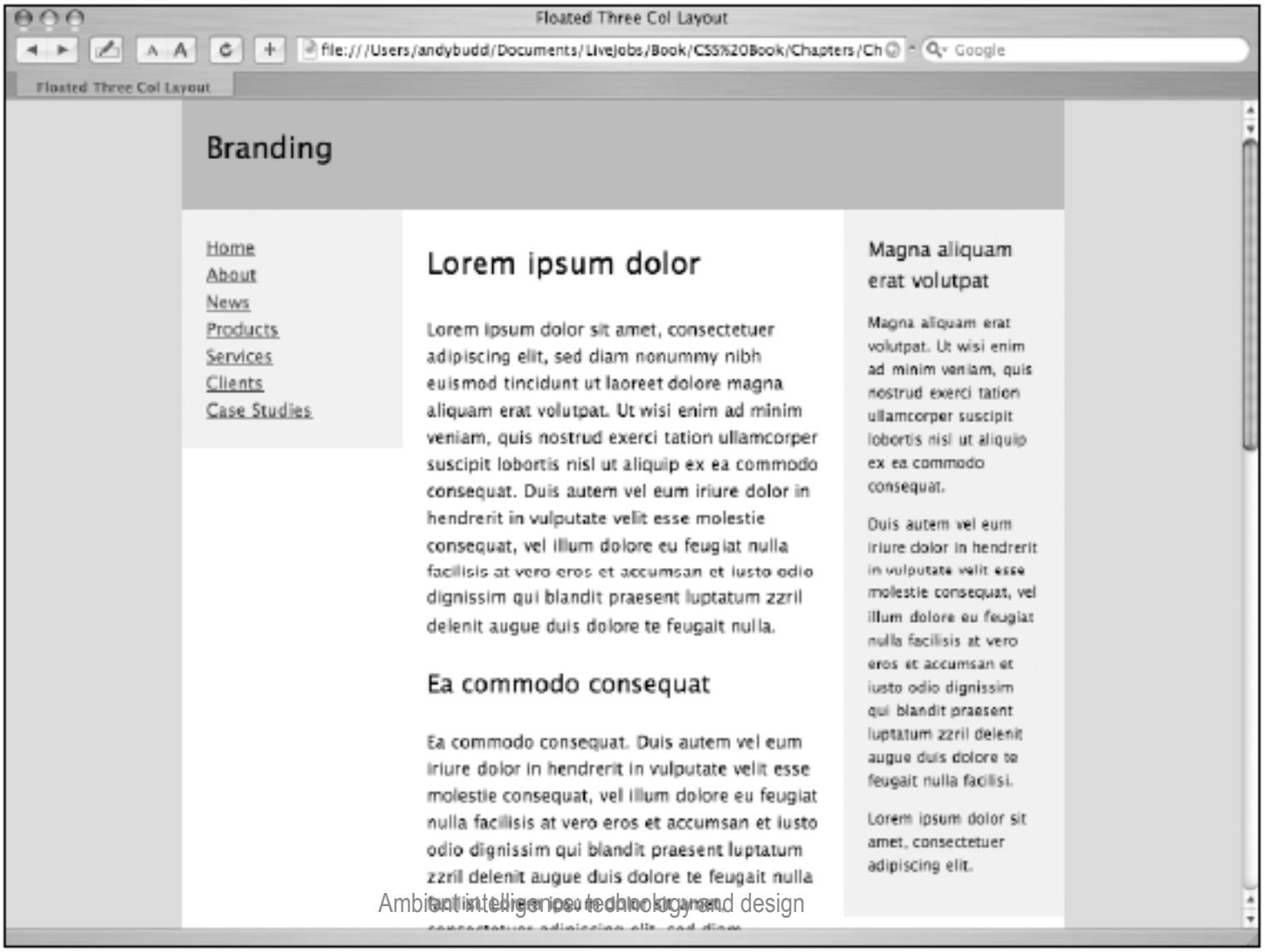
Three-column floated layout

- Better: remove the padding from the content elements and apply it to the content of the secondary columns

```
#mainContent {
width: 320px;
float: left;
}
#secondaryContent {
width: 180px;
float: right;
}
```

```
#secondaryContent h1, #secondaryContent h2,
     #secondaryContent p {
padding-left: 20px;
padding-right: 20px;
}
```

Three-column floated layout



References

- Andy Budd, Cameron Moll, Simon Collison, “CSS Mastery, Advanced Web Standards Solutions”
  - www.cssmastery.com/

Style & Layout in the web

THE BOOTSTRAP FRAMEWORK

Bootstrap

- Open Source CSS (and Javascript) framework
- Allows applying “modern” styles
  - Sensible and nice-looking defaults
  - Easy to apply custom themes
- Takes care of cross-browser issues
- Simplified layout model

- Developed by Twitter
  - http://getbootstrap.com/

Bootstrap philosophy

- Based on CCS classes
- Each class applies one “effect”
- Various classes may be combined in the same element
- Wide range of “standard” classes
- Wide range of additional “components”
  - Ready-to use interactive elements or groups of elements
- Mobile-first
  - Responsive

Using Bootstrap

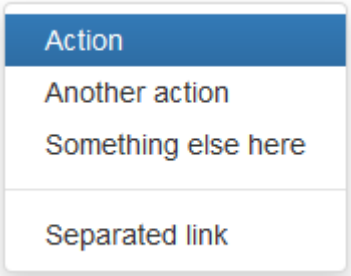
```
<!-- Latest compiled and minified CSS -->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4
/css/bootstrap.min.css">

<!-- Optional theme -->
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/css/bootstrap-
theme.min.css">

<!-- Latest compiled and minified JavaScript -->
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/js/bootstrap.min.js"></script>
```

- 3 files
  - Bootstrap
  - Theme
  - Javascript support
- Or download locally

Page structure

- Start with **Basic template**
  - http://getbootstrap.com/getting-started/#template
- Or, choose from published **Examples**
  - http://getbootstrap.com/getting-started/#examples

Buttons & Menus



Tables

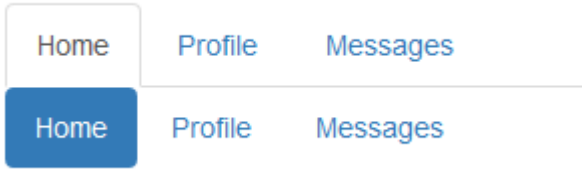
#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

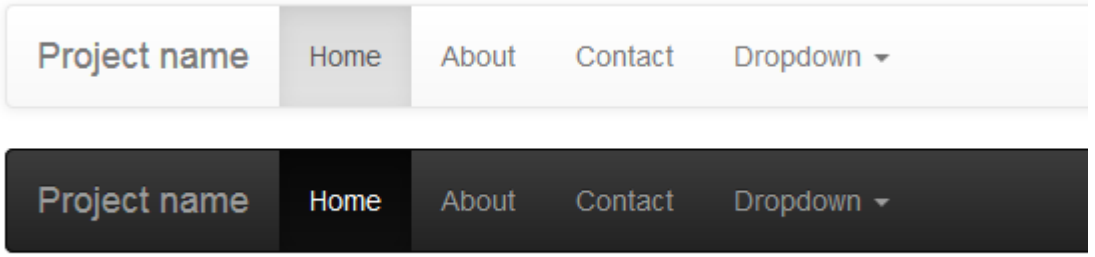
#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
	Mark	Otto	@TwBootstrap
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

Navigation bars





Grids & Columns

License

- These slides are distributed under a Creative Commons license “Attribution – NonCommercial – ShareAlike (CC BY-NC-SA) 3.0”
- You are free to:
  - Share — copy and redistribute the material in any medium or format
  - Adapt — remix, transform, and build upon the material
  - The licensor cannot revoke these freedoms as long as you follow the license terms.
- Under the following terms:
  - Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.
  - NonCommercial — You may not use the material for commercial purposes.
  - ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
  - No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.
- http://creativecommons.org/licenses/by-nc-sa/3.0/