

Education SERVICES

Modern CSS Techniques



TEKsystems Education Services

presents

Modern CSS Techniques

Copyright

This subject matter contained herein is covered by a copyright owned by:

Copyright © 2015 Robert Gance, LLC

This document contains information that may be proprietary. The contents of this document may not be duplicated by any means without the written permission of TEKsystems.

TEKsystems, Inc. is an Allegis Group, Inc. company. Certain names, products, and services listed in this document are trademarks, registered trademarks, or service marks of their respective companies.

All rights reserved

20750 Civic Center Drive
Suite 400, Oakland Commons II
Southfield, MI 48076
800.294.9360

COURSE CODE IN1403 / 01.20.2015



©2015 Robert Gance, LLC

ALL RIGHTS RESERVED

This course covers Modern CSS Techniques

No part of this manual may be copied, photocopied, or reproduced in any form or by any means without permission in writing from the author—Robert Gance, LLC, all other trademarks, service marks, products or services are trademarks or registered trademarks of their respective holders.

This course and all materials supplied to the student are designed to familiarize the student with the operation of the software programs. THERE ARE NO WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, MADE WITH RESPECT TO THESE MATERIALS OR ANY OTHER INFORMATION PROVIDED TO THE STUDENT. ANY SIMILARITIES BETWEEN FICTITIOUS COMPANIES, THEIR DOMAIN NAMES, OR PERSONS WITH REAL COMPANIES OR PERSONS IS PURELY COINCIDENTAL AND IS NOT INTENDED TO PROMOTE, ENDORSE, OR REFER TO SUCH EXISTING COMPANIES OR PERSONS.

This version updated: 1/20/2015

Notes

Chapters at a Glance

Chapter 1	The Web Development Environment	9
Chapter 2	CSS Foundations	35
Chapter 3	CSS Box Model	70
Chapter 4	CSS 3 Features	107
Chapter 5	Responsive Solutions	126
Chapter 6	Responsive Typography and Images	162
Chapter 7	Fluid Layouts and Grid Systems	176
Chapter 8	CSS Pre-processing Systems	189
	Course Summary	206

Notes

Modern CSS Techniques



Course Objectives

Understand **CSS language fundamentals** including idioms and best practices

Use **CSS layouts** to design any page structure

Create and work with **responsive solutions**

Understand the uses of a **CSS pre-processing system**

Utilize techniques for **Progressive Enhancement**



Course Agenda

Day 1

- Web Development Environment
- CSS Foundations
- CSS Box Model



Course Agenda

Day 2

- Page Layout
- CSS3 Features and Progressive Enhancement
- Responsive Design Principles
- Media Queries



Course Agenda

Day 3

- Fluid Grids
- Bootstrap
- Typography & Responsive Images
- Using Pre-processing Systems



Introductions

Name 

Where you're from 

What you work on 

Reason for attending 





Logistics



Typical Daily Schedule*

8:30	Start Day
9:45	Morning Break
11:30 – 12:30	Lunch
1:45	Afternoon Break 1
3:00	Afternoon Break 2
4:30	End of Day

** Your schedule may vary*



Get the Most from Your Experience

Ask Questions



Chapter 1

The Web Development Environment

Web Development Considerations
and Practices

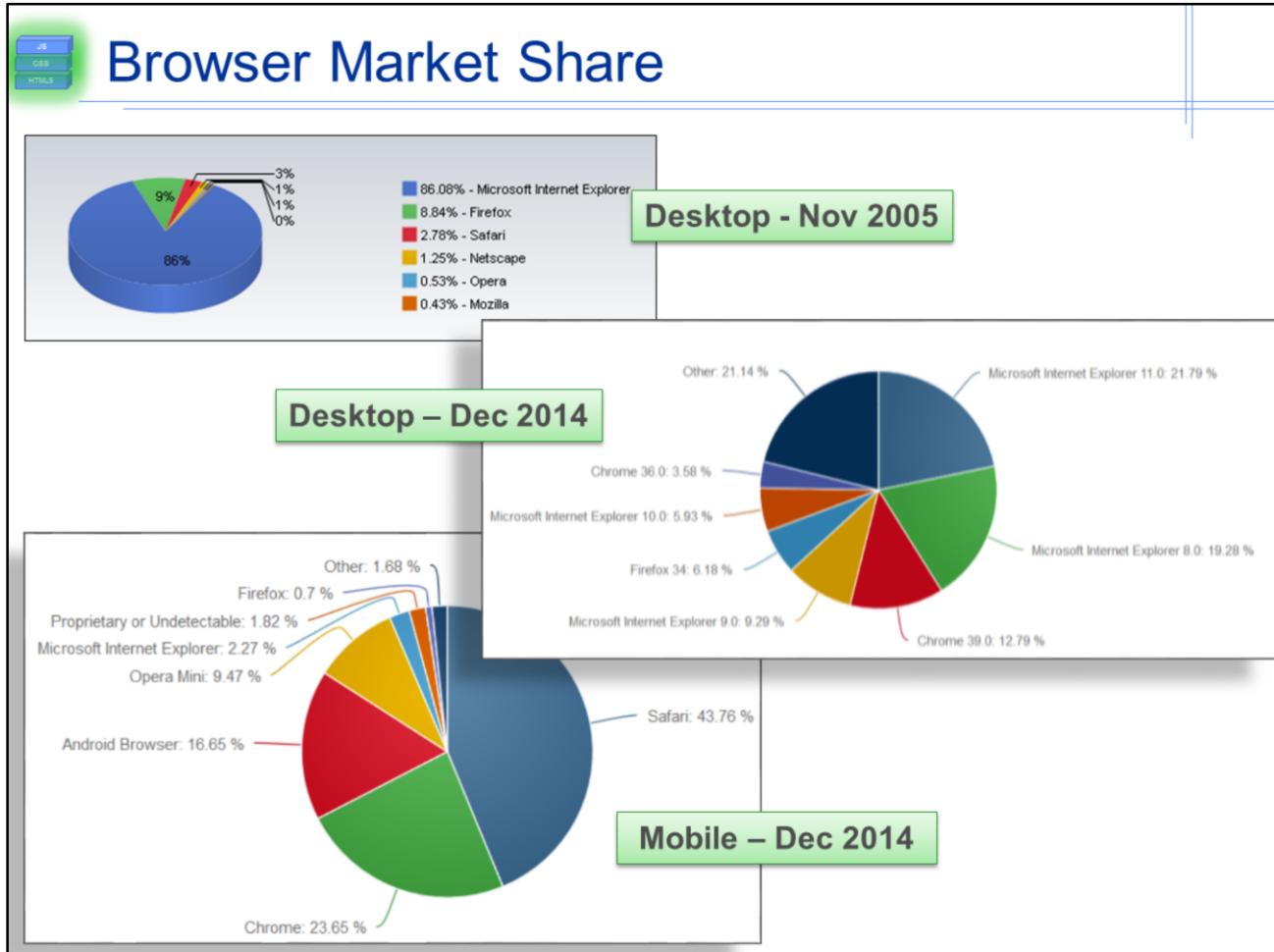


Overview

Introduction

Architecture

Browser Tools



Source: marketshare.hitslink.com

Note the changes in desktop browser market share over the last 6 years. No longer should solutions be written for single-browsers. Even in situations where applications are targeted for internal clients whose browsers are strictly controlled, careful consideration should be given to targeting that single browser.



A Layered Architecture



Layered architectures
address the issues
encountered when
applications become large
and complex

An architecture makes solutions more manageable when developing complex applications - architecture on the client side is analogous to trying to provide structure on the server-side.



Why a Layered Architecture?

- Attempts to **support all user-agents** (browsers)
 - Uses a universally understood markup document
 - Layers additional stylesheets and scripts as supported by the browser
- Allows **modern devices** to take advantage of newer features
- Provides structure for **future design strategies**

Layered architectures guard against changes to future end-user applications while at the same time attempt to work in the widest possible user-agents.



Presentation Layer: CSS

- **Cascading Style Sheets** define the look-and-feel of an application
 - Separation of content and presentation is instrumental in ensuring responsive, maintainable designs

<http://www.csszengarden.com/>

The screenshot shows the CSS Zen Garden homepage. On the left, there's a sidebar with navigation links like 'css Zen Garden', 'The Beauty of CSS Design', 'The Road to Enlightenment', 'Participation', 'Benefits', and 'Help'. The main content area displays a grid of 12 different website designs, each with a title and a brief description. Two specific versions are highlighted with green boxes and arrows pointing to them: the 'Default CSS' version (a simple dark-themed site) and the 'Under the Sea' theme (a site with a nautical theme featuring sea creatures and a ship). The 'Under the Sea' theme box also contains the text 'CSS Zen Garden Verde Moderna Design'.

The presentation layer is controlled by CSS. By developing solutions where the presentation is entirely managed through CSS, a flexible, responsive layout can be created. In order to do this, it is vital to understand how to manipulate HTML elements (represented as DOM nodes within a browser) to generate a very specific, controlled (even dynamic) layout.

CSSZenGarden is a great example of a layered architecture.



Behavioral Layer: JavaScript

- Enterprise JavaScript source should be
 - unobtrusive
 - standards-based
 - cross-browser/platform/device capable
 - focused for team development
 - modularized

A **JavaScript Library**
(jQuery, Dojo, YUI, ExtJS, Prototype, ...)

A **Dependency Management Library**
(requirejs, StealJS, LabJS, ...)

An **MV* Framework** (Backbone, JVMC, AngularJS, Knockout, Sammy, ...)

A **Templating System** (Handlebars, Mustache, Underscore Templates, EJS, jquery-tmpl, ...)

Inter-module Communication frameworks
(jquery-pubsub)

Testing Environments
(FuncUnit, qUnit, Selenium, JSUnit, Jasmine, SinonJS)

Concatenation, Minification and Build Systems
Tools (Bootstrap, steal.build, steal.clean)

Enterprise JavaScript is so named because it involves the use of frameworks that may not normally come up in smaller development environments. Larger team-based (enterprise) development typically have slightly more complex requirements for application development, testing, the build process, and the development environments.



Structural Content Layer: Semantic HTML

- At the heart of a Layered Architecture is *semantic HTML*
 - “Relating to meaning”
 - In practice, using tags that describe the content they contain

Don't confuse Semantic HTML with XHTML, they aren't the same

- Examples:
 - `<h1>` for most important text,
 - `<div>` for content divisions,
 - `<p>` for blocks of text,
 - `<table>` for data only

```
<table>
<tr id="head"><td>...</td></tr>
<tr id="body"><td>...</td></tr>
<tr id="footer"><td>...</td></tr>
</table>

<div>
<div class="hd">...</div>
<div class="bd">...</div>
<div class="ft">...</div>
</div>
```



Semantic HTML requires using HTML tags, not for presentation, but to provide as much semantic meaning as possible to the content. This is often difficult because HTML is a limited tag set, however, we can achieve some relative success in semantically marking up our documents. Don't confuse semantic HTML with XHTML. XHTML requires HTML to follow the syntactic rules of XML, it does not require semantic markup. It's possible to use XHTML and still have a poorly written document.

Using semantic HTML implies we will be using HTML elements to achieve maximum benefit for the content's purpose, not for the presentation's purpose.



Why Semantic HTML?

- Use of semantic HTML improves:
 - SEO rankings
 - Ease of maintenance (no tag soup)
 - Page size/load time
 - Accessibility
- Semantic HTML forms the basis of
“Progressive Enhancement”

A semantically-driven approach to HTML increases useful life-span of that HTML.

“structural semantic markup provides lean, meaningful, accessible pages” –
natekoechley



Progressive Enhancement

- *Creating the best possible solution for the widest number of users*
- Approach:
 - Develop a base level solution that puts content at the center of the design
 - Layer enhancements on top by adding presentational and behavioral features

The image contains two side-by-side screenshots of the Yahoo homepage. Both screenshots show a bowl of guacamole with a lime next to it. The top screenshot is labeled "yahoo.com with modern browsers" and has a yellow header bar. The bottom screenshot is labeled "yahoo.com in IE6-8" and has a red header bar. Both screenshots include the same text: "Secret to keeping guacamole green" and "Leaving the pit in only keeps part of the avocado from browning, and using lemon juice doesn't do much, either. ['Genius' trick »](#)". There are also small navigation elements like "1 – 5 of 90" and a search bar.

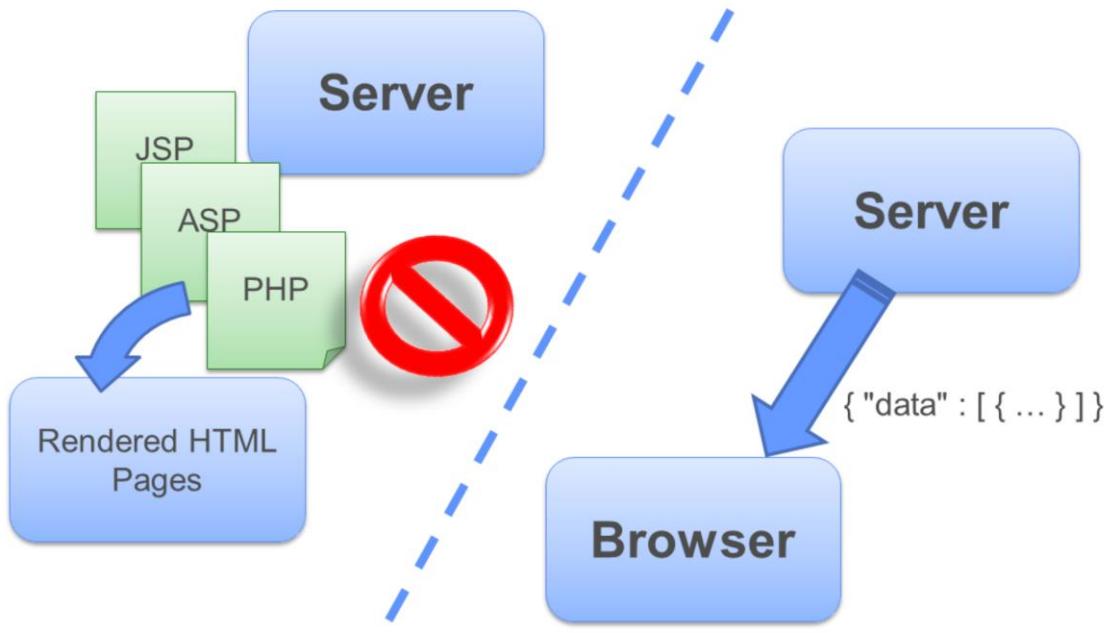
There are numerous examples of progressive enhancement in action. A YUI datatable widget is a good example because it transforms a basic HTML table (seen by all browsers) into a stylized, interactive one for browsers that support it.

Other examples of progressive enhancement include the use of CSS rounded corners, where IE browsers (8 and earlier) would see squared corner solutions while other browsers could see the more visually appealing rounded corner versions (assumes use of CSS rounded corner properties and not use of background image techniques).



Modern Front-End Solutions

- Modern browser-based architectures are heavily **service-driven** rather than page-oriented



Modern day thin-client solutions, especially those that exhibit the single-page, desktop-style application model, heavily emphasize service-oriented architectures. The data exchange format is often via JSON (JavaScript Object Notation) or perhaps JavaScript data or even proprietary data formats. While XML is not commonly used for most browser-based implementations, it can be selected as a data communications format.



Developer Tools

- Developer tools are essential for today's front-end devs:
 - Firebug (3rd Party Firefox plugin)
 - Firefox Developer Tools
 - Chrome Developer Tools
 - IE Developer Tools

While any of these tools may be used to help with development, you should generally **NOT** use IE as your **primary** test tool. It will lead to non-functioning solutions in other browsers

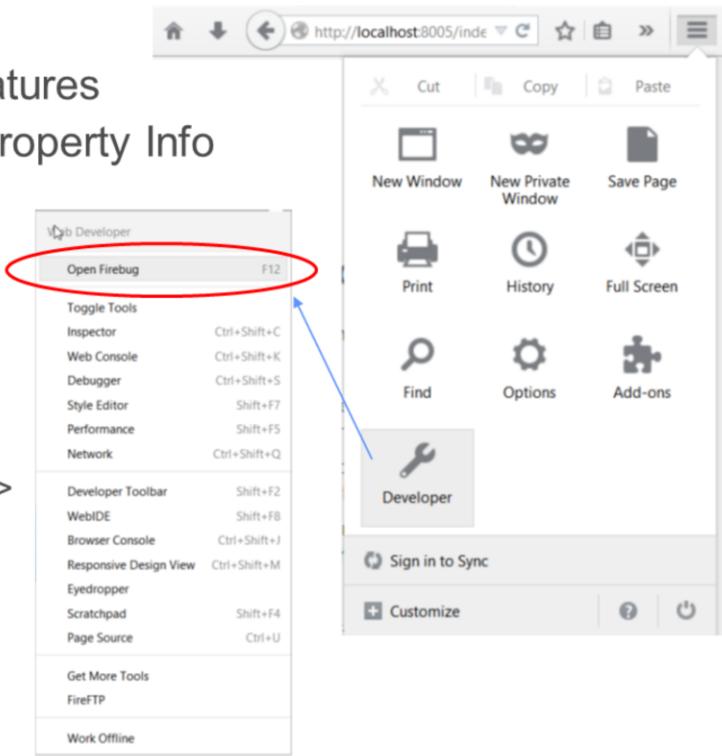
While IE11 has come a long way and features the best level of compatibility to date with the standards, most developers often prefer Chrome due to its speed and accuracy.



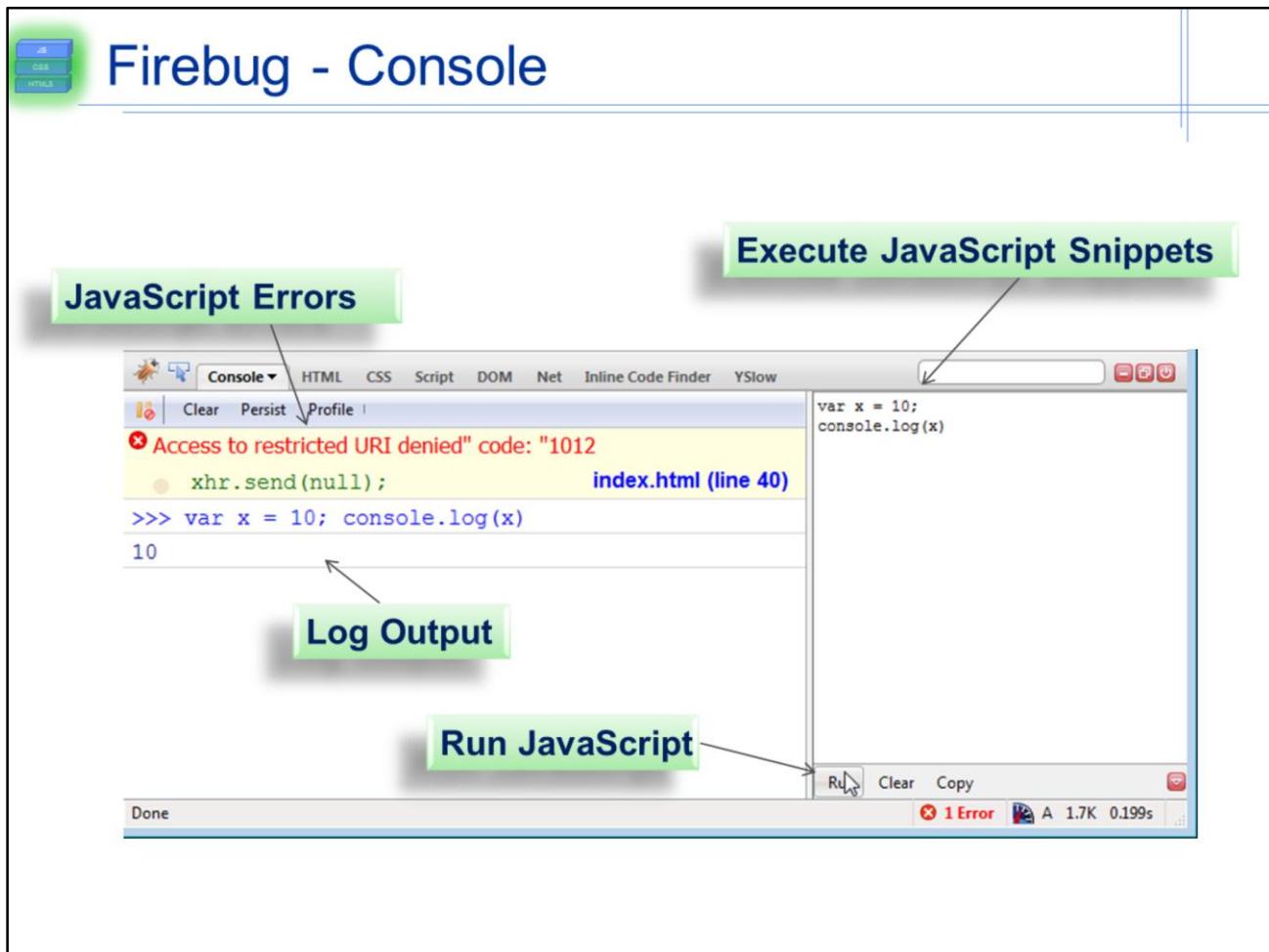
Firebug

Install from
<http://getfirebug.com>

- Firefox plug-in that provides
 - A Debugger
 - DOM Inspection Features
 - Box Model & CSS Property Info
 - Event Notifications
 - Profiling Info
 - XHR Monitoring
- F12 to launch, or Options Menu > Developer > Open Firebug...



Test to see if Firebug is installed by pressing F12.

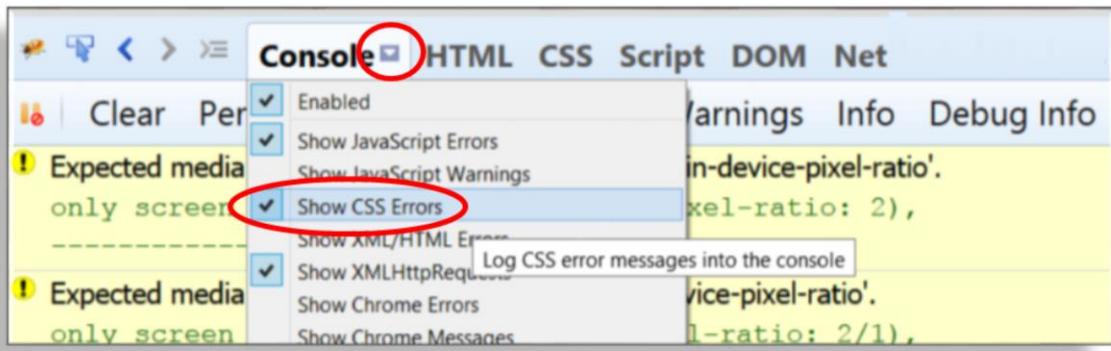


The Firebug console is useful for viewing errors, warnings, and log output. It can also be useful to test JavaScript Snippets.



Firebug - Console (for this class)

- For this class, it might be useful to turn on CSS Errors (off by default) if you use Firebug:



Though these items are referred to as CSS Errors, Firebug reports CSS items that the Mozilla rendering engine did not understand--it doesn't mean that you should remove the error as it may be a perfectly fine statement in other browsers.

Firebug - Using the *HTML* Tab

The screenshot shows the Firebug interface over a Yahoo homepage. The browser menu bar includes Home, Mail, Answers, Groups, Flickr, Tumblr, Games, Live, Screen, Mobile, and More. The main content area displays a news article about Tampering charges filed against the Jets. On the left, a sidebar lists categories like Mail, News, Sports, Finance, Weather (which is selected), Autos, Homes, Dating, Shopping, and others. A blue arrow points from the 'Weather' category in the sidebar to the 'HTML' tab in the Firebug toolbar. The Firebug toolbar also includes Console, HTML (selected), CSS, Script, DOM, and Net. The HTML tab shows the DOM tree for the current page, with the path: Edit > a.ell.fz-s > li > ul.navlist > div#defau..._default > div#defau..._default > div.pr-m > div.stick...t-holder. A blue callout box with the text 'Selecting the 'Inspector Tool' allows for quickly locating the HTML in the source' is positioned near the toolbar.

Selecting the 'Inspector Tool' allows for quickly locating the HTML in the source

Firebug - HTML Tab - Style Information

The screenshot shows the Firebug interface over a Yahoo! homepage. In the Firebug toolbar at the top, the 'HTML' tab is selected. Below the toolbar, the main content area displays the Yahoo! homepage with various news items and navigation links. A specific element, a list item containing a link to a news article, is selected in the HTML source code. A red circle highlights the 'Style' tab in the Firebug toolbar. A callout box points from this 'Style' tab to a tooltip containing the following text:

Selecting an HTML element in the source, Provides the CSS for that element only

The tooltip also points to the 'Style' tab in the Firebug toolbar.

Once an element is selected, you may view the CSS related to this element in the Style tab (as shown).

Firebug - HTML Tab - Layout (Box Model)

The screenshot shows the Firebug interface over a Yahoo homepage. The Layout tab is highlighted with a red circle. On the right, a visual representation of the element's box model is shown, indicating dimensions of 128 x 31 pixels. A blue callout box contains the text: "The Layout Tab provides a visual indication of the element's properties".

The Layout Tab provides a visual indication of the element's properties

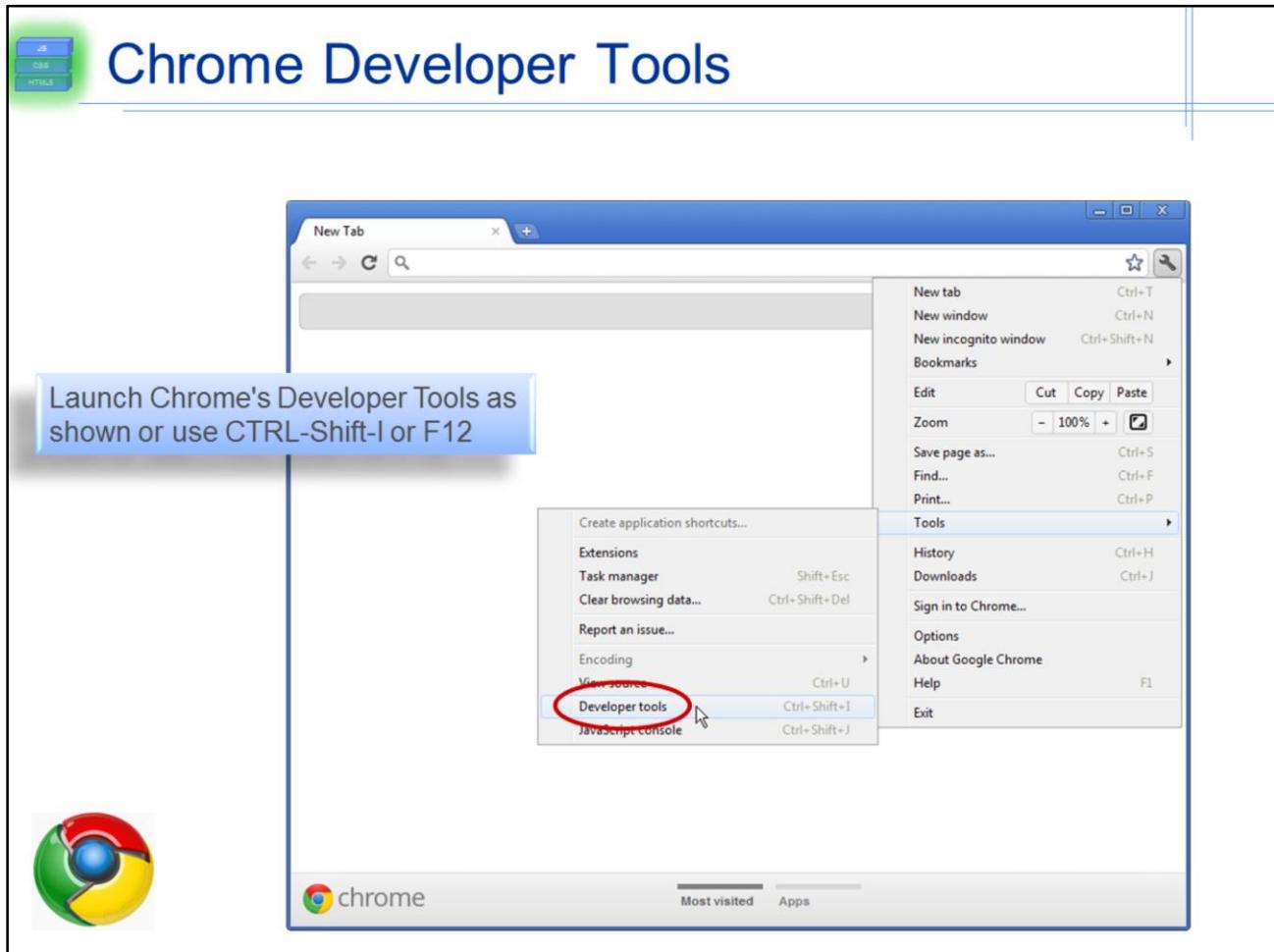
Once an element is selected, you may also get a visual cue of the element's properties using the Layout Tab.

Firefox Developer Tools

- Native browser built-in tools
(doesn't require a 3rd party install)

The screenshot shows the Firefox browser window with the developer tools open. A large blue arrow points from the text "Native browser built-in tools" towards the developer toolbar. The developer toolbar is a floating panel on the right side of the screen, containing icons for various tools like Firebug, Inspector, and Network. A red circle highlights the "Toggle Tools" option in the toolbar's context menu. The main Firefox interface shows a URL bar with "http://localhost:8005/index.html" and a developer console at the bottom displaying some log output.

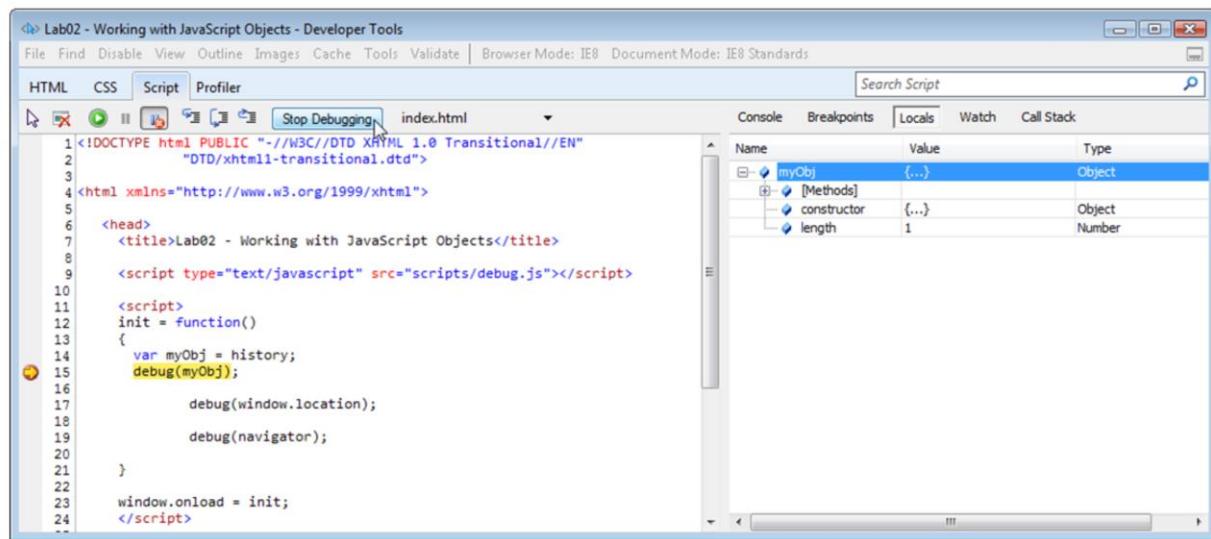
Firefox has a set of built in developer tools in lieu of Firebug if desired. These tools ship with Firefox and do not have to be installed. (CTRL-SHIFT-K).



The screenshot shows the Yahoo homepage within the Chrome Developer Tools Elements tab. A red circle highlights the magnifying glass icon in the toolbar, which is used to select elements. A blue arrow points from this icon to a callout box containing the text: "Use the 'Inspector Tool' to select elements on a page, then view the corresponding HTML source below". Another blue arrow points from the highlighted element in the DOM tree to another callout box containing the text: "As with Firebug's HTML Tab, Chrome's Elements Tab serves a similar purpose". The DOM tree on the left shows the structure of the page, including the navigation bar and various news items. The right panel displays the selected element's styles, including the CSS rule: `li { list-style-type: none; }`.

IE 8-10 Developer Tools

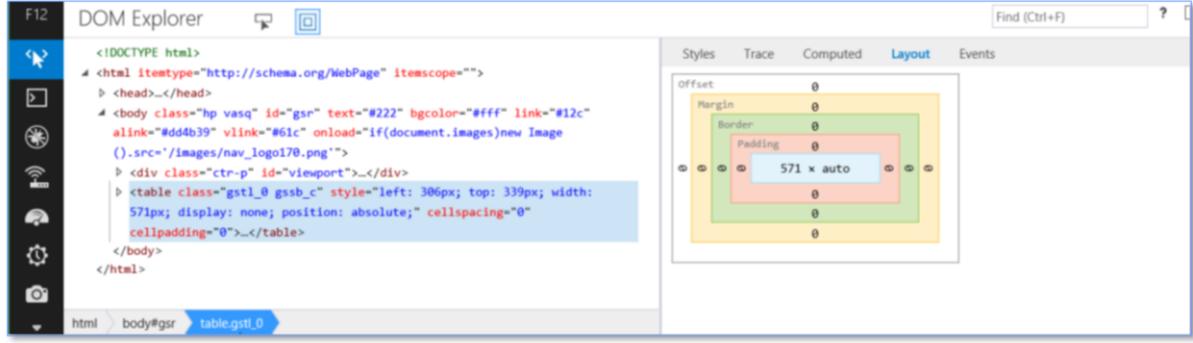
- IE 8+ provides a built-in debugger and DOM inspector also
 - Much more crude and limited in capabilities



To open the developer tools, use an F12.

IE11 Developer Tools

- Use IE11 has improved its developer tools
 - Similar to Firebug in many ways



The screenshot shows the IE11 Developer Tools interface. On the left is the DOM Explorer panel, which displays the HTML structure of a page. A specific table element is selected, highlighted with a blue background. The right side of the interface is the Layout tab, which provides a visual representation of the selected element's dimensions and padding. The Layout tab includes a tree view showing the element's offset, margin, border, and padding. The padding for the table is explicitly shown as "571 x auto". The status bar at the bottom indicates the path: html > body#gsr > table.gstl_0.

IE11 now provides developer tools on par with Firebug/Chrome Dev Tools. Selecting Browser/Document Modes can be done on the Emulation tab. (Bottom icon on the left).



Frontend IDE Options

- Brackets
 - brackets.io
- JetBrains WebStorm
(commercial license)
 - www.jetbrains.com/webstorm/
- Sublime Text
 - SublimeText.com
- Komodo IDE
 - www.komodoide.com
- IntelliJIDEA (commercial license)
 - www.jetbrains.com/idea/



Brackets



WebStorm

Sublime Text



Komodo IDE
by ActiveState



IntelliJIDEA



Cloud9

Spket IDE

aptana®



NetBeans



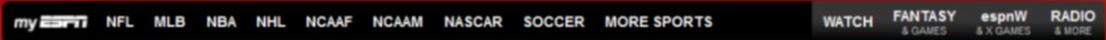


Exercise 1 - It's Your Turn

Explore ESPN.com using Chrome Developer Tools or Firebug

Perform Exercise 1

Using the Browser Developer Tools



Locate step-by-step instructions for this lab in the back of your manual



Summary

- Modern frontend development should combine several technologies (JavaScript, CSS, HTML) into a *layered architecture*
- Consideration for different browsing environments should be designed at the beginning of the development process
- Use newer tools and IDEs to facilitate the development process

Chapter 2

CSS Foundations

Rules of the Presentation Layer



Overview

Basic Syntax

Rules of Specificity

Browser Normalization



Modern CSS

- **Cascading Style Sheets** provide the *presentation* content to a layered semantic architecture
- Modern CSS solutions may employ
 - **Layout Systems** (Bootstrap, YUI Grids, Foundation, Skeleton, etc.)
 - **Pre-processing Tools** (LESS, SASS, etc.)
 - **Responsive Designs**
 - **Progressive Enhancement Features**



Brief History

- First proposed in 1994 by Hakon Wium Lie
- Standard maintained by **W3C**
- **CSS 1** standard in **1996**
- **CSS 2** Recommendation status in **1998**
- **CSS 2.1** fixed inconsistencies in CSS2
 - Reached Recommendation status in **2011**
- **CSS 3** is broken in 50+ modules all at various stages of standardization and adoption



How CSS "Hooks" into HTML

- CSS uses selectors to bind to HTML elements
 - This is accomplished in several ways:

ID selectors: #quote1

```
<p id="quote1">No great thing is created suddenly</p>
<p class="author">Epictetus</p>
```

**Class
selectors:
.author**

```
<div id="quote2">
  <p class="proverb">Inches make champions
    <cite>Vince Lombardi</cite>
  </p>
</div>
```

HTML selectors: div cite

**Combinations of these selectors:
#quote2 .proverb cite**



CSS Rule Placement

1. **Inline** (within the HTML tag)

```
<p style="padding: 5px">This is...</p>
```

Don't use this approach as it violates a layered architecture



CSS Rule Placement

2. Embedded

```
<html>
  <head>
    <style type="text/css">
      p { padding: 5px; }
    </style>
  </head>
  <body>
    <p>This is...</p>
```



CSS Rule Placement

3. External (using link tag)

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

absolute or relative path

Relationship between external document and the HTML source

content type

External CSS files are cached by browser – which can help with performance when you reuse the same CSS rules on multiple pages



Common CSS Properties

Text

Formatting

color
letter-spacing
text-align
text-decoration
text-indent
text-transform
line-height

Font

Related

font-size
font-family
font-weight
font-style
font-variant
font
white-space

Color and

Background

background-color
background-image
background-position
background-repeat
background-attachment
background-color

Box Model

width
height
border
margin
padding
overflow
float
clear
position
top, left,
bottom, right
display
z-index
visibility

Lists

list-style-image
list-style-position
list-style-type
list-style

Links

a:link, a:visited, a:hover,
a:active



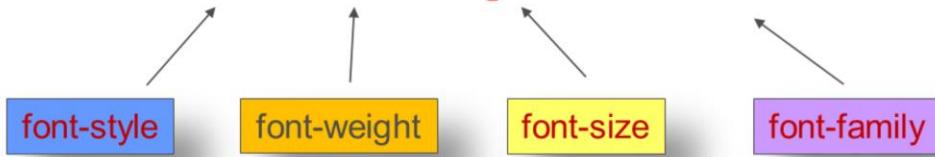
Font Properties

- Some common font properties:

```
font-family: [family name/generic family]  
font-style: [normal|italic|oblique]  
font-size: [keyword or unit]  
font-weight: [keyword]  
font-variant: [normal|small-caps]
```

- Can be combined into shorthand:

```
font: italic bold 12px arial sans-serif;
```



Not recommended to use font shorthand: if you don't specify the font-weight, font-style, or font-variant then these values will automatically default to a value of "normal" (which may override any normally inherited values!)



Text Properties

- Some common text properties:

```
letter-spacing:    [normal|+ or - value]
word-spacing:     [normal|+ or - value]
text-indent:      [+ or - length or % value]
text-align:        [left|right|center|justify]
text-decoration:  [none|line-through|underline]
text-transform:   [lowercase|uppercase|capitalize]
white-space:     [normal|nowrap|pre]
vertical-align:  [super|top|middle|bottom|sub]
line-height:      [normal|any unit]
```

Text decoration can have multiple values at the same time

“pre” will retain preformatted text white-spacing

“line-height” defines the spacing between the lines (eg, single space, double space etc...)



Using Shorthands

The following three rules express the same thing:

```
body {  
    margin-top: 5px;  
    margin-right: 5px;  
    margin-bottom: 5px;  
    margin-left: 5px;  
}
```

```
body { margin: 5px 5px 5px 5px; }
```

```
body { margin: 5px; }
```

The order of the middle example is “top, right, bottom, left” (clockwise from top).



Shorthands

- Set or Skip

```
body {  
    padding-right: 5px;  
    padding-bottom: 5px;  
}
```

Browser's default (or
inherited value) will
apply to missing styles

- Symmetry Shorthand

```
body { padding: 5px 5px; }
```

– top/bottom, left/right

- Side to Side Symmetry

```
body { margin: 5px 5px 5px; }
```

– top, right/left, bottom



Shorthand - Properties

```
div {  
    border-width: 5px;  
    border-style: solid;  
    border-color: #CCC;  
}
```

```
div {  
    border-top-width: 5px;  
    border-right-width: 5px;  
    border-bottom-width: 5px;  
    border-left-width: 5px;  
    border-top-style: solid;  
    border-right-style: solid;  
    border-bottom-style: solid;  
    border-left-style: solid;  
    border-top-color: #CCC;  
    border-right-color: #CCC;  
    border-bottom-color: #CCC;  
    border-left-color: #CCC;  
}
```

```
div { border: 5px solid #CCC; }
```



Order of values in the shorthand version can make a difference in some older browsers



HTML (Type) Selector

```
p { margin: 5px }
```

```
<body>
  <div>
    <p>Lorem ipsum dolor...</p>
  </div>

<p class="intro">This is a paragraph</p>
```

Targets *all* instances of a particular element



Descendant Selector

```
ul em { text-decoration: underline; }
```

```
<body>
```

targets this

```
<ul>
```

```
    <li><em>first</em> item</li>
    <li>second item</li>
```

```
</ul>
```

```
<p>This is a <em>great</em> paragraph</p>
```

not this

Targets any element that is a descendent



Universal Selector

```
* { margin: 5px; }
```

- Can be combined with other selectors

```
ul * { margin: 5px; }
```

```
div > * { margin: 5px; }
```

Targets *all* elements



Class Selectors

- A form of selector, using the “class” attribute
- Classes can be applied to multiple elements on a page, allowing re-use of declarations
- Uses dot notation within CSS

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

```
<div class="warning"><p>Florida continues to get battered by  
hurricanes.</p></div>
```

```
<p><span class="warning">Beware</span> of the  
hurricanes.</p>
```

Notice we're using the same class “warning” on different elements



Class Selectors

- Can be referenced by a particular type of element

```
.warning { color: red; }  
span.warning { font-weight: bold; }
```

```
<div class="warning"><p>Florida continues to get  
battered by hurricanes.</p></div>
```

```
<p><span class="warning">Beware</span> of the  
hurricanes.</p>
```

span.warning and **span .warning** are different!



Class Selectors

- Elements can declare multiple classes

```
<div class="warning first"><p>Florida continues to  
get battered by hurricanes.</p></div>
```

- UI Libraries use classes extensively

```
<div role="presentation" id="widget_searchInput"  
      class="dijit dijitReset dijitInline dijitLeft dijitTextBox" widgetid="searchInput">  
  
    <div class="dijitReset dijitInputField dijitInputContainer">  
      <input type="text" autocomplete="off" data-dojo-attach-point="textbox,focusNode"  
            class="dijitReset dijitInputInner" tabindex="0" id="searchInput" value="">  
    </div>  
</div>
```

This example shows a Dojo implementation of a TextBox and Button.



ID Selectors

- Similar to class selectors, but each ID attribute must be unique on a page
- IDs are case sensitive
- Use hash (#) notation

```
#mainNav { margin: 5px; }  
#mainNav li { float: left; }
```

```
<ul id="mainNav">  
  <li><a href="home.html">Home</a></li>  
  <li><a href="about.html">About</a></li>  
</ul>
```



Before and After Pseudo Classes

- Can be used to insert generated content before or after an element's *content* (not before or after the element itself)

```
p.caution:before { content: "Beware: " }  
p.caution:after { content: "!!!!" }
```

```
<p class="caution">Don't talk to strangers</p>
```

Renders:

“Beware: Don’t talk to strangers!!!”

You should not use :before and :after for general use because it mixes the layers in a layered architecture, but it can be very useful as a hack (presented in the clearfix discussion at the end of the next chapter).



CSS3 Selectors: Direct Child

- Selects a specified direct child of a parent

```
p > em { text-decoration: underline; }
```

```
<body>
```

target this

```
<p>This is a <em>great</em> paragraph</p>
```

```
<p>This is not a <span><em>great</em></span></p>
```

not this



CSS3 Selectors: Direct Adjacent Siblings

- Selects sibling immediately following a specified element

```
h2 + h3 { margin-top: 10px; }
```

```
<div>
  <h2>Economy Recovers!</h2>
  <h3>Fed is Optimistic</h3> ← target this
  <p>Lorem ipsum dolor...</p>
</div>

<div>
  <h2>Economy Recovers!</h2>
  <p>Lorem ipsum dolor...</p>
  <h3>Fed is Optimistic</h3> ← not this
</div>
```



CSS3 Selectors: Indirect Adjacent Siblings

- Selects a sibling of a specified node
 - Must be following, but not immediately following

```
.first ~ li { margin-top: 10px; }
```

```
<ul>
  <li class="first">...</li>
  <li>...</li> ←
  <li>...</li>
  <li class="last">...</li>
</ul>
```

target's these

Useful when selecting "a bunch
of items after the first one"



CSS3 Selectors: Attribute Selectors

- Selects elements based on presence or values of attributes

```
img[title] {border: 2px solid #000;}
```

Styles any with a title attribute.

```
img[title="Figure"] {border: 2px solid #000;}
```

Styles any with the title attribute "Figure", symbol = signifies an exact match.

```
img[title~="Figure"] {border: 2px solid #000;}
```

Note the additional “~”. Styles any with a title attribute that includes "Figure" in a space-separated list of words.

```
*[lang|= "en"] {color: #ccc;}
```

Styles any element with a lang attribute that begins with "en" in a hyphen-separated list.



:first-child, :last-child, nth-child

- Styles either the first, last child, or nth-child of an element

```
<div>
  <p>Foo (I should be red)</p>
  <p>Foo (I should be green)</p>
  <p>Foo (I should be blue)</p>
</div>
```

```
p:first-child {color: red;}
p:last-child {color: blue;}
p:nth-child(2) {color: green;}
p:nth-child('odd') { font-style: italic; }
```

Foo (I should be red)
Foo (I should be green)
Foo (I should be blue)

Foo (I should be blue)

- For browsers not supporting these, the fallback is to attach class names manually to HTML elements

Some of the selectors presented here are actually from CSS2.1.



Choosing Selectors

- There are a multitude of options for how to construct a given selector

```
<div class="mod">
  <ul id="mainNav">
    <li><a id="homeLink" href="home.html">home</a></li>
    <li><a id="aboutLink" href="about.html">about</a></li>
    <li><a id="careersLink" href="careers.html">careers</a></li>
  </ul>
</div>
```

all
refer
to

<code>a</code> <code>#aboutLink</code> <code>a#aboutLink</code> <code>#mainNav li #aboutLink</code> <code>div.mod ul#mainNav li a#aboutLink</code> <code>.mod #aboutLink</code>	<code>{ color: white; }</code> <code>{ color: blue; }</code> <code>{ color: green; }</code> <code>{ color: yellow; }</code> <code>{ color: red; }</code> <code>{ color: black; }</code>
--	--

What color will this anchor be?

The answer is red. The reason is explained on the next slide.



Selector Weight (Specificity)

- Calculated as follows:
 - Count the ID attributes (= a)
 - Count the Class or other attributes (= b)
 - Count the elements (= c)
- Concatenating a+b+c gives you the selector weight or specificity:

```
*                  {} /* a=0 b=0 c=0 -> specificity = 0 */
li                 {} /* a=0 b=0 c=1 -> specificity = 1 */
ul li              {} /* a=0 b=0 c=2 -> specificity = 2 */
ul ol+li          {} /* a=0 b=0 c=3 -> specificity = 3 */
h1 + *[REL=up]   {} /* a=0 b=1 c=1 -> specificity = 11 */
ul ol li.first    {} /* a=0 b=1 c=3 -> specificity = 13 */
li.first.level   {} /* a=0 b=2 c=1 -> specificity = 21 */
#x34y            {} /* a=1 b=0 c=0 -> specificity = 100 */
```

Specificity is a misleading term: longer selectors can appear to be more specific, but can actually have lower weight (last, and next to last examples above)



!important

- Takes precedence over all other rules

```
div {  
    font-size: 1.5em;  
    color: #000 !important;  
    padding: 0.2em;  
}
```

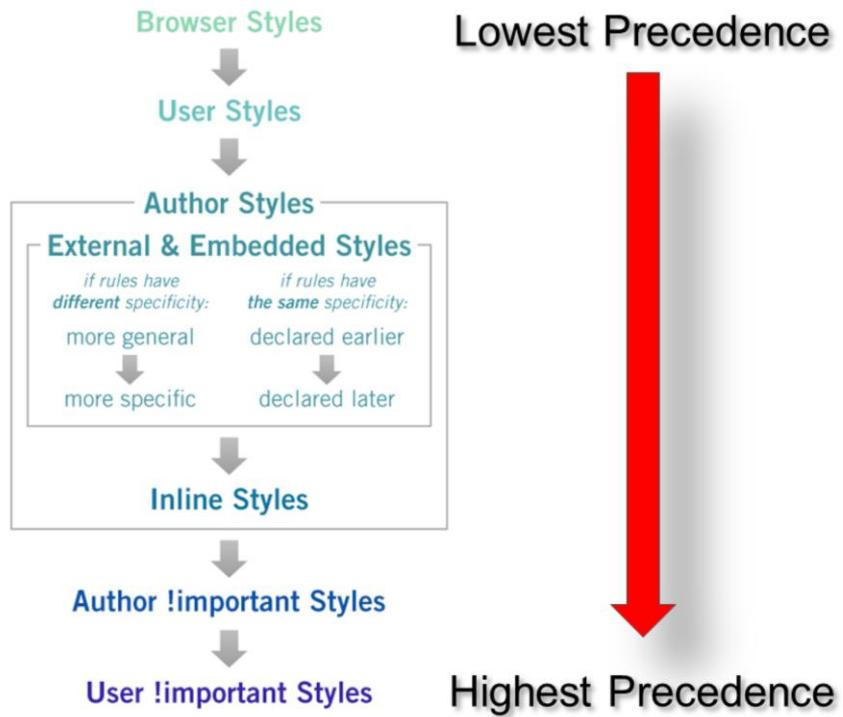
Generally, use this for debugging only!

The use of !important is a bad practice to begin taking advantage of. Yes, the argument could be made that when using certain frameworks, in order for your selectors to "beat" their selectors, you might use !important. However, once you use it, it has to be used again if you ever want to beat *that* selector.

It leads to a slippery slope that finds !important being used all over the place before long.



Cascade Precedence



<http://www.communitymx.com/content/article.cfm?page=2&cid=2795D>



Normalizing Browser Styles

- Browsers yield different default style values
- It's imperative to set a level playing field
- **reset.css**

<http://yui.yahooapis.com/combo?2.7.0/build/reset/reset.css>

```
body,div,d1,dt,dd,ul,ol,li,h1,h2,h3,h4,h5,h6,pre,form,fieldset,  
input,textarea,p,blockquote,th,td {  
    margin:0;  
    padding:0;  
}  
table {  
    border-collapse:collapse;  
    border-spacing:0;  
}  
[...]
```



HTML5 Resets

- With the arrival of HTML5, new values will need to be placed into a reset
- HTML5 Doctor has based their reset on Eric Meyer's HTML4 reset
 - <http://html5resetcss.googlecode.com/files/html5reset-1.6.1.css>
- Others exist as well
 - <http://html5reset.org/>
 - <http://html5boilerplate.com/>



HTML5Doctor Reset CSS

```
html, body, div, span, object, iframe, h1, h2, h3, h4, h5, h6, p, blockquote, pre,  
abbr, address, cite, code, del, dfn, em, img, ins, kbd, q, samp, small, strong, sub,  
sup, var, b, i, dl, dt, dd, ol, ul, li, fieldset, form, label, legend, table, caption, tbody,  
tfoot, thead, tr, th, td, article, aside, canvas, details, figcaption, figure, footer,  
header, hgroup, menu, nav, section, summary, time, mark, audio, video {  
    margin:0; padding:0; border:0; outline:0; font-size:100%;  
    vertical-align:baseline; background:transparent;  
}  
  
article,aside,details,figcaption,figure,footer,header,hgroup,menu,nav,section {  
    display:block;  
}  
  
a {  
    margin:0; padding:0; font-size:100%;  
    vertical-align:baseline; background:transparent;  
}
```

Partial listing of the full reset

html5doctor.com Reset Stylesheet



Summary

- Typically embed style or externally link stylesheets into solutions
 - The latter approach affords better reusability
- All modern browsers support the newer CSS 3 selectors
 - Use of **Direct Child & Attribute Selectors** should become part of your CSS arsenal
- Remember specificity controls what is rendered, not document order!

Chapter 3

CSS Box Model

Laying Out Your Pages



Overview

Box Model Properties

Positioning

Floating



CSS Box Model

- The CSS box model describes the rectangular boxes generated for *each element* defined in the HTML
- Elements are rendered using box model properties depending on whether an element is a **block** element or an **inline** element

Block Elements

div
ul
li
form
h1-h6
p

Inline Elements

span
input
a
img

Block elements fill the width of their parent, next one renders beneath last

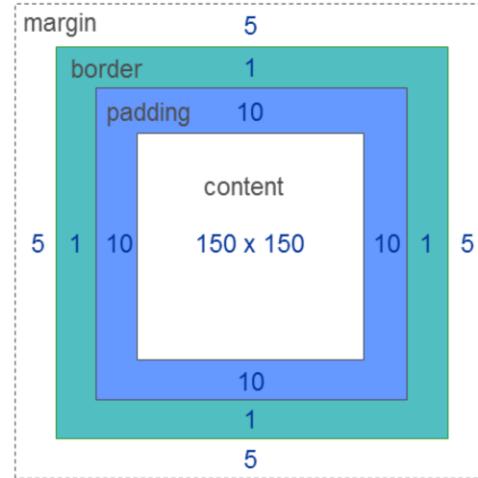
Inline elements render side-by-side, size to their content, and cannot control vertical properties such as **height** and **margin-top/bottom**



Box Dimensions

- Each box has a **content** area (e.g., text, an image, etc.) and optional surrounding **padding**, **border**, and **margin** areas

```
{  
    height: 150px;  
    width: 150px;  
    padding: 10px;  
    border-width: 1px;  
    margin: 5px;  
}
```



Margin is transparent: background colors or images will not show in that area.



Positioning Schemes

1. Normal Flow

- Content flows from top to bottom, and left to right
- Block-level elements generate line breaks

2. Absolute Positioning

- Content removed entirely from normal flow and positioned with respect to an ancestor element

3. Float

- Content first laid out in normal flow, then taken out of flow and shifted as far right or left as possible

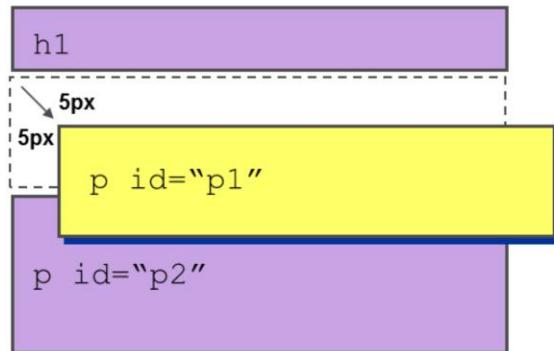
We use the “position” and “float” properties to assign a scheme for each element



Relative Positioning

- Part of the normal flow scheme
- Should really be called “offset positioning”
- Content is laid out according to normal flow, and then offset from its original position

```
<h1></h1>
<p id="p1"></p>
<p id="p2"></p>
[...]
p#p1 {
    position: relative;
    top: 5px;
    left: 5px;
}
```



The space that the element would have occupied in normal flow is maintained. Element now layers over neighboring elements.

Properties can be top | right | bottom | left

Values can be positive or negative

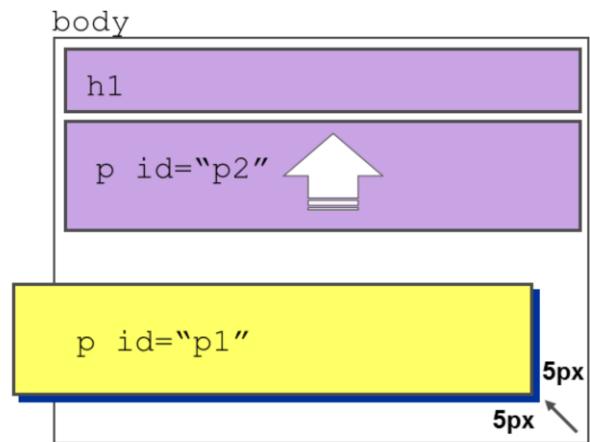


Absolute Positioning

- Removed from normal flow

```
<body>
  <h1></h1>
  <p id="p1"></p>
  <p id="p2"></p>
</body>
[...]
```

```
p#p1 {
  position: absolute;
  bottom: 5px;
  right: 5px;
}
```



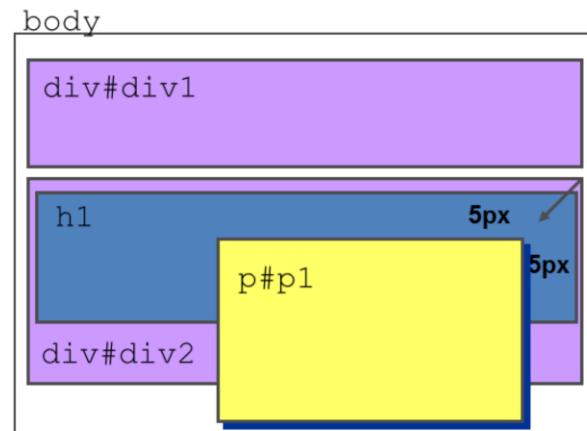


Absolute Positioning

- Positioned with respect to the body or the next highest *positioned* ancestor
 - Travel up the document tree until encountering an element with `position: relative|absolute`

```
<div id="div1"></div>
<div id="div2">
  <h1></h1>
  <p id="p1"></p>
</div>
[...]

p#p1 {
  position: absolute;
  top: 5px;
  right: 5px;
  width: 50%;
}
div#div2 {
  position: relative;
}
```



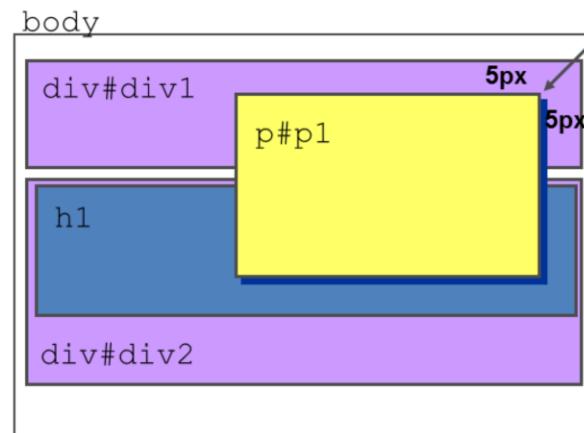


Absolute Positioning

- Positioned with respect to the body or the next highest *positioned* ancestor
 - Travel up the document tree until encountering an element with `position: relative|absolute`

```
<div id="div1"></div>
<div id="div2">
  <h1></h1>
  <p id="p1"></p>
</div>
[...]

p#p1 {
  position: absolute;
  top: 5px;
  right: 5px;
  width: 50%;
}
```



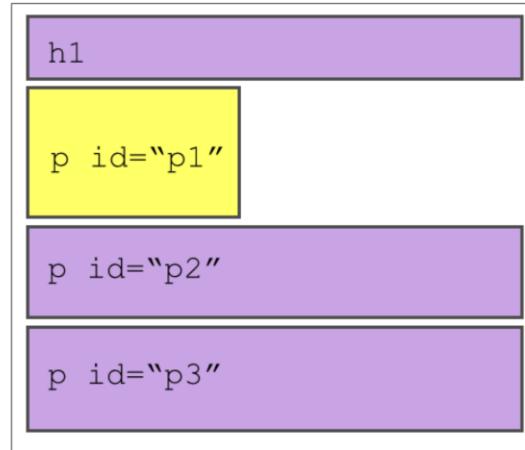
Showcase what happens when positioning is removed from ancestor div (now in relation to the body)



Floated Elements

- First, content is laid out according to normal flow

```
<h1></h1>
<p id="p1"></p>
<p id="p2"></p>
<p id="p3"></p>
[...]
p#p1 {
    width: 40%;
```

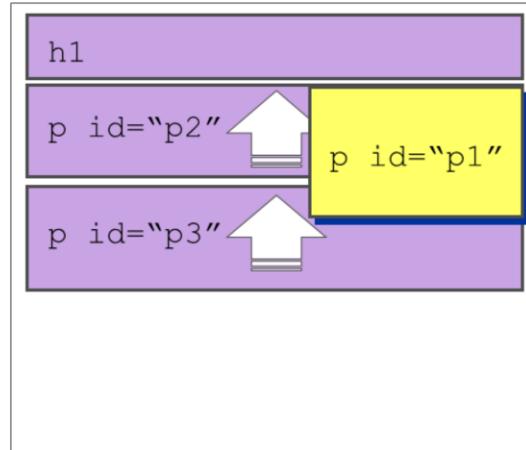




Rules of Floating (1 of 5)

- Floats shift up then left or right as far as possible
 - Subsequent block elements move up behind floated element

```
<h1></h1>
<p id="p1"></p>
<p id="p2"></p>
<p id="p3"></p>
[...]
p#p1 {
    width: 40%;
    float: right;
}
```



Floats are, in relation to block level elements, removed from the flow. They will lift up to the first sibling element that precedes it that is not floated.

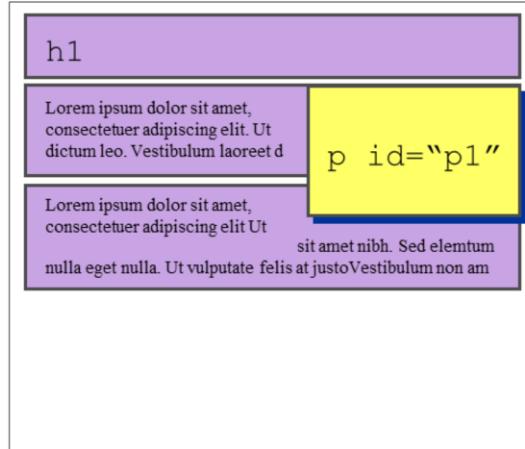


Rules of Floating (2 of 5)

- Inline content (i.e. text) flows *around* floated elements

```
<h1></h1>
<p id="p1">Lorem...</p>
<p id="p2">Lorem...</p>
<p id="p3"></p>
[...]

p#p1 {
    width: 40%;
    float: right;
}
```

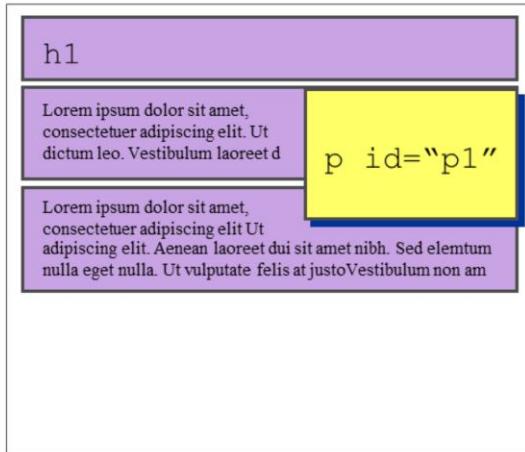




Rules of Floating (3 of 5)

- Floats are block-level elements
 - Will override value of “display” property
 - However, must have an explicit width set, else shrink-to-fit

```
p#p1 {  
    width: 40%;  
    float: right;  
}
```



(So, it's not entirely removed from the normal flow) Key difference from absolutely positioned elements.



Rules of Floating (4 of 5)

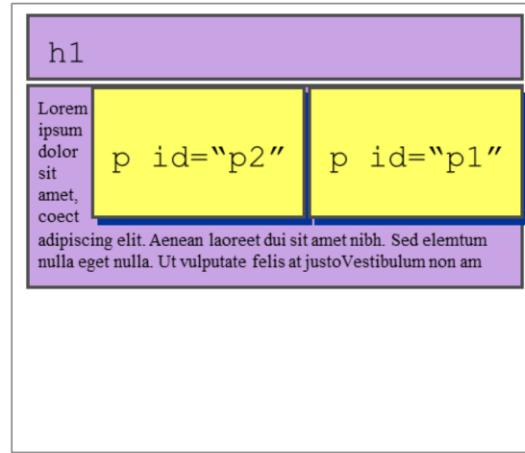
- Floats are shifted left or right until their outer edge touches the containing block edge *or the outer edge of another float*

```

<h1></h1>
<p id="p1">/p>
<p id="p2"></p>
<p id="p3">Lorem...</p>
[...]

p#p1,
p#p2 {
    width: 40%;
    float: right;
}

```



All floated objects interact with each other on the same “pane” or “context”. (vs. Absolutely Pos. elements will stack, each on their own layer.)

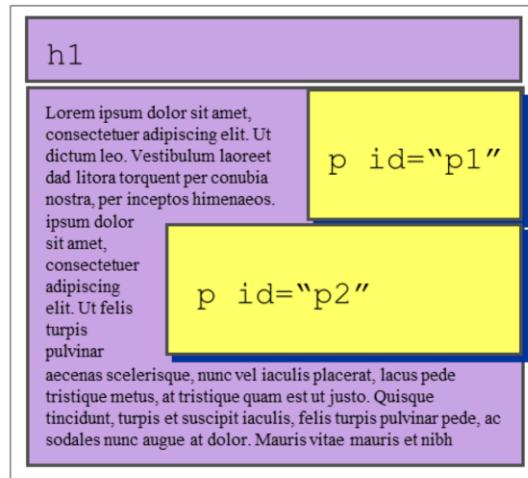


Rules of Floating (5 of 5)

- If there isn't enough horizontal room on the current line for the float, it is shifted downward, line by line, until a line has room for it.

```
<h1></h1>
<p id="p1">/p>
<p id="p2"></p>
<p id="p3">Lorem...</p>
[...]
```

```
p#p1 {
    width: 40%;
    float: right;
}
p#p2 {
    width: 70%;
    float: right;
}
```





Background Properties

```
background-color:      [keyword|hex|rgb|rgba|hsl|hsla]
background-image:     [url(path) ]
background-repeat:   [no-repeat|repeat-x|repeat-y]
background-position: [x-pos y-pos]
background-attachment: [scroll|fixed]
background-size:      [width height|contain|cover]
background-origin:    [padding-box|border-box|content-box]
background-clip:      [padding-box|border-box|content-box]
```

New in CSS3

Correct Shorthand:

```
background: color position size repeat origin clip attachment image;
background: pink 10px 50% no-repeat url(smiley.jpg);
```



examples/03_css/bgnd_prop.html

IE8 and earlier do not support multiple background images. Newer browsers do.

background-size can be one of several values including a width and height or
 contain - image will fit within background
 cover - image will completely cover the background,
 possibly clipping it

background-origin: specifies where background-position will be relative to

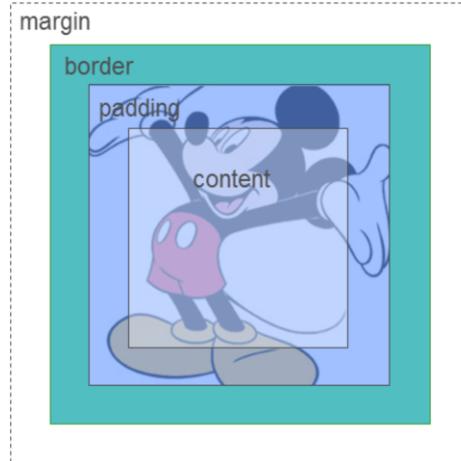
background-clip: specifies the area to be painted by the background-image



Using the *background* Property

- Any background properties (image, color etc..) will apply to the **content** and **padding** areas.

```
{  
height: 150px;  
width: 150px;  
padding: 10px;  
border-width: 1px;  
margin: 5px;  
background: url(mickey.jpg)  
no-repeat;  
}
```

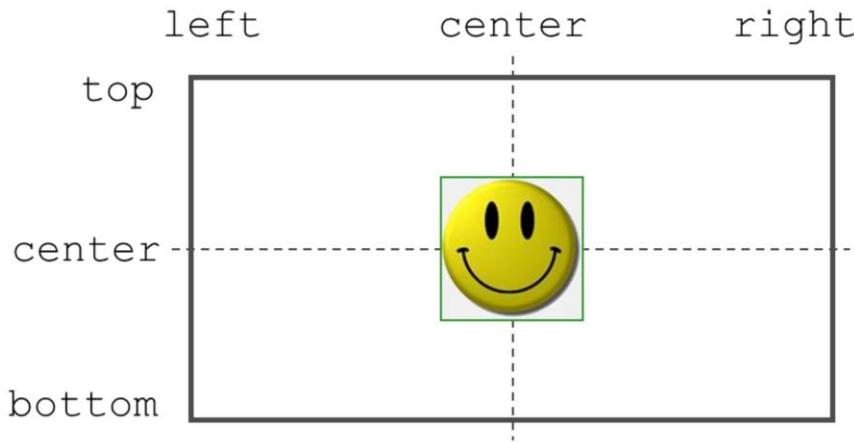


Margin is transparent: background colors or images will not show in that area.



Background Positioning

- By **keyword**

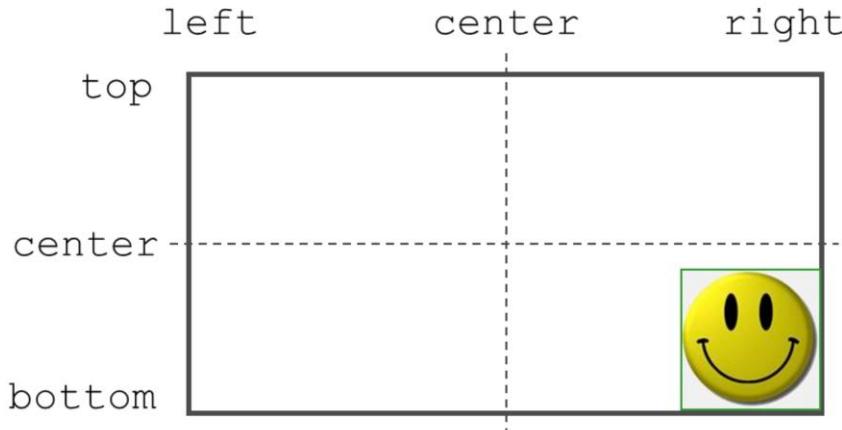


```
background: url(smiley.jpg) no-repeat center center;
```



Background Positioning

- By **keyword**



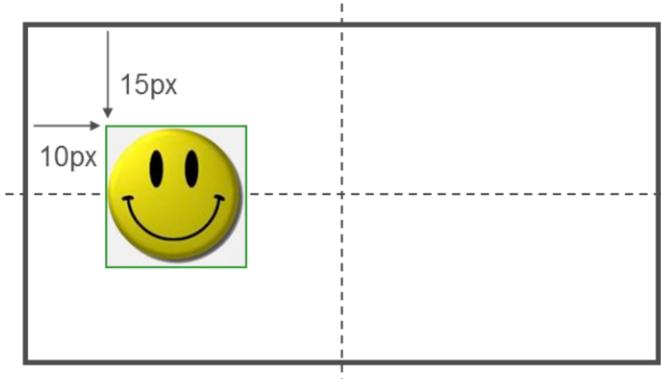
```
background: url(smiley.jpg) no-repeat right bottom;
```

Keyword here sets the right of the image with the right of the container, and the bottom with the bottom



Background Positioning

- By **fixed unit**



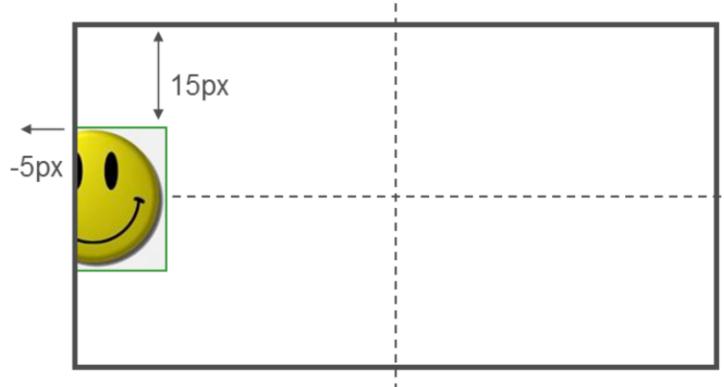
```
background: url(smiley.jpg) no-repeat 10px 15px;
```

Sets the start position (top/left) of the background image within the container element



Background Positioning

- By **fixed unit**

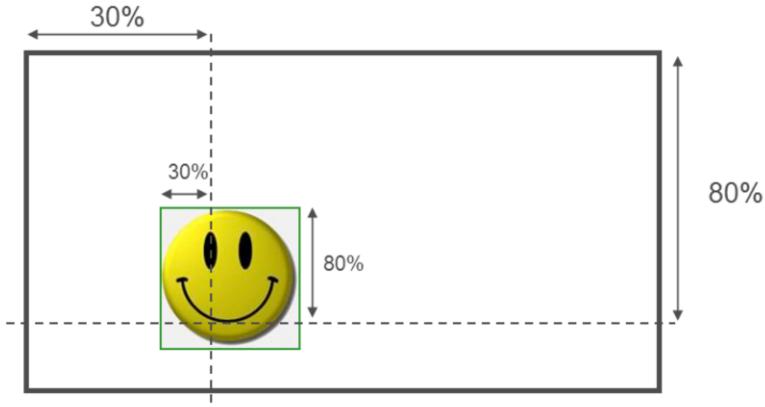


```
background: url(smiley.jpg) no-repeat -5px 15px;
```



Background Positioning

- By percentage



```
background: url(smiley.jpg) no-repeat 30% 80%;
```

Matches the x-axis percentage of the container with the same x-axis percentage of the background image

Same with y-axis

Very hard to predict results – usually avoided.



Exercise 2 - It's Your Turn

- Create the following by using CSS (HTML is already provided). Work from the labs/lab02/starter folder.

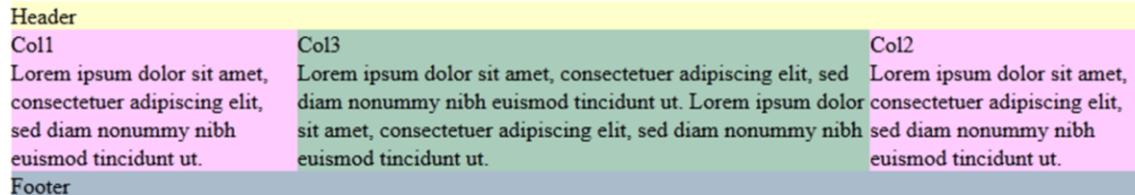


- Hints:
 - You should only have to complete lab02.css
 - Use floating to make list items stack horizontally
 - Add the background image to each anchor
 - The image is approx. 100 x 75 pixels
 - Hints: The image is on the anchor, to give inline elements a height and width you must make them block elements
 - For this exercise, give the a height



Floating for Layout

- Nearly every modern web page uses floating to control layout:



```
<div id="wrapper">
  <div id="header"></div>
  <div id="container">
    <div id="col1"></div>
    <div id="col2"></div>
    <div id="col3"></div>
  </div>
  <div id="footer"></div>
</div>
```

```
#header { background-color: #ffc; }
#col1 { background-color: #fcf;
         float: left;
         width: 25%; }
#col2 { background-color: #fcf;
         float: right;
         width: 25%; }
#col3 { background-color: #acb; }
#footer { background-color: #abc; }
```

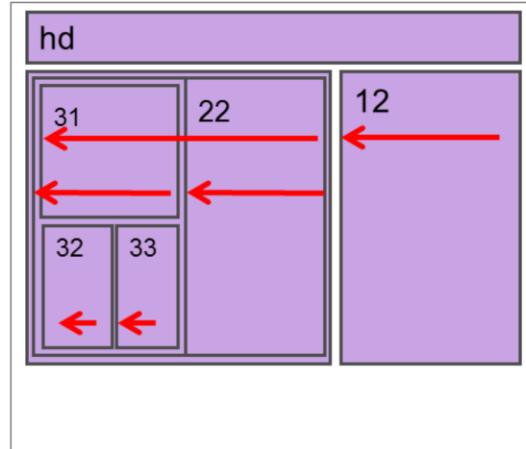
See examples in examples/03_css/floating0.html



Two-Columns

- Many layouts are decomposed into two columns

```
<div id="wrapper">
  <div id="hd"></div>
  <div id="col11">
    <div id="col21">
      <div id="col31"></div>
      <div id="col32"></div>
      <div id="col33"></div>
    </div>
    <div id="col22"></div>
  </div>
  <div id="col12"></div>
</div>
```



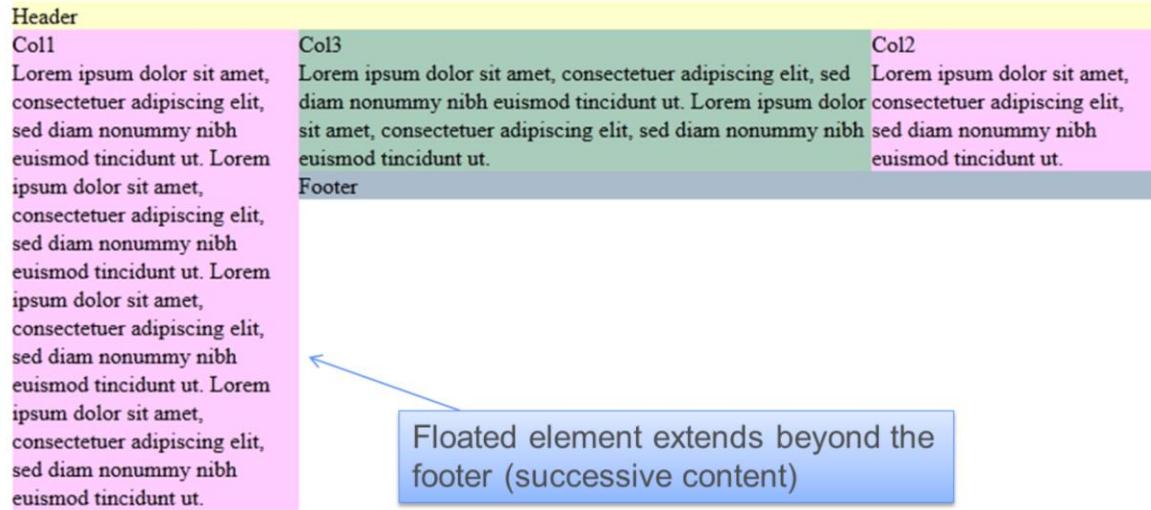
```
#col11, #col21, #col32 { width: 55%; float: left; }
#col12, #col22, #col33 { width: 45%; float: left; }
```

See examples/03_css/floating1.html



Dilemma with Floating

- What happens if floated content gets too large?

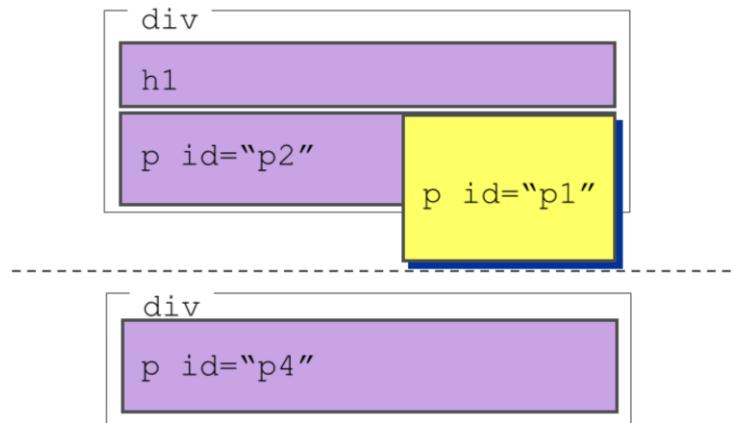


See examples in examples/03_css/floating2.html



Solving this Dilemma: Clearing

- Often, you want to define a break point where content stops going up behind a float

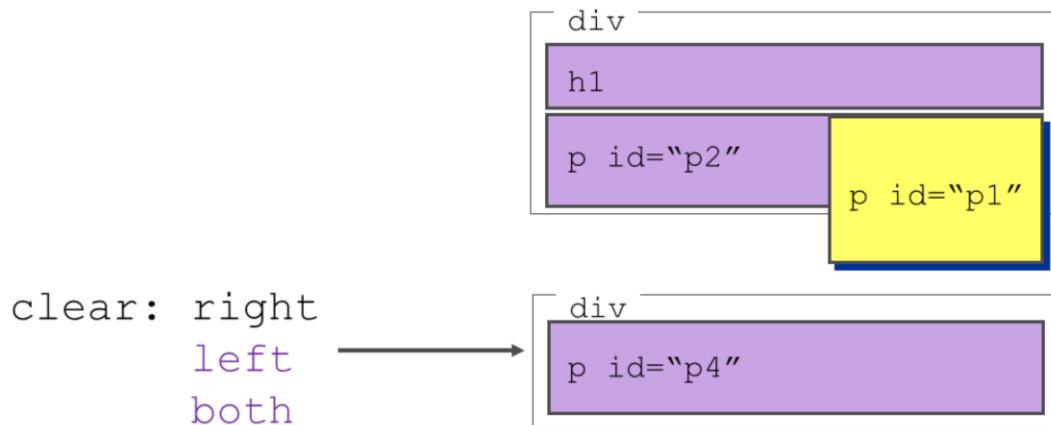


4 ways to do this...



Tactic #1: Clearing Floats

- The “clear” property on the subsequent element

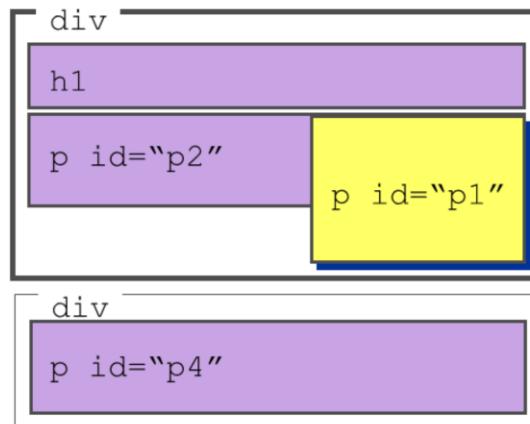


The disadvantage to this approach is that the subsequent div must "know" to clear the content above it. There is a coupling or dependency created.



Tactic #2: Clearing Floats

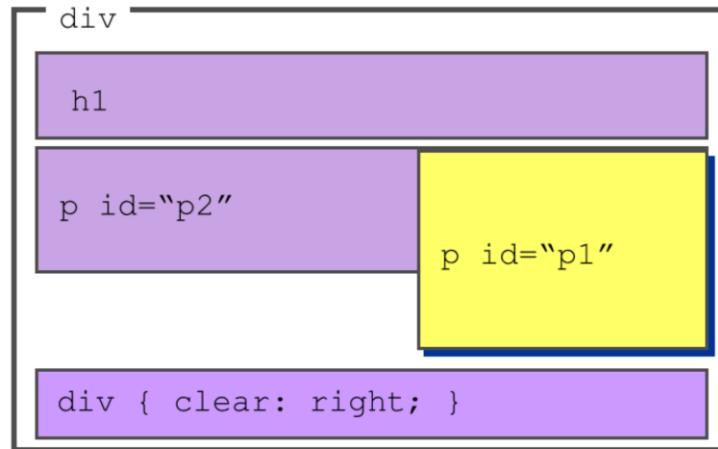
- Force the parent to enclose the float;
"Float nearly everything"



This approach leads to the task of needing to float everything.



Tactic #3: Self-clearing DIV



By placing a `<div>` inside the parent of a floated element, but after the floated element, we can give it the `clear` property.

This `<div>` has no height, cannot be seen, but ensures the parent always encapsulates the floated element.

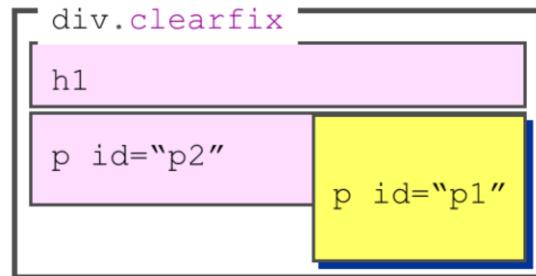
This approach introduces a non-semantic element, usually a `div` placed after the floated content which then clears. This works nicely, but the `div` serves no purpose but to aid in presentation.



Tactic #4: The Best Technique

- The clearfix technique

```
.clearfix:after {  
    content: "";  
    display: block;  
    clear: both;  
}  
  
.clearfix {  
    zoom: 1; /* IE hack */  
}
```



What is this hack with IE?

Read on...



clearfix Now Added

- clearfix has been added to the containing element now

Header

Col1

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut.

Col3

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut.

Col2

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut.

Footer

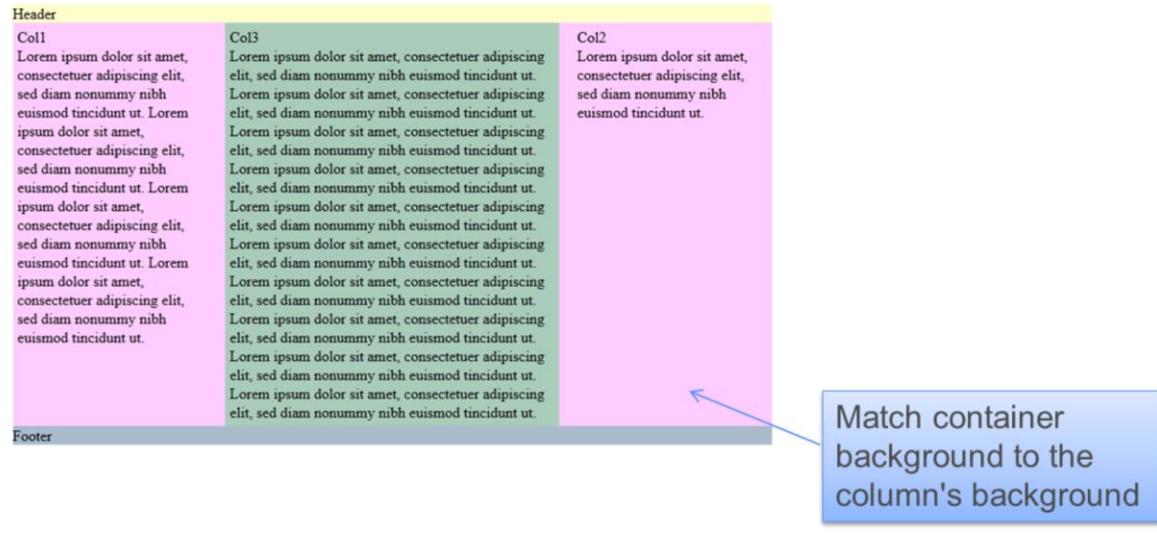
```
#container:after {  
    content: '';  
    display: none;  
    clear: both;  
}  
  
#container { zoom: 1; }
```

See examples in examples/03_css/floating3.html



What About My Uneven Column Heights?

- Often CSS will require a little crafty thinking or some trickery
 - Our columns are given the appearance of being the same height

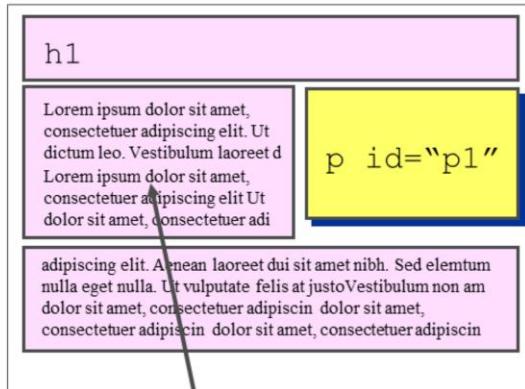
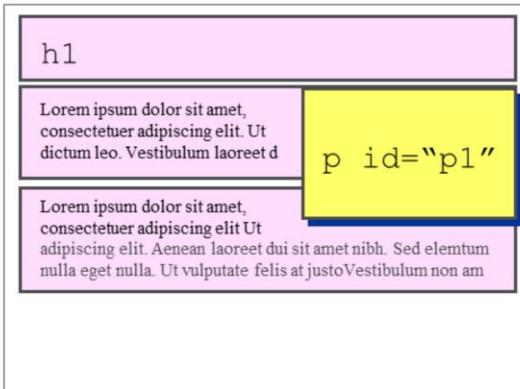


See examples in examples/03_css/floating4.html

For this trick to work, the middle column should be the longest one. But generally this requirement is not a problem.



Why zoom:1 for IE?



The need to specify zoom: 1 for IE will disappear if support for IE8 and earlier can be dropped

"hasLayout" answers many of the reasons for IE rendering bugs

In older IE's, a block element with "hasLayout" will not run behind the floated element as the spec dictates. Instead, the entire block will run alongside the float.

hasLayout is an important feature to understand how IE behaves when content renders. For more on this hidden property, read this very well written article:
<http://www.satzansatz.de/cssd/onhavinglayout.html>

hasLayout is used to either "trigger" layout if set or NOT trigger layout if NOT set. hasLayout, however, is a readonly property and therefore it cannot be directly set. Instead, a number of properties can be used to "trigger" hasLayout in IE. (refer to the above cited article for these). This also explains why zoom:1 on the previous page happens to work for the clearfix on IE.



Exercise 3 - It's Your Turn

- Return to lab 02 and modify the solution by **removing the 'height'** on the or the <div> depending on your solution
- Use 'clearfix' technique to force the purple background to remain in tact. Ensure the solution renders as before, this time without the or <div> height





Exercise 4 - It's Your Turn

Multi-column Layout

Adv. JavaScript and HTML5

Jim Pinkado
2113 Redwood Blvd. Long Beach CA
90314
(204)740-9000 (work)
pinkman@rocketmail.com
Distribution Manager at Hollywood West Novelties

Contact Id	Name	Address	Phone Num	Phone Type	Email	Company	Job Title
500	Red Vectors	4517 Elm St. Riverside	(301)356-8921	home	fthompson@yahoo.com	ABC Inc.	President
501	Bob Green	2101 Eucalyptus Ave.	(202)901-2121	home	dbreali@hotmail.com	Tupelo Industries	Product Manager
502	John Brown	asdf	(719)421-8875	home	j67712@gmail.com	Last Rites LLC	Undertaker
503	Tina O. Range	82 Pine Dr. Lakewood	(212)432-0944	home	tornut@bechp.com	Best Holistic Practices	Customer Service Representative
504	Berry	3012 Mahogany Ln.	(202)685-2323	home	bleubry@yahoo.com	Roller Heights	Owner
505	Blumenthal	Denver CO 80101				Packaging	
506	Jim Pinkado	2113 Redwood Blvd.	(204)740-9000	work	jpinkman@rocketmail.com	Hollywood West	Distribution Manager
507	Alicia Grey	Long Beach CA 90314				Novelties	
508	Violet Waters	415 Poplar Ct. St.	(211)670-6780	home	aigrey@blanksystems.com	Bank Systems	Lead Technical Engineer
509	Sandy White	Louis MO 72210	(302)390-1181	home	waters@medcare.com	Home Medical Services	Regional Account Representative
510	Kay Black	WA 92230	(213)221-4143	home	swhite@bricks.com	Bricks and More	Product Sales
511	John Brown	1241 Maple Pl. Phoenix	(401)322-8728	home	kbs2101@yahoo.com	Certified Signing Auditor	Graphics Artist
512	Alicia Grey	TX 72110				Last Rites LLC	Undertaker
	Berry	3012 Mahogany Ln.	(202)685-2323	home	bleubry@yahoo.com	Bank Systems	Lead Technical Inc. Engineer
	Blumenthal	Denver CO 80101				Roller Heights	Owner
						Packaging	

Contact ID:	501
Name:	Bob Green
Address:	2101 Eucalyptus Ave. Philadelphia
Primary Phone:	(202)901-2121
Type:	home
Email:	dbreali@hotmail.com
Company:	Tupelo Industries
Job Title:	Product Manager

[Create](#) [Update](#) [Delete](#)

Contact ID: 505

[Search](#)

[Search \(XML response\)](#)

Locate step-by-step instructions for this lab in the back of your manual



Summary

- The CSS Box Model defines numerous properties that help control an element's look
- Lay boxes out using floating and/or absolute positioning
- Some favor the use of `display:inline-block` instead of floating
 - While this alternative approach is okay to use, be aware that `display:inline-block` can cause an undesired space to occur between elements
 - Extra work must then be taken to remove that space if desired

Chapter 4

CSS 3 Features

Enhancing Solutions with Style



Overview

Color Options

Gradients

Rounded Corners

Transforms



The CSS 3 Standard

- CSS 3 standard managed by W3C is so large it has been broken into over 50 modules
- The **selectors** module has been given the highest priority and is already *Recommendation* status
- Most other modules are in working draft status and are subject to change, including:
 - **animations, transformations, transitions**
- Parts of the standard have been around for years

Good reference on CSS3 standards status: <http://www.css3.info/modules/>



CSS 3 Feature Support

Feature	IE 6	IE 7	IE8	IE9 +	FF 3	FF 3.6	FF4+	SF 4	SF 5+	Ch	Op 10	Op 10.5
Hover Effects	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Rounded Corners	N	N	N	Y	Y	Y	Y	N	Y	Y	N	Y
Drop Shadows	N	N	Y	Y	N	Y	Y	Y	Y	Y	N	Y
Gradients	N	N	Y	Y	N	Y	Y	Y	Y	Y	N	N
Transparency (RGBA)	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y
Transparency (Opacity)	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Transitions	N	N	N	N	N	N	Y	Y	Y	Y	N	Y
2D Transforms	N	N	N	Y	N	Y	Y	Y	Y	Y	N	Y
Data URIs	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

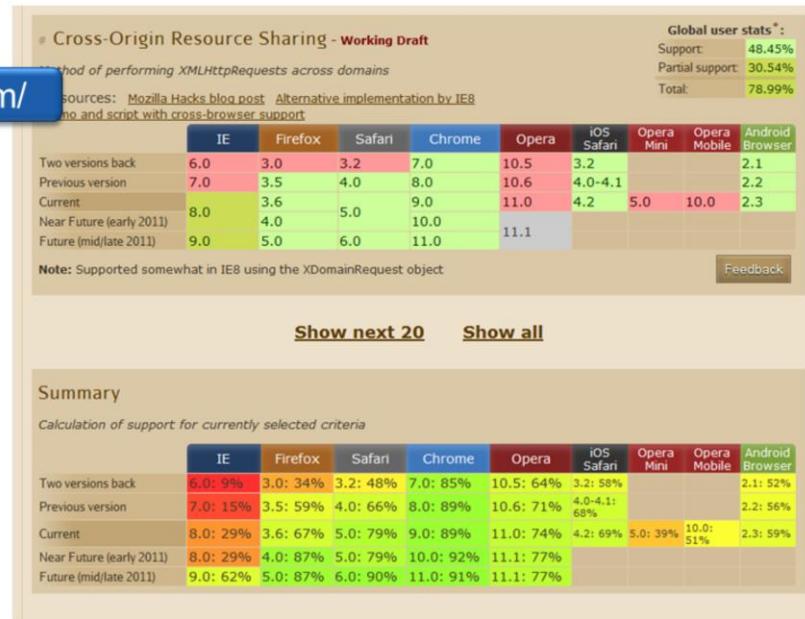
IE9 and earlier do not support 3D transforms, border-images, animations, transitions, multiple column layouts, and text-shadow.



Browser Support for CSS 3

- Browsers currently implement varying support for CSS 3

<http://caniuse.com/>



Use this site as a browser support reference

There are a number of sites that provide matrices, summaries, or charts of browser support. Some of these sites include:

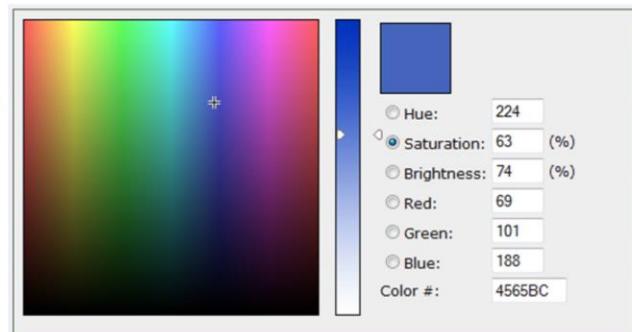
html5demos.com
html5readiness.com
findmebyip.com/litmus
en.wikipedia.org/wiki/Comparison_of_web_browsers

The one shown here provides information current through the latest (even next) release of each browser.



New CSS Color Options

- CSS3 provides additional ways of specifying color selections
 - Classic approach: hex values, RGB percentage/value
 - New approaches:
 - **RGBA**
 - **HSL (Hue/Saturation/Lightness)**
 - **HSLA**



RGBA & HSLA offer a fourth value: **alpha transparency** where:
0 = fully transparent
1 = completely opaque

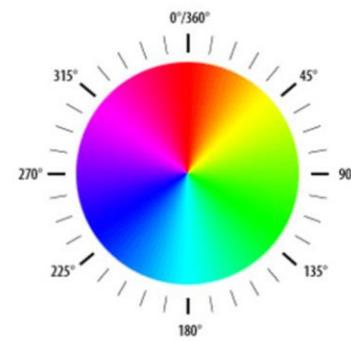
<http://www.w3.org/TR/css3-color/>

Hexadecimal notation does not have an equivalent way to express alpha-transparency values.



How HSL Works

- HSL (Hue / Saturation / Lightness) works as follows:
 - **Hue** is defined as a degree on a color wheel
 - 0 (360) is red
 - 240 is blue
 - 120 is green
 - **Saturation** is a percentage of the full color (0% – 100%)
 - **Lightness** (0% dark – 100% white)





New Color Examples

```
div {  
  color: rgb(0,0,0);  
  background-color: rgb(255,255,255);  
}  
  
div {  
  color: rgba(0,0,0,0.6);           /* semitransparent black */  
  background-color: rgba(255,255,255,0.6); /* semitrans. wht */  
}  
  
div {  
  color: hsl(0,0%,0%);           /* black */  
  background-color: hsl(0,0%,100%); /* white */  
}  
  
div {  
  color: hsla(0,0%,0%,0.6);      /* semitransparent white */  
  background-color: hsla(0,0%,100%,0.6); /* semitrans black */  
}
```



New Color Improvements

- Use a solid background color for IE6-8 as a fallback:

```
selector {  
    background-color: #f00; /*fallback */  
    background-color: rgba(255,0,0, 0.5); ← All other browsers  
}
```

IE6-8

All other browsers

Note that the `-ms-filter opacity` (and the standard `opacity`) are slightly different in meaning than the `background-color alpha transparency` value. The later will only make the `background-color` transparent, while the `-ms-filter (or opacity)` option will make the entire element transparent including foreground text color, border, padding, etc.



Progressive Enhancement with Colors

- Browsers will drop the remaining properties in a rule if they do not recognize one
 - Place more generalized property first to capture all browsers
 - Place the more specialized property last

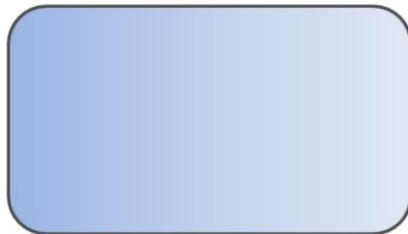
```
selector {  
    color: #f00; /* fallback */  
    color: rgba(255, 0, 0, 0.5);  
}
```

Newer browsers will be progressively enhanced by taking advantage of the second color property



Rounded Corners

- Rounded corners can be achieved using CSS only
 - No longer requires use of images or markup
 - IE8 and earlier see non-rounded (square) corners



```
div {  
    border-radius: 5px; /* Opera 10.5, IE 9,  
}                                SF5, Chrome, FF4 */
```

<http://www.w3.org/TR/css3-background/>

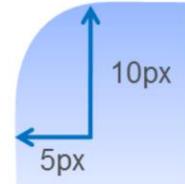
Most browsers today support the std property and extensions are generally not necessary.



Rounded Corners Syntax

- Specific properties can define each corner

```
border-top-left-radius: 5px 10px;  
border-top-right-radius: 10px;  
border-bottom-left-radius: 10px;  
border-bottom-right-radius: 10% 5%;
```



- Shorthand syntax:

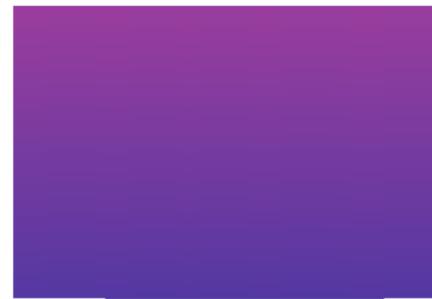
```
border-radius: [ <length> | <percentage> ]{1,2,4}  
           [ / [ <length> | <percentage> ]{1,2,4} ]?  
  
border-radius: 5px 10px 5px 10px / 10px 5px 10px 5px;  
border-radius: 5px;  
border-radius: 5px 10px / 10px;
```

There are several ways to specify rounded-corner values. Either as one or two values for each corner or via the use of shorthands (shown at the bottom). Use the / operator to separate the x-values from the y-values.



Gradients

- Gradients use the ***background-image*** property
- For non-supporting browsers:
 - Use solid background color



```
.com-gradient {  
    background-color: #b82cb8;  
    background-image: linear-gradient(to bottom, #b82cb8, #2b29ba);  
    /* Ch26+, FF16+, IE10+ */  
}
```

Gradients receive various levels of support in browsers. Because of this a determination has to be made as to the need for gradients within non-supporting browsers. If the gradient is important to the design of the UI or presentation, then an appropriate solution must be achieved for all browsers. If it is not important, an option to simply degrade to a solid color background can be made.

Our rule of thumb continues, specify the generic "all-supported" value first, in this case the standard color value. Then "enhance it" with the browser-specific extensions, and finally specify the standards-based version last.

The screenshot shows the CSS3Please website. On the left, there's a code editor window containing CSS code for creating rounded corners and shadows. The code includes vendor-specific properties like -webkit-border-radius and -moz-background-clip. On the right, there's a preview area with a large red "CSS3, please!" heading and a message about the live editing feature.

```
/*
CSS3, Please! The Cross-Browser CSS3 Rule Generator
=====

You can edit the underlined values in this css file,
but don't worry about making sure the corresponding
values match, that's all done automagically for you.

Whenever you want, you can copy the whole or part of
this page and paste it into your own stylesheet.
*/
/* [to clipboard] [toggle rule off] */
.box_round {
    -webkit-border-radius: 12px; /* Saf3-4, iOS 1-3.2, Android <1.6 */
        border-radius: 12px; /* Opera 10.5, IE9, Saf5, Chrome, FF4+, Ios6 */
    /* useful if you don't want a bg color from leaking outside the border: */
    -moz-background-clip: padding; -webkit-background-clip: padding-box; background-clip: padding-box;
}

/*
[to clipboard] [toggle rule off] */
.box_shadow {
    -webkit-box-shadow: 0px 0px 4px 0px #ffffff; /* Saf3-4, iOS 4 */
        box-shadow: 0px 0px 4px 0px #ffffff; /* Opera 10.5, Ios6 */
}
```

CSS3, please!

This element will receive instant changes as you edit the CSS rules on the left. Enjoy!

Provides:

- 1- Interactive interface
- 2- Rules are in proper fallback order
- 3- Supported browsers are listed
- 4- Easy to copy-and-paste

CSS3Please.com provides a good way to remember the proper progressive enhancement order of elements in CSS. The page shows the supported browser version and the correct order in which the rules need to be placed.



Transforms

- Transformations such as `skew`, `rotate`, and `scaling`, `matrix`, `translate` can be performed on elements
- Use the standard and vendor-specific versions

```
.scale {  
    left: 25px;  
    -webkit-transform: scale(2.5);  
    -moz-transform: scale(2.5);  
    -o-transform: scale(2.5);  
    -ms-transform: scale(2.5);  
    transform: scale(2.5);  
}
```

The `transform` property allows elements to be skewed, rotated, scaled, and translated. Implementations may use this technique as long as the non-transformed browser versions are fully usable without being transformed.



Transforms

- Valid functions include:
 - **matrix** applies a transformation matrix
 - **rotate(deg)**
 - **scale(factor)** scales by a factor
 - **scaleX(factor)**
 - **scaleY(factor)**
 - **skew(angle)** skews on both axes
 - **skewX(angle)**
 - **skewY(angle)**
 - **translate(amount)** translates along x and y axis
 - **translateX(amount)**
 - **translateY(amount)**

The following provides a list of the transform functions available to supporting browsers. Be sure to indicate the proper CSS property prefix for specific browsers.



What Can Transforms Do?

- Check out some cool examples:

<http://www.creativebloq.com/css3/20-stunning-examples-css-3d-transforms-11112759>



There are numerous galleries that showcase great effects that can be generated with CSS animations, transforms, and transitions.



Exercise 5 - It's Your Turn

A CSS 3 Dialog Popup

Adv. JavaScript and HTML5

Contact Search

Contact ID	Name	Address	Phone Num	Type	Email	Company	Job Title
500	Red Vectors	4517 Elm St. Riverside NJ 07075	(301)356-8921	home	rthompson@yahoo.com	ABC Inc.	President
501	Bob Green	2101 Eucalyptus Ave. Philadelphia PA 09119	(202)601-2121	home	dtrees@hotmail.com	Tupelo Industries LLC	asdf
502	John Brown	asdff	(719)421-8875	home	b6712@gmail.com	Last Rites	Undertaker
503	Tina O. Range	82 Pine Dr. Lakewood CA 90713	(212)432-0944	home	tormit@bestwp.com	Best Holistic Practices	Customer Service Representative
504	Berry Blumenthal	301 Main Ln. Denver CO 80101	(719)421-8875	home	b6712@gmail.com	Blanks Systems Inc.	Representative
505	Jim Pinkado	2113 Redwood Blvd. Long Beach CA 90314	(204)740-9000 (work)	home	jpinkman@rocketmail.com	Blanks Systems Inc.	Owner
506	Alicia Grey	415 Poplar St. Louis MO 72210	(211)870-6780	home	aigrey@blanksystems.com	Lead Technical Engineer	Designer
507	Violet Waters	821 Ash Seattle WA	(719)421-8875	home	b6712@gmail.com	Holmes Heights	Packaging
508	Sandy White	225 Hickory Rd. Phoenix AZ 85310	(213)221-4143	home	swhite@bricks.com	Bricks Systems Inc.	Product Sales
509	Kay Black	1241 Maple Pl. Piano TX 72110	(401)322-8728	home	kbb2101@yahoo.com	Certified Signing Authorities LLC	Graphics Artist
510	John Brown	asdff	(719)421-8875	home	b6712@gmail.com	Last Rites	Undertaker
511	Alicia Grey	415 Poplar Ct. St. Louis MO 72210	(211)870-6780	home	aigrey@blanksystems.com	Blanks Systems Inc.	Lead Technical Engineer
512	Berry Blumenthal	301 Main Mahogany Ln. Denver CO 80101	(202)685-2323	home	b6712@gmail.com	Holmes Heights	Packaging

Contact ID:

Contact ID:

Locate step-by-step instructions for this lab in the back of your manual



Summary

- CSS 3 features have become very broadly supported by modern browsers
 - If IE6-8 support is not needed, most CSS 3 features may be implemented
- Older browsers might need proper fallback
- Place properties in a specific order to achieve progressively enhance newer browsers

Chapter 5

Responsive Solutions

Creating single solutions for multiple devices



Overview

What is Responsive Design?

CSS Media Queries

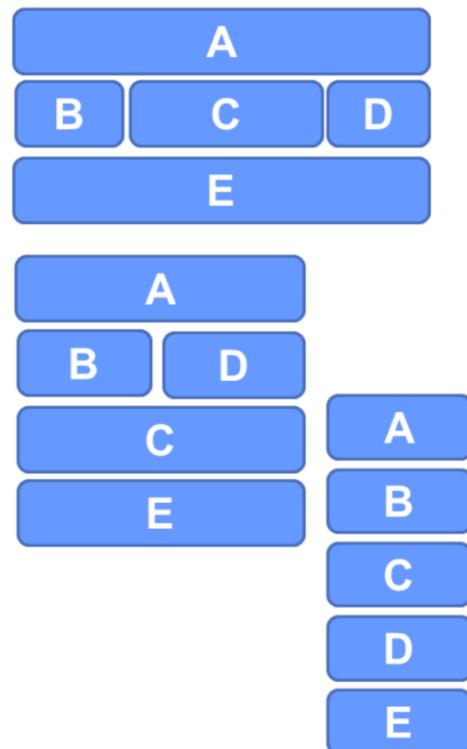
JavaScript and Responsiveness



What is Responsive Design?

What is Responsive Design?

Responsive design is the art of developing web applications to *optimize the layout* according to the user's view (screen width and height)



Responsive design is not a single, specific technology. In fact, it is the combination of several techniques (discussed next) that provide for the variations on the solution in the browser.



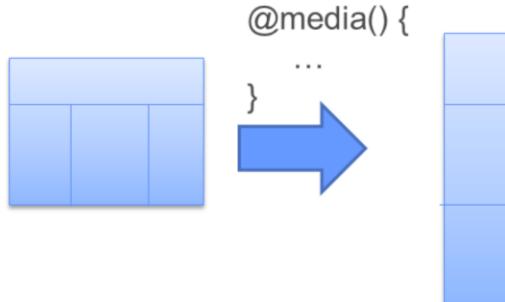
Principles of Responsive Solutions



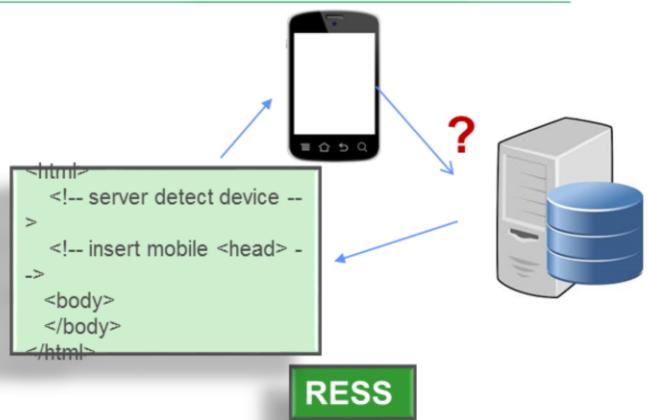
Fluid Grids



Flexible Images



Media Queries



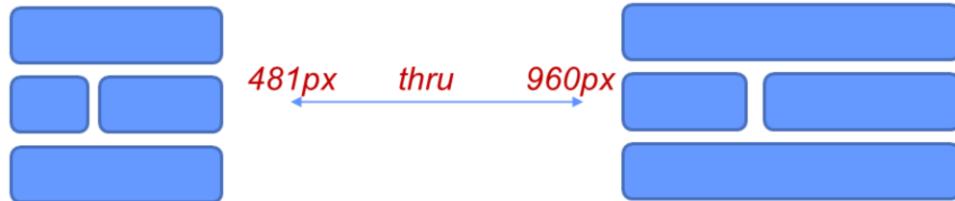
RESS

RESS = RWD + Server-side components. See slide 3 slides ahead.

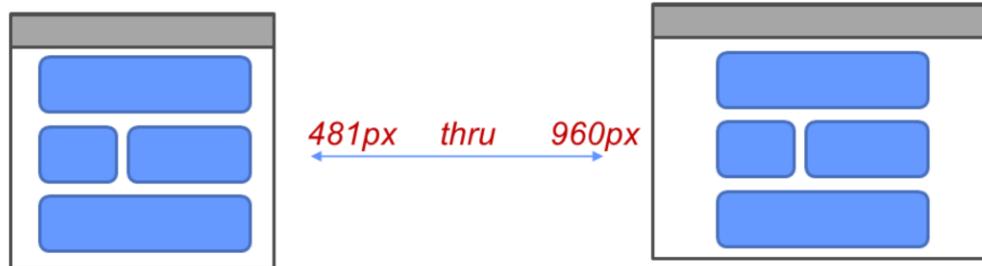


Responsive vs. Adaptive Design

- Responsive solutions have **fluid** changes
 - Can change element sizes even for a specific layout



- Adaptive solutions are **discrete** (static) changes
 - Think of it as a series of "fixed width" solutions



Both techniques will typically use media queries and support multiple layouts. The adaptive approach will "snap" into a particular layout and stay that way even as your browser resizes within the range of that layout. The example above shows what happens to each type of layout in an example (made up) range where the layout doesn't change.

Check out a further example here:
<http://liquidaptive.com/>



Responsive Design Considerations

Feature	Issue	Resolution
Tables	Desktop is okay, too wide on mobile	Transform the table. Discussed later.
Multi-columns	3-columns desktop, 1-column mobile	Use Fluid Grids
Images	Hi-res desktop, lo-res mobile	Use Flexible Images: Scalable and adaptive images
Content	Too much for mobile device users to see and read clearly	Reduce to essential content. Move other content to secondary "pages"

The 3 cornerstones to responsive solutions are: flexible grids (adjustable not fixed width columns), media queries, and flexible images (images that use CSS to resize them by specifying max-width: 100%, which causes them to fill only the size of the container).



Understanding the Viewport

- The viewport meta tag tells the browser how to scale the size of a page

```
<meta name="viewport"  
      content="user-scalable=yes, initial-scale=1.0, minimum-scale=0.5  
              maximum-scale=2.0, width=device-width" />
```

- Commonly used version:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Sets the browser to the device's width and scales it to 100%

Viewport only works on mobile devices, desktop browsers will not be affected

The width can be set to the device-width in most mobile devices now. The initial-scale determines the amount of zoom.

Check out this article for more info:

<http://dev.opera.com/articles/an-introduction-to-meta-viewport-and-viewport/>



Responsive Examples (1 of 3)

The screenshot shows a desktop view of the website. At the top, there's a navigation bar with a gear icon and the text "forefathersgroup.com". Below it is a header featuring the word "FOREFATHERS" in a stylized font with a small "ESTD 2011" below it. To the right is a teal-colored envelope icon. The main content area has a light beige background with a textured pattern. It features four grey rectangular boxes: "WORKSHOP" (The prodigy of all collective exhibitions), "SPECIALTIES" (Design of the first water is our cup of tea), "JOURNAL" (A genuine exposition of our traveling tales), and "ORIGINS" (How we gathered the force of a freight train). Below these are two rows of links: "WEBSITES • DEVELOPMENT • BRANDING • ILLUSTRATIONS". In the center, there's a large, ornate title "BEHOLD! THE DESIGN FRONTIER" with decorative scrollwork. Below the title, it says "A DESIGN COMPANY THE LIKES OF WHICH YOU'VE NEVER SEEN". To the left of the title is a black silhouette of a man in a top hat and coat, holding a cane. To the right is a small teal button labeled "Desktop".

forefathersgroup.com

The screenshot shows the same website as above, but viewed on a mobile device. The layout is much more compact. The "Desktop" button from the desktop version is now at the bottom right. A large teal button labeled "Mobile" is positioned in the center. The main content area is reduced to a single paragraph: "Witness this never before seen revelation of beauty and wonder! Enlivened by mirth, provoking surprise and marvelous feats, the Forefathers design group fashions the world's most electrifying brands, websites and illustrations." The desktop-specific sections like "WORKSHOP", "SPECIALTIES", "JOURNAL", and "ORIGINS" are no longer visible.

forefathersgroup removes unessential images when content is squeezed down.

The screenshot displays two versions of The Boston Globe website side-by-side, illustrating responsive design:

- Desktop View:** On the left, the desktop version shows a complex layout with multiple columns. It includes a header with navigation links like NEWS, METRO, ARTS, BUSINESS, SPORTS, OPINION, LIFESTYLE, MAGAZINE, and TODAY'S PAPER. Below the header are several news articles with images and captions. A sidebar on the right contains sections like "Latest news" and "Globe Insiders". A green box labeled "Desktop" highlights the main content area.
- Mobile View:** On the right, the mobile version shows a simplified layout with fewer columns. The main article is displayed prominently. A green box labeled "Mobile" highlights the main content area.

A top navigation bar at the very top includes icons for CSS and HTML5, along with the text "Responsive Examples (2 of 3)".

Notice the dynamic layout of columns based on screen resolution.

The image displays three responsive versions of the Cisco website, each labeled with a green box indicating the device type:

- Mobile:** Shows a simplified navigation menu with links for Products & Services, Support, Solutions, Partners, How to Buy, Training, Events, News, and Careers. A large "Cisco Technology Radar" graphic is prominent.
- Desktop:** Shows a more complex navigation bar with links for Products & Services, Support, Training & Events, Partners, and a search bar. It features a large "Cisco Technology Radar" graphic and a "Latest News" section.
- Tablet:** Shows a介于 mobile and desktop versions. It has a navigation bar similar to the desktop version but with a smaller "Latest News" section below it.

A small green box in the top left corner contains icons for CSS and HTML5.

The screenshot shows the Firefox browser interface. On the left, the menu bar includes 'File', 'Edit', 'View', 'History', 'Downloads', 'Add-ons', 'Options', and 'Help'. A 'Web Developer' submenu is open, listing 'Firebug', 'Toggle Tools', 'Web Console', 'Inspector', 'Debugger', 'Style Editor', 'Profiler', 'Network', 'Developer Toolbar', 'Responsive Design View' (which is highlighted with a red oval), 'Scratchpad', 'Page Source', 'Get More Tools', 'Character Encoding', and 'Work Offline'. The main content area displays a web page about the 2008 Beijing Olympics. A large blue arrow points from the 'Responsive Design View' menu option to the right, where a preview window shows the same page as it would appear on a mobile device with a resolution of 360x478 pixels.

It also rotates from portrait to landscape and simulates touch events

While not a perfect replacement for the real devices, the Responsive Design View can go a long way in providing how an application will look when applied to custom heights and widths.



Breakpoints

- **Breakpoints** are pre-defined *pixel widths* at which an application changes occur

The 2008 Summer Olympic Games, officially known as the Games of the XXIX Olympiad, is a major international multi-sport event which is being celebrated in Beijing, People's Republic of China, from August 8 (with football starting on August 6) to August 24, 2008 and followed by the 2008 Summer Paralympics from September 6 to September 17, 2008.

There are eight new sports added to the 2008 games, more than twice as many as on the schedule of the 2004 games. The 2008 Beijing Olympics will also mark the third time that Olympic events will have been held in the territories of two different National Olympic Committees (NOC), with the equestrian events to be held in Hong Kong.

The Olympic games were awarded to Beijing after an exhaustive ballot of the International Olympic Committee (IOC) on July 13, 2001. The official logo of the games, titled "Dancing Beijing," features a stylized calligraphic character jing (京, meaning capital), referencing the host city. The mascots of Beijing 2008 are the five Fuwa, each representing both a colour of the Olympic rings and a symbol of Chinese culture. The Olympic slogan, One World, One Dream, calls upon the world to unite in the Olympic spirit. Several new NOCs have also been recognised by the IOC.

The Chinese government has promoted the games to highlight China's emergence on the world stage and has invested heavily in new facilities and transportation systems. A total of 37 venues will be used to host the events including 12 newly constructed venues. Earlier in 2007, former IOC president Juan Antonio Samaranch had said that he believes that the Beijing games will be "the best in Olympic history," and current president Jacques Rogge asserts that the IOC has "absolutely no regrets" in choosing Beijing to host the 2008 games.



At 600px, this application changes its appearance

The 2008 Summer Olympic Games, officially known as the Games of the XXIX Olympiad, is a major international multi-sport event which is being celebrated in Beijing, People's Republic of China, from August 8 (with football starting on August 6) to August 24, 2008 and followed by the 2008 Summer Paralympics from September 6 to September 17, 2008.

10,500 athletes are expected to compete in 302 events across 28 sports. This is more than twice as many as on the schedule of the 2004 games. The 2008 Beijing Olympics will also mark the third time that Olympic events will have been held in the territories of two different National Olympic Committees (NOC), with the equestrian events to be held in Hong Kong.

The Olympic games were awarded to Beijing after an exhaustive ballot of the International Olympic Committee (IOC) on July 13, 2001. The official logo of the games, titled "Dancing Beijing," features a stylized calligraphic character jing (京, meaning capital), referencing the host city. The mascots of Beijing 2008 are the five Fuwa, each representing both a colour of the Olympic rings and a symbol of Chinese culture. The Olympic slogan, One World, One Dream, calls upon the world to unite in the Olympic spirit. Several new NOCs have also been recognised by the IOC.

The Chinese government has promoted the games to highlight China's emergence on the world stage and has invested heavily in new facilities and transportation systems. A total of 37 venues will be used to host the events including 12 newly constructed venues. Earlier in 2007, former IOC president Juan Antonio Samaranch had said that he believes that the Beijing games will be "the best in Olympic history," and current president Jacques Rogge asserts that the IOC has "absolutely no regrets" in choosing Beijing to host the 2008 games.

- Breakpoints are determined by the developer
 - These should be based on best look for the application, not based on common device sizes

The term is a bit confusing, but breakpoints have nothing to do with the debugging term for breakpoints. Breakpoints should not be based on common device sizes--meaning



Creating Responsive Solns

```
<div id="col-1" class="col"> The 2008 Summer Olympic Games,...</div>
<div id="col-2" class="col"> The Olympic games were ...</div>
<div id="col-3" class="col"> The Chinese government has ...</div>
```

```
<style type="text/css" media="screen">
    body { width: 100%; }
    div.col{ width: 33%; float: left; }
    #col-1 { background: #ffcccc; }
    #col-2 { background: #ccffcc; }
    #col-3 { background: #ccccff; }
    body.narrow div.col{ width: 100%; float: none; }
    body.medium div.col{ width: 50%; }
    body.medium div#col-2{ float: right; }
    body.medium div#col-3{ float: right }
</style>
```

Refer to examples/05_responsive/01_responsive_design_static_content.html



Creating Responsive Solns (cont)

```
var LayoutMgr = {
    doLayout: function() {
        var width = $(document).width();
        if(width !== this.lastWidth) {
            var bodyClass = 'narrow';
            if(width > 800)
                bodyClass = 'wide';
            else if(width >= 600 && width <= 800)
                bodyClass = 'medium';

            $('body').removeClass();
            $('body').addClass(bodyClass);
            this.lastWidth = width;
        }
    },
    lastWidth: null
};

$(window).load(LayoutMgr.doLayout);
$(window).resize(LayoutMgr.doLayout);
```

Response.js or Adapt.js are popular JavaScript libraries designed to assist in creating responsive solutions

Refer to examples/05_responsive/01_responsive_design_static_content.html



CSS 2.1 Media Types

- CSS 2.1 defined media types

all
braille
embossed
handheld
print
projection
screen
speech
tty
tv

CSS3 Media Queries

is the follow-on to CSS 2.1
media types

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

```
<link rel="stylesheet" type="text/css" href="print.css" media="print" />
```

<http://www.w3.org/TR/CSS21/media.html>



Using Media Queries

- **Media Queries** are a CSS 3 feature that allow for targeting classes of devices and obtaining device characteristics

```
<link rel="stylesheet" type="text/css"  
      media="screen and (max-device-width: 480px)" href="inner.css" />
```

- Queries can be applied to CSS elements directly

```
@media screen and (max-width: 400px) {  
    .lastChild { padding: 5%; width: 42%; }  
}
```

In the media query above, the statement checks the device's max width, if it is less than 480px, load the inner.css file. The second example applies the .lastChild class to HTML elements only if the max-width is no more than 400px (it must also be a screen-based device).



Media Query Syntax

```
<link rel="stylesheet" type="text/css"  
      media="media_type and (expression)" href="external.css" />
```

or

```
@import url(external.css) media_type and (expression);
```

or

```
@media media_type and (expression) {  
  
    /* define embedded rules this way  
  
}
```

Multiple expressions may be applied by using another 'and (expression)' after the previous one. The first and second forms above apply external css files when the media_type and expressions are met. The third syntax above assumes rules will be embedded within the @media{} braces.



Media Queries Syntax (cont.)

- In addition to **and** operations in media queries, **or** operations can be performed:

```
<link rel="stylesheet" type="text/css"
      media="(orientation: landscape), min-width:800px"
      href="enhanced.css" />
```

- Negation may also be used:

```
@media not print and (orientation: portrait) {
    div#main {
        color: blue;
    }
}
```

The not and or operators are less commonly used.



Media Query Expressions Properties

- All of these are valid expression variables for media queries:

width
height
device-width
device-height
orientation
device-aspect-ratio

color
color-index
monochrome
resolution
scan
aspect-ratio
grid

hover
luminosity
hover
pointer
script
touch
types

Low support,
these are a
part of *Media
Queries Level
4 Spec
(coming)*

```
@media screen and (min-width: 180px) and (max-width: 480px)
  and (color) {
  .secondary_col { display: none; }
}
```

Test your browser here:

<http://www.quirksmode.org/css/tests/mediaqueries/>

width, height, device-width, device-height, color, resolution all support a min- and max-prefix as well.

orientation can be 'portrait' or 'landscape'

color reports whether or not it supports color or how many bits depth color is supported.
(e.g. @media all and (color) {....})

resolution supports min-resolution, max-resolution. Units for this can be dpi (dots per inch)

CSS Media Queries Level 4 Spec: <http://dev.w3.org/csswg/mediaqueries4/>



Media Query Device Support

CSS3 Media Queries - **Recommendation**

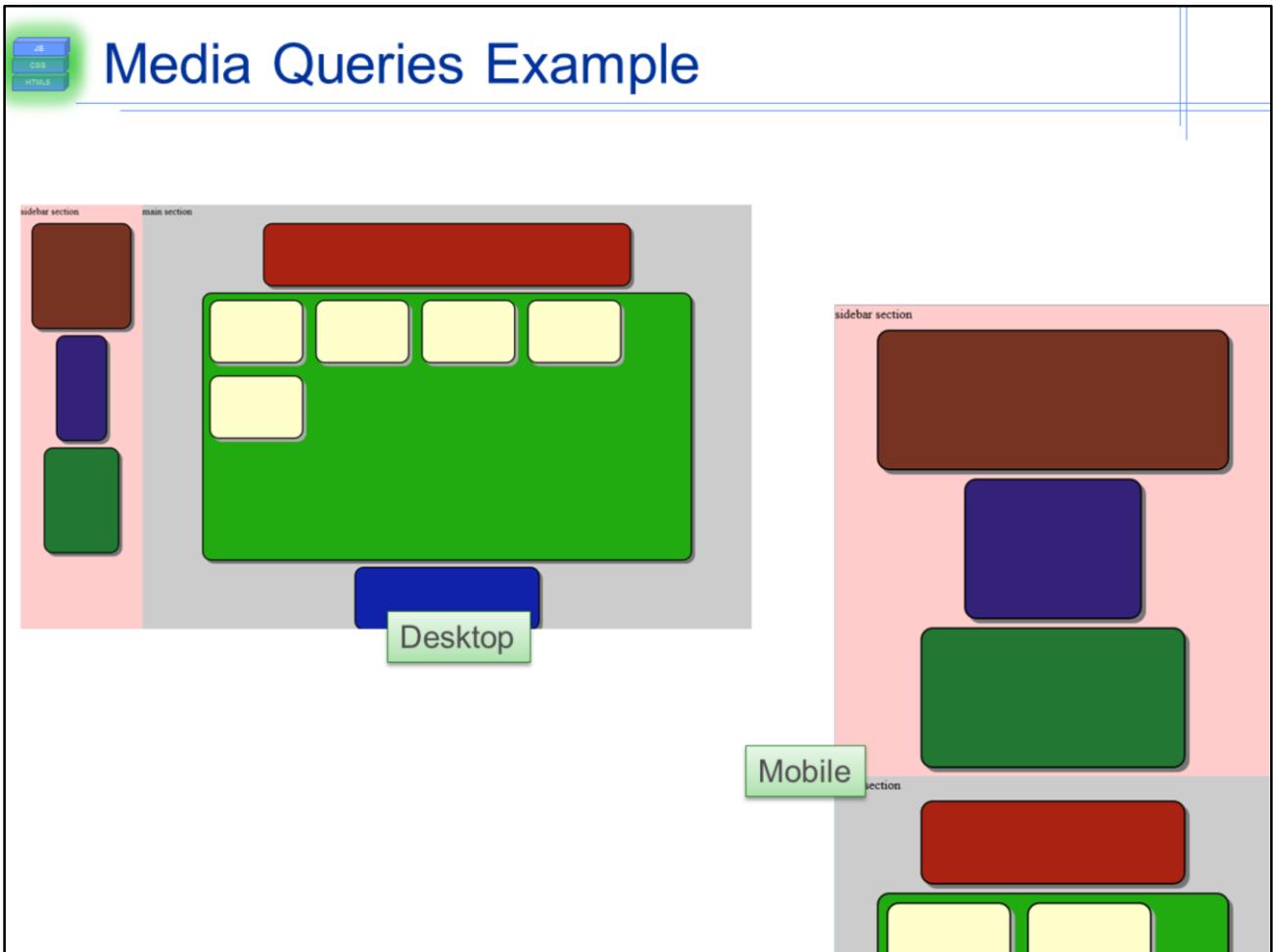
Method of applying styles based on media information. Includes things like page and device dimensions

								*Usage stats:	Global
								Support:	78.34%
								Partial support:	0.02%
								Total:	78.36%
Show all versions	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Opera Mobile	Android Browser
						3.2			2.1
	7.0	3.6				4.0-4.1		10.0	2.3
	8.0	12.0				4.2-4.3		11.5	3.0
Current	9.0	13.0	19.0	5.1	12.0	5.0	5.0-6.0	12.0	4.0
Near future	10.0	14.0	20.0	5.2					
Farther future		15.0	21.0						

[Notes](#) [Known issues \(1\)](#) [Resources \(4\)](#) [Feedback](#)

Incomplete support by older webkit browsers refers to only acknowledging different media rules on page reload

Media queries are supported by all modern devices. Only legacy IE browsers will not. However, this is generally okay as IE legacy browsers are usually run within standard desktop/laptop devices which are often the default implementations that do not require a media query-based solution.



examples/05_responsive/02_media_queries.html



Media Queries Example

```
#wrapper {  
    width: 80%; margin: 10px auto;  
    max-width: 1200px;  
    min-width: 600px;  
    background-color: #fcc;  
}  
  
#main {  
    margin-left: 200px;  
    background-color: #ccc;  
}  
  
#sidebar { width: 200px;  
    float: left;  
    background-color: #fcc;  
}
```

```
@media screen  
and (max-width: 600px) {  
    #wrapper {  
        width: 570px;  
        margin: 0 auto;  
    }  
    #main {  
        margin-left: 0;  
        width: 90%;  
    }  
    #sidebar {  
        width: 90%; float: none;  
    }  
}
```

[examples/05_responsive/02_media_queries.html](#)



Media Queries Rules

```

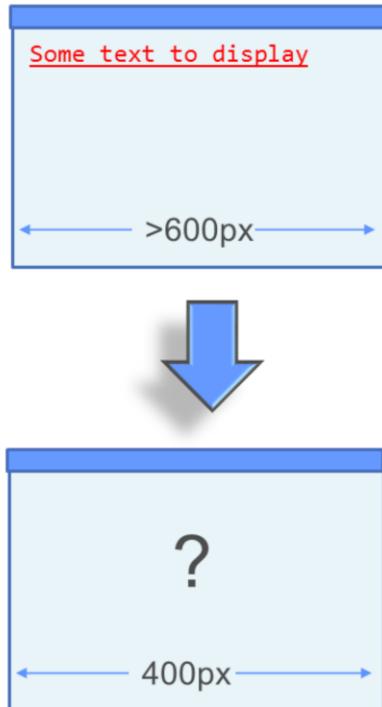
article {
    text-decoration: underline;
}

article.lead {
    color: red;
}

@media screen and (max-width: 600px) {
    .lead {
        color: blue;
    }
}

<article class="lead">
    Some text to display
</article>

```



[examples/05_responsive/03_media_queries_rules.html](#)

The media query MUST follow the same rules of CSS. In this case, when the browser is 1000px, the default CSS kicks-in but the media query doesn't get evaluated. Therefore, the text is rendered in red and underlined. When the device is less than 600px, the media query is valid. But the results are perhaps not what you think. The default CSS is still valid, therefore, the article.lead selector is still more specific than the .lead selector within the query. Therefore the color red will override the color blue.



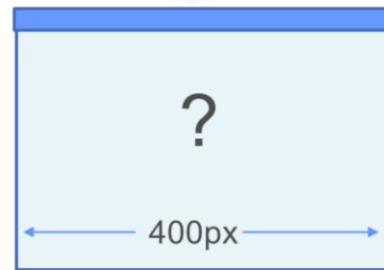
Media Queries Rules - Revised

```
article {  
    text-decoration: underline;  
}  
  
.Lead {  
    color: red;  
}  
  
@media screen and (max-width: 600px) {  
    .Lead {  
        color: blue;  
    }  
}  
  
<article class="Lead">  
    Some text to display  
</article>
```

The article element name was removed

Some text to display

<-- >600px -->



examples/05_responsive/03b_media_queries_rules.html

Here the value changes to blue because both .lead rules are evenly weighted + the .lead rule came later in the document.



Media Queries - Mobile First Approach

- A re-working of the CSS emphasizes the mobile-based implementation instead

```
<link rel="stylesheet" type="text/css" href="main.css"  
media="screen, handheld" />
```

Loads for all device types

```
<link rel="stylesheet" type="text/css" href="enhanced.css"  
media="screen and (min-width: 40.5em)" />
```

This loads in addition to the
above for wide screens

While it may seem subtle, the default CSS will now apply to all versions, while only enhanced stuff will be loaded in the desktop environment.



Mobile-First Methodology

```
.Lead {  
  color: blue;  
}  
  
@media screen and (min-width: 600px) {  
  .Lead {  
    color: red;  
  }  
}
```

Our re-worked version now loads all versions by default

Media Queries "kick-in" only for the larger screens

[examples/05_responsive/04_media_queries_mobile_first.html](#)

The change in emphasis is subtle but loads default values for all devices first, then layers CSS for additional devices on top of this.



More Media Queries

```

nav ul li span {
  background-color: #ffc73d;
  border: 1px solid #aa5522;
  padding: 5px;
  display: block;
}
.products li {
  float: left;
  width: 50%;
}
@media screen and (min-width: 600px) {
  .products li {
    width: 33.3333332%;
  }
}
@media screen and (min-width: 800px) {
  .products li {
    width: 16.6666667%;
  }
}

```

```

<nav id="main-nav">
  <ul class="products">
    <li><span>Item 1</span></li>
    <li><span>Item 2</span></li>
    <li><span>Item 3</span></li>
    <li><span>Item 4</span></li>
    <li><span>Item 5</span></li>
    <li><span>Item 6</span></li>
  </ul>
</nav>

```

Item 1	Item 2
Item 3	Item 4
Item 5	Item 6

Item 1	Item 2	Item 3
Item 4	Item 5	Item 6

Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
--------	--------	--------	--------	--------	--------

examples/05_responsive/05_media_queries_evolving.html

This version shows 3 different layouts that change at the 600 and 800px breakpoint levels. The media query has the capability to completely change the look of an app short of actually changing the DOM.



Media Queries in JavaScript

- To perform media-queries dynamically within JavaScript use `window.matchMedia()`



`matchMedia()` is a way to perform Media Query checks from within your JavaScript. It works the same way the CSS except within JavaScript.



Patterns for Media Queries (1 of 3)

- While any combination of patterns can be created using media queries, several *patterns* for layouts have evolved

The following site provides the inspiration for these media query patterns:

<http://mediaqueri.es/>

Let's examine these in more detail...

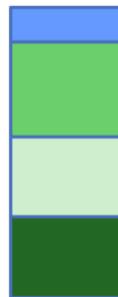
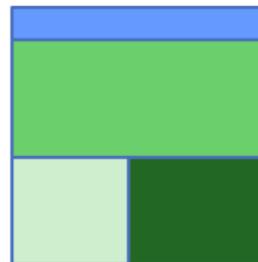
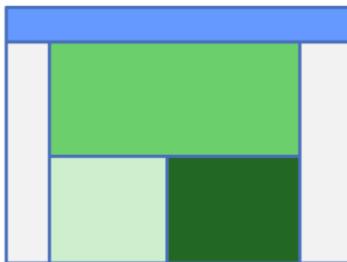
The patterns mentioned here give ideas on how sites can change responsively. The patterns here were named by Luke Wroblewski, author of the recommended book: "Mobile First" development. These patterns are listed in order of popularity.

For the following patterns, the site: <http://mediaqueri.es> was used as a reference.

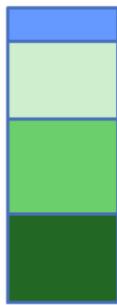
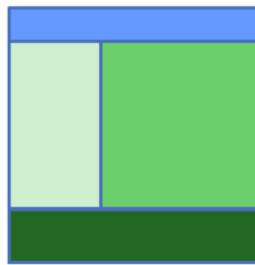
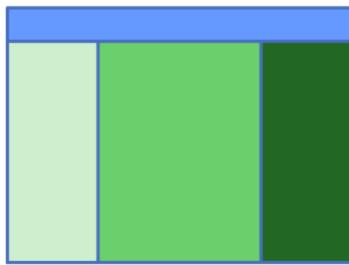


Patterns for Media Queries (2 of 3)

- Mostly Fluid



- Column Drop

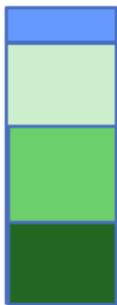
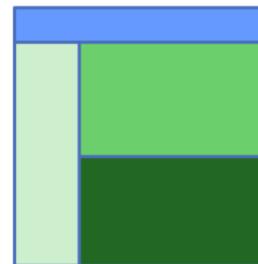


The mostly fluid solution is the most popular and one of the easiest to implement. It simply changes sizes at the lowest device width's into a stacked container.

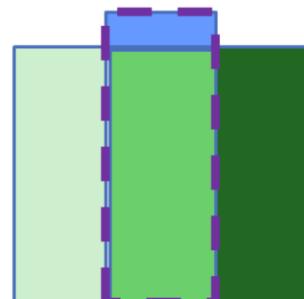
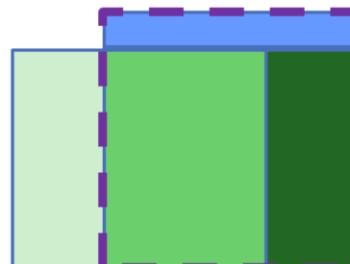


Patterns for Media Queries (3 of 3)

- Layout Shifter



- Off Canvas



An explanation might be required for the off canvas pattern. When a button is tapped in the smaller sizes the off-screen piece slides into view. The example below is taken from: <http://www.kaemingk.com/en/our-showroom/christmas/>



Exercise 6 - It's Your Turn

Responsive Solutions

Adv. JavaScript and HTML5

Contact Search							
Contact ID:	501						
Name:	Bob Green						
Address:	2101 Eucalyptus Ave. Philadelphia PA 20119						
Primary Phone:	(202)901-2121						
Type:	home						
Email:	asdf						
Company:	Tupelo Industries						
Job Title:	asdf						
<input type="button" value="Create"/> <input type="button" value="Update"/> <input type="button" value="Delete"/>							
Contact Id	Name	Address	Phone Num	Phone Type	Email	Company	Job Title
500	Red Vectors 4517 Elm St. Riverside NJ 08075	(301)306-8921 home	t.thompson@yahoo.com	ABC Inc.	President		
501	Bob Green 2101 Eucalyptus (202)901-2121 home asdf			Tupelo Industries	asdf		
502	John Brown asdf (719)421-8875 home jb6712@gmail.com			Last Rites LLC	Undertaker		
503	Tina O. Range 82 Pine Dr. Lakewood CA 90712	(212)432-0944 home	tomut@besthp.com	Best Holistic Customer Practices Service			

Adv. JavaScript and HTML5

Contact Search							
Contact ID:	501						
Name:	Bob Green						
Address:	2101 Eucalyptus Ave. Philadelphia PA 20119						
Primary Phone:	(202)901-2121						
Type:	home						
Email:	asdf						
Company:	Tupelo Industries						
Job Title:	asdf						
<input type="button" value="Create"/> <input type="button" value="Update"/> <input type="button" value="Delete"/>							
Contact Id	Name	Address	Phone Num	Phone Type	Email	Company	Job Title
500	Red Vectors 4517 Elm St. Riverside NJ 08075	(301)306-8921 home	t.thompson@yahoo.com	ABC Inc.	President		
501	Bob Green 2101 Eucalyptus (202)901-2121 home asdf			Tupelo Industries	asdf		
502	John Brown asdf (719)421-8875 home jb6712@gmail.com			Last Rites LLC	Undertaker		
503	Tina O. Range 82 Pine Dr. Lakewood CA 90712	(212)432-0944 home	tomut@besthp.com	Best Holistic Customer Practices Service			

Instructions for this task are found
in labs/lab06/starter/index.html



Handling Tables

- Responsive data tables are a problem because they occupy too much space

100	Fritz	Thompson	76
101	Eric	Colbert	55
102	Stephen	Cheadle	82
103	Anne	Mouvier	61
104	Amy	Hilbert	47
105	Ester	Freeman	93
106	Paul	Cruthers	81
107	Brian	Duvall	53
108	Charles	Simpson	87
109	Juan	Figueroa	61

Horizontally scrolling
the page versus
text that is too small
NEITHER IS IDEAL!

100	Fritz	Thompson	76500
101	Eric	Colbert	55000
102	Stephen	Cheadle	82600
103	Anne	Mouvier	61400
104	Amy	Hilbert	47320
105	Ester	Freeman	93700
106	Paul	Cruthers	81000
107	Brian	Duvall	53200
108	Charles	Simpson	87600
109	Juan	Figueroa	61700

Handling tables can be tricky. There is no ideal solution. Scrolling tables to the side is an option for dealing with the width while making the text so small to make them fit is another (allowing pinching to expand the text).



A Vertical Table

- One approach toward handling tables is to convert them to behave vertically in a narrow device

Contact Id	Name	Address	Phone Num	Phone Type	Email	Company	Job Title
500	Red Victors	4517 Elm St. Riverside NJ 08075	(301)356-8921	home	ftthompson@yahoo.com	ABC Inc.	President
501	Bob Green	2101 Eucalyptus Ave. Philadelphia PA 09119	(202)901-2121	home	dbreal@hotmail.com	Tupelo Industries	
502	John Brown	331 Birch Cir. Black Hills SD 82101	(719)421-8875	home	jb6712@gmail.com	Last Rites LLC	
503	Tina O. Range	82 Pine Dr. Lakewood CA 90713	(212)432-0944	home	tormut@besthp.com	Best Holistic Practices	
504	Berry Blumenthal	3012 Mahogany Ln. Denver CO 80101	(202)685-2323	home	bleubry@yahoo.com	Roller Heights Pack	
505	Jim Pinkado	2113 Redwood Blvd. Long Beach CA 90314	(204)740-9000	work	jpinkman@rocketmail.com	Hollywood West No	
506	Alicia Grey	415 Poplar Ct. St. Louis MO 72210	(211)870-6780	home	algrey@blanksystems.com	Blank Systems Inc.	
507	Violet Waters	821 Ash Way Seattle WA 92230	(302)390-1181	home	waters@medcare.com	Home Medical Serv	
508	Sandy White	906 Hickory Rd. Phoenix, AZ 83010	(213)221-4143	home	swhite@bricks.com	Bricks and More	
509	Kay Black	1241 Maple Pl. Plano TX 72110	(401)322-8728	home	ksb2101@yahoo.com	Certified Signing Au	



Contact ID	500
Name	Red Victors
Address	4517 Elm St. Riverside NJ 08075
Phone Num	(301)356-8921
Phone Type	home
Email	ftthompson@yahoo.com
Company	ABC Inc.
Position	President
Contact ID	501
Name	Bob Green
Address	2101 Eucalyptus Ave. Philadelphia PA 09119
Phone Num	(202)901-2121
Phone Type	home
Email	dbreal@hotmail.com
Company	Tupelo Industries
Position	Product Manager
Contact ID	502
Name	John Brown
Address	331 Birch Cir. Black Hills SD 82101
Phone Num	(719)421-8875
Phone Type	home
Email	jb6712@gmail.com
Company	Last Rites LLC
Position	Undertaker
Contact ID	503
Name	Tina O. Range
Address	82 Pine Dr. Lakewood CA 90713
Phone Num	(212)432-0944
Phone Type	home
Email	tormut@besthp.com
Company	Best Holistic Practices
Position	Customer Service Representative
Contact ID	504
Name	Berry Blumenthal
Address	3012 Mahogany Ln. Denver CO 80101

examples/05_responsive/08_tables.html



Vertical Table CSS

```
@media screen and (max-width: 768px) {
  table, th, td, tr { display: block; }
  tr { border: 1px solid #ccc; }
  tr th { display: none; }
```

```
td {
  border: none;
  border-bottom: 1px solid #ccc;
  position: relative;
  padding-left: 50%; }
```

```
td:before {
  position: absolute;
  top: 0;
  left: 6px;
  width: 45%;
  white-space: nowrap; }
```

Hide the header row

Put a line between each row

Position the 'before' content

This is non-semantic, but allows us to present each data item with a header name

```
td:nth-of-type(1):before { content: "Contact ID"; }
td:nth-of-type(2):before { content: "Name"; }
td:nth-of-type(3):before { content: "Address"; }
td:nth-of-type(4):before { content: "Phone Num"; }
td:nth-of-type(5):before { content: "Phone Type"; }
td:nth-of-type(6):before { content: "Email"; }
td:nth-of-type(7):before { content: "Company"; }
td:nth-of-type(8):before { content: "Position"; }
```

[examples/05_responsive/08_tables.html](#)



Exercise 6b - It's Your Turn

Responsive Tables

Adv. JavaScript and HTML5

Contact Search

Contact ID	501
Name	Bob Green
Address	2101 Eucalyptus Ave. Philadelphia
Primary Phone	(202)901-2121
Type	home
Email	asf
Company	Tupelo Industries
Job Title	asdf
<input type="button" value="Create"/> <input type="button" value="Update"/> <input type="button" value="Delete"/>	

Contact	Name	Address	Phone Num	Phone Type	Email	Company	Job Title
500	Red Vectors	4517 Elm St. Riverside NJ 08075	(301)356-8921	home	thompson@yahoo.com	ABC Inc.	President
501	Bob Green	2101 Eucalyptus Ave. Philadelphia PA 09119	(202)901-2121	home	asf	Tupelo Industries	asdf
502	John Brown	42 Pine Dr. Lakewood CA	(719)421-8875	home	j6712@gmail.com	Last Rites LLC	Undertaker
503	Tina O. Range	82 Pine Dr. Lakewood CA	(212)432-0944	home	tomut@besthp.com	Best Holistic Customer Practices	Service

Adv. JavaScript and HTML5

Contact Search

Contact ID	500
Name	Red Vectors
Address	4517 Elm St. Riverside NJ 08075
Phone Num	(301)356-8921
Phone Type	home
Email	thompson@yahoo.com
Company	ABC Inc.
Position	President

Contact ID	501
Name	Bob Green
Address	2101 Eucalyptus Ave. Philadelphia PA 09119
Phone Num	(202)901-2121
Phone Type	home
Email	asf
Company	Tupelo Industries
Position	Product Manager

Contact ID	514
Name	Red Vectors
Address	4517 Elm St. Riverside NJ 08075
Phone Num	(301)356-8921
Phone Type	home
Email	thompson@yahoo.com
Company	ABC Inc.
Position	President

Contact ID	
Name	
Address	
Primary Phone	
Type	
Email	
Company	
Job Title	

Instructions for this task are found
in labs/lab06b/starter/index.html

Chapter 6

Responsive Typography and Images

Dealing with images and fonts in a
responsive world



Overview

Unit Systems

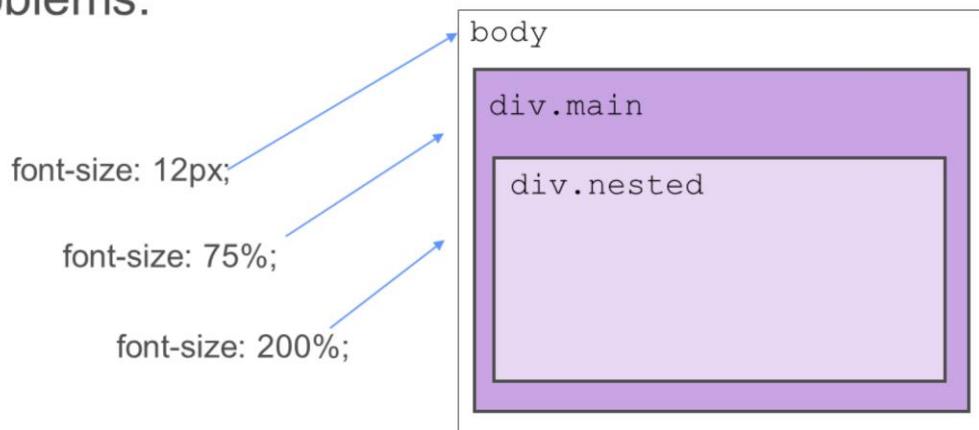
Creating Responsive Text

Responsive Images



Problems with relative units: em, %

- Generally use relative fonts (percent, ems) when building scalable solutions
- However, these unit systems are not without problems:



What is the font-size of `div.nested`?

[examples/06_typography/01_relative_unit_issues.html](#)

Answer: 18px.

Relative fonts are always relative to their parent element. This can be tricky to calculate and track as your solution becomes increasingly complex.



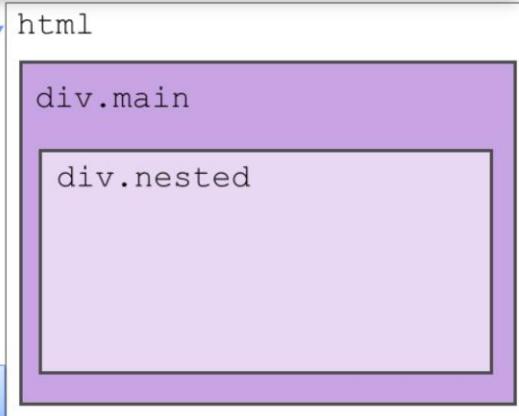
Solving this problem with REMs

- A **REM** is a "root-em"
 - All REM values are with respect to the root element



font-size: 12px;
font-size: 0.75rem;
font-size: 2rem;

What is the font-size of div.nested?



[examples/06_typography/02_rems.html](#)

Answer: 24px

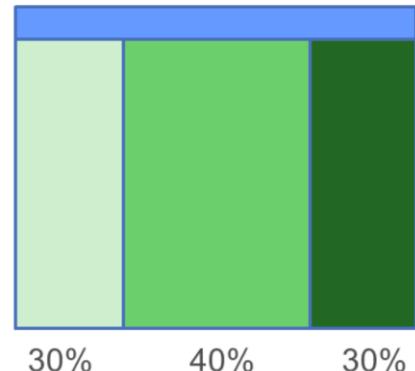
The root-em is a way to use relative fonts, but the root-em value is not inherited as an em value is.



Additional Problems with Relative Units

- Consider the following issue:

If you want to **add a fixed 10px padding**, this causes the total to grow beyond 100%



Mixing fixed + relative units is tricky!

Here we've defined a nice fluid layout at 30-40-30 percent. If you want to add a 10px padding to any or all columns, this will cause the columns to grow beyond 100%, thus ruining the solution. So, how do you do it? Do you convert the fixed 10px to a percentage? If so, what percentage do you use? The bottom line is that fixed units and percentages don't mix too well.



Viewport-Relative Units

- Rather than relying on a value relative to a parent element, it might make sense to rely on a unit relative to the viewport



```
div.nested {
    font-size: 10vh;
    width: 50vw;
}
```

50% of the viewport width

10% of the viewport height

[examples/06_typography/03_viewport_units.html](#)

Viewport-relative units are not in wide use yet. IE browsers do not properly support them and even numerous mobile devices have spotty support.



Making Text Responsive

- Viewport-relative units are not widely supported yet
 - Text can be made scalable still:

```
html{font-size:18px;}
```

```
@media screen and (orientation:portrait){  
    div.nested{ font-size: 2rem; }  
}  
@media screen and (orientation:landscape){  
    div.nested{ font-size: 1rem; }  
}
```

Text size changes upon orientation changes



[examples/06_typography/04_scaling_fonts.html](#)



Making Images Flexible

- Images can be **scaled** or **adapted**
- *Scaling* involves controlling the viewport and max-width of the image
- *Adaptive* images will download a different image depending on the device size
 - Usually controlled through a media-query



Scaling Images Fluidly

- Use **max-width** on an image using a relative unit, parent is allowed to scale with viewport

```
<article id="main-article">
  <p class="article-info">
    In Amsterdam, ...
  </p>
  <section class="details" >
    
    <p>A Sea of Bikes</p>
  </section>
</article>
```

In Amsterdam, bikes outnumber cars! You must take extra precaution when walking—paying particular attention to oncoming traffic. But this traffic is found on the sidewalks and bike paths!



Keeps image from expanding past the parent. Height: auto keeps the aspect-ratio constant

```
.article-info {
  float: left;
  width: 47%;
  padding: 5px;
}

section.details {
  float: left;
  width: 50%;
}

img.scale {
  max-width: 100%;
  height: auto;
}
```

examples/06_typography/05_responsive_images.html

In this example, the image is allowed to scale with the parent by fixing its size to its parent but allowing the parent to change sizes depending on the device size.



Fluid and Adaptive Images

- Frameworks exist to help load adaptive images
- ResponseJS** can manipulate HTML or images
 - jQuery-based framework
 - Simply add custom attributes to HTML elements

```
<body data-responsejs='{ "create": [ { "breakpoints": [641],  
          "mode": "src", "prefix": "src" } ]}'>  
  <article id="main-article">  
    <p class="article-info">...</p>  
    <section class="details" >  
        
      <p>A Sea of Bikes</p>  
    </section>  
  </article>  
  <script src="../../libs/response.js"></script>  
</body>
```

Loads hi-res
image if
>640 size is
detected

examples/06_typography/06_adaptive_and_scalable_images.html

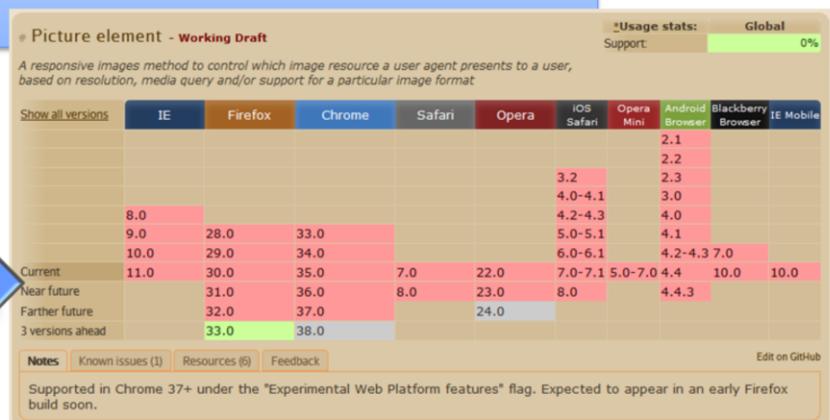
This example uses a JavaScript framework to load images by simply marking them with custom tags. Placing a bootstrap tag in the body and data-srcXXX tags in the tags causes the library to automatically load the higher resolution image if / when the test becomes valid (true).



CSS Picture Elements

- Native support for adaptive images is coming...

```
<picture width="500" height="500">
  <source media="(min-width: 40.5em)" src="hi-res.jpg">
  <source media="(min-width: 20em)" src="med-res.jpg">
  <source src="lo-res.jpg">
  
  <p>Picture of bicycles</p>
</picture>
```



But not yet!

The spec is in working draft now: <http://www.w3.org/TR/html-picture-element/>

But essentially this will remove the need to have a ResponseJS framework.



Object-fit

- Object-fit is new, but will scale images within an HTML element



```
img.scale {
  width: 100%;
  height: 100%;
  object-fit: contain;
}
```



article (parent) ↗

```
img.scale {
  width: 100%;
  height: 100%;
  object-fit: fill;
}
```



```
<article id="main-article">
  
</article>
```

```
img.scale {
  width: 100%;
  height: 100%;
  object-fit:
  cover;
}
```



[examples/06_typography/08_object_fit.html](#)

Object-fit is similar to the new features added to the background-image property in CSS3 except that these apply to `` images. There are relatively new, but provide some control over how images scale. Using 'contain', the image will maintain its aspect-ratio, but perhaps not fill the entire parent. Using 'fill', the image will alter the aspect-ratio to fill the parent. Using 'cover', will also maintain the aspect-ratio, but will enlarge until the small dimension (either height or width) fills the parent. In this case the image enlarged until the width was completely covered. This causes clipping on the image. By default the object is centered. Control the positioning with object-position.



text-size-adjust

- **text-size-adjust** (coming soon) determines if text should enlarge and if so, by how much

```
/* says text on iPhone should not be adjusted, e.g. in an orientation change */
p { -webkit-text-size-adjust: none; }
```

```
/* says adjust text size downward by half */
p { -moz-text-size-adjust: 50%; }
```

This property currently requires vendor extensions (until it matures in the standards)



[examples/06_typography/07_text_size_adjust.html](#)

This property is not standard yet. Here is its current working draft:

<http://dev.w3.org/csswg/css-size-adjust/>

More on the property:

<https://developer.mozilla.org/en-US/docs/Web/CSS/text-size-adjust>



Summary

- Use height:auto, max-width: 100% to scale images fluidly
- Scale fonts using relative units
 - Choose remss to scale according to the root
 - Viewport-relative units are a bit early to use
- Use adaptive image frameworks to serve low-bandwidth images
- Be aware of new properties coming: object-fit, text-adjust-size, and <picture> elements
 - But we can't use these just yet!

Chapter 7

Fluid Layouts and Grid Systems

Implementing Layouts using Pre-built
Frameworks



Overview

Creating a Fluid Grid Layout

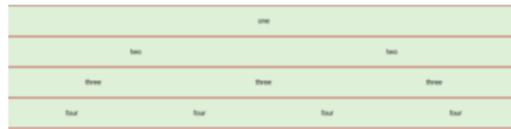
Using a CSS Grid Framework



Building a Fluid Grid

- A flexible grid provides a way for columns to change sizes under different device sizes

```
[class*='col-'] { float: left; }
[class*='col-']:last-of-type { float: right; }
.grid:after {
  content: "";
  display: table;
  clear: both;
}
.col-1 {width: 8.33333%;}
.col-2 {width: 16.6666667%;}
.col-3 {width: 25%;}
.col-4 {width: 33.333333%;}
...
.col-12 {width: 100%;}
```



```
<div id="row1" class="grid">
  <div class="col-12">one</div>
</div>
<div id="row2" class="grid">
  <div class="col-6">two</div>
  <div class="col-6">two</div>
</div>
<div id="row3" class="grid">
  <div class="col-4">three</div>
  <div class="col-4">three</div>
  <div class="col-4">three-</div>
</div>
<div id="row4" class="grid">
  <div class="col-3">four</div>
  <div class="col-3">four</div>
  <div class="col-3">four</div>
  <div class="col-3">four</div>
</div>
```

examples/07_fluid_layouts/01_flex_grid.html

This example shows that it is relatively easy to build a grid system.

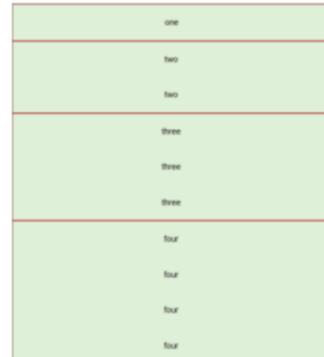


Fluid Grids + Media Queries

- Our fluid grid can also be controlled by media queries



> 480px



< 480px

```
@media screen and (max-width: 480px) {  
    .col-1, .col-2, .col-3, .col-4, .col-5, .col-6,  
    .col-7, .col-8, .col-9, .col-10, .col-11, .col-12 { width: 100%; }  
}
```

examples/07_fluid_layouts/01_flex_grid.html

Our grid solution breaks at 480px. Note: this is not a mobile-first implementation.



Flexbox

- Flex-box is a CSS-based way to lay out pages using giving high-level elements resizing capabilities

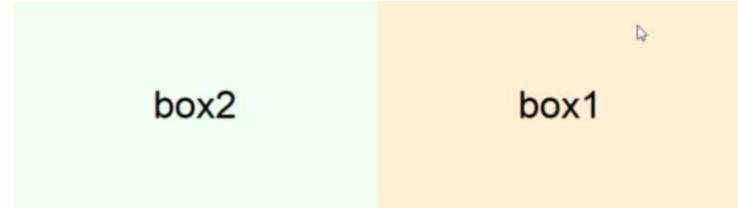


```
#box1, #box2 {
  width: 50%;
  height: 200px;
}

#box1 { background-color: papayawhip; }
#box2 { background-color: honeydew; }

#flexcontainer {
  display: flex;
  flex-direction: row-reverse;
}
```

examples/07_fluid_layouts/02_flexbox.html



```
<div id="flexcontainer">
  <div id="box1">
    box1
  </div>
  <div id="box2">
    box2
  </div>
</div>
```

Notice that the flex-direction can actually change the order of the boxes from the DOM! Legal values include row, row-reverse, column, column-reverse.

Other flexbox properties include the ability to completely re-order the columns (use the flex-order property on the child elements supplying an ordinal value).

Another property, called justify-content can align content along the main axis (the direction boxes are placed, such as row or column) or the cross-axis, which is the opposite axis.

The justify-content can be either: flex-start, center, space-around which place boxes at the axis beginning, center, or evenly justified.

The align-items property can place justify items on the cross-axis using: stretch (fills the height of the parent), center (centers within the parent), and flex-end (bottom-aligns on the parent). You may also align items individually using align-self.

Furthermore, the ultimate in flexibility comes from the flex property which is a shorthand for three properties: flex-grow, flex-shrink, and flex-basis. The flex-basis value takes precedence over the width property that the browser applies to the box. Flex-grow and flex-shrink represent a ratio (of the value of all children) for how to grow and shrink. An example flex property might look like this: flex: 1 2 200px; More on this can be found in a

second flex example in the student files.



Layout Frameworks

- CSS Frameworks can provide a faster time-to-production approach than building your own grid system
 - Provide legacy browser hacks
 - Often provide other responsive components
- Twitter's **Bootstrap** is a popular mobile-first, fluid grid layout



Bootstrap Template

```
<!DOCTYPE html>
<html>
  <head>
    <title>Bootstrap Template</title>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link href="../../libs/css/bootstrap/dist/css/bootstrap.css"
          rel="stylesheet" media="screen">

    <!--[if lt IE 9]>
      <script src="../../libs/css/bootstrap/assets/js/html5shiv.js"></script>
      <script src="../../libs/css/bootstrap/assets/js/respond.min.js"></script>
    <![endif]-->

    <script src="../../libs/jquery.js"></script>
    <script src="../../libs/css/bootstrap/dist/js/bootstrap.js"></script>

  </head>
  <body>

    </body>
</html>
```

examples/07_fluid_layouts/04_bootstrap.html



Bootstrap Class Names

- Bootstrap provides a **12-column grid** that can break at different predefined breakpoints
 - Its behavior is affected by the class names you use:

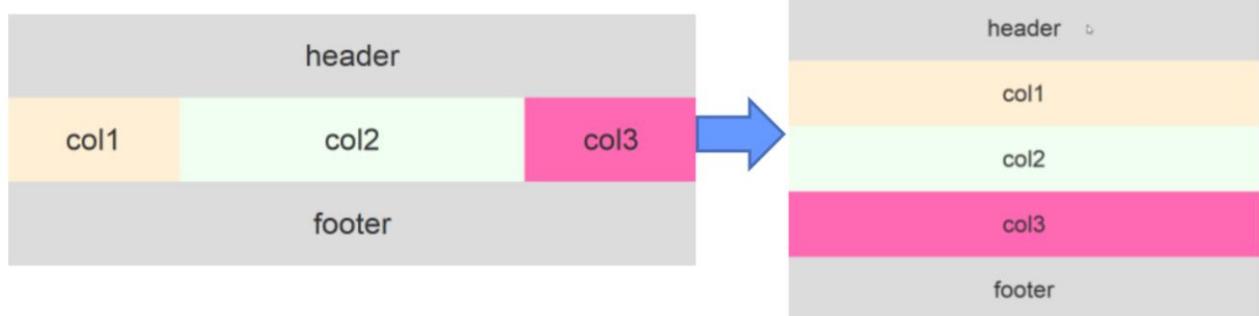
	Extra Small	Small	Medium	Large Devices
Breakpoint	< 768px	> 768px	> 992px	>1200px
Class Name	.col-xs-	.col-sm-	.col-md-	.col-lg-

To create 3 even columns in a 12-column grid that breaks at 992px use:

```
<div class="container">
  <div class="row">
    <div class="col-md-4"></div>
    <div class="col-md-4"></div>
    <div class="col-md-4"></div>
  </div>
</div>
```



Bootstrap Layout Example 1

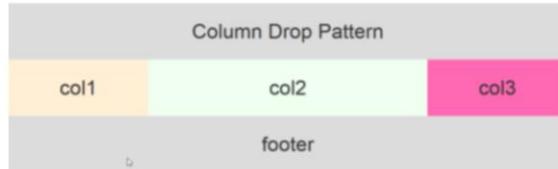


```
<div id="container" class="container">
  <header class="col-md-12">header</header>
  <section class="row">
    <article id="col1" class="col-md-3">col1</article>
    <article id="col2" class="col-md-6">col2</article>
    <article id="col3" class="col-md-3">col3</article>
  </section>
  <footer class="col-md-12">footer</footer>
</div>
```

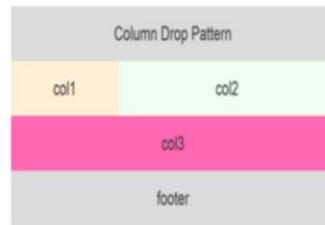
examples/07_fluid_layouts/04_bootstrap.html



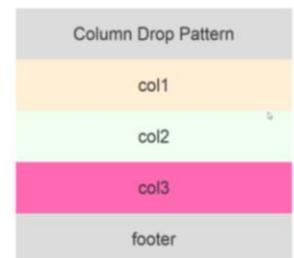
Bootstrap Layout Example 2



> 1200px



768-1200px



< 768px

```
<div id="container" class="container">
  <header class="col-md-12">header</header>
  <section class="row">
    <article id="col1" class="col-md-4 col-lg-3">col1</article>
    <article id="col2" class="col-md-8 col-lg-6">col2</article>
    <article id="col3" class="col-md-12 col-lg-3">col3</article>
  </section>
  <footer class="col-md-12">footer</footer>
</div>
```

examples/07_fluid_layouts/05_bootstrap_layout2.html



Bootstrap's Numerous Classes

- Bootstrap offers *numerous classes* to provide a look-and-feel. These are only a few:

lead	list-inline	img-responsive	btn-info	checkbox-inline
text-left	list-unstyled	help-block	btn-danger	control-label
text-center	table	active	btn-success	has-warning
text-right	table-bordered	success	btn-warning	has-error
text-justify	table-hover	info	btn-lg	has-success
text nowrap	table-condensed	warning	btn-sm	input-lg
text-lowercase	table-responsive	danger	btn-xs	input-sm
text-uppercase	form-group	btn	btn-md	input-md
text-capitalize	form-inline	btn-block	radio	input-xs
initialism	form-control	btn-default	radio-inline	disabled
	form-horizontal	btn-primary	checkbox	

- Bootstrap is compiled using LESS
 - These variables are available:

```
@grid-columns: 12;
@grid-gutter-width: 30px;
@grid-float-breakpoint: 768px;
```

Bootstrap provides dozens of classes and LESS functions and variables that can help customize the bootstrap environment. It can be a very powerful tool when combined with LESS (discussed later).



Summary

- Building your own fluid grid is not difficult
 - Resulting solution is your own, no external dependencies
- Bootstrap provides a huge number of features yet comes at a small cost
 - It can save a huge amount of time not having to create your own solutions



Exercise 7 - It's Your Turn

Implement the Column Drop Pattern with Bootstrap

Work from labs/lab07/starter/index.html.

Two steps:

1. Add bootstrap to the implementation
2. Set the columns on leading, trailing, main divs to achieve the effect

Desktop View:

Contact ID	Name	Address	Phone Num	Phone Type	Email	Company	Job Title
500	Red Victors	4517 Elm St, Riverside NJ 08075	(301)356-8921	home	fthompson@yahoo.com	ABC Inc.	President
501	Bob Green	2101 Eucalyptus Ave, Philadelphia PA 09119	(202)901-2121	home	dbreal@hotmail.com	Tupelo Industries	Product Manager
502	John Brown	331 Birch Cir, Black Hills SD 82101	(307)555-1234	work	jbrown@tupelo.com	Black Hills	Customer Service Representative
503	Mike Thompson	123 Main St, Greenville NC 28701	(803)555-1234	work	mthompson@blackhills.com	Greenville	Sales Manager
504	David White	456 Elm St, Atlanta GA 30303	(404)555-1234	work	dwhite@blackhills.com	Atlanta	Regional Account Manager
505	Emily Black	567 Peachtree St, Atlanta GA 30303	(404)555-1234	work	eblack@blackhills.com	Atlanta	Product Sales Manager
506	Alice Green	123 Main St, Greenville NC 28701	(803)555-1234	work	agreen@blackhills.com	Greenville	Technical Support Engineer
507	Steve White	456 Elm St, Atlanta GA 30303	(404)555-1234	work	swhite@blackhills.com	Atlanta	Customer Support Representative
508	Danny Black	567 Peachtree St, Atlanta GA 30303	(404)555-1234	work	dbrown@blackhills.com	Atlanta	Graphic Artist
509	Amy Black	123 Main St, Greenville NC 28701	(803)555-1234	work	abrown@blackhills.com	Greenville	Graphic Artist

Tablet View:

Contact ID	Name	Address	Phone Num	Phone Type	Email	Company	Job Title
500	Red Victors	4517 Elm St, Riverside NJ 08075	(301)356-8921	home	fthompson@yahoo.com	ABC Inc.	President
501	Bob Green	2101 Eucalyptus Ave, Philadelphia PA 09119	(202)901-2121	home	dbreal@hotmail.com	Tupelo Industries	Product Manager
502	John Brown	331 Birch Cir, Black Hills SD 82101	(307)555-1234	work	jbrown@tupelo.com	Black Hills	Customer Service Representative
503	Mike Thompson	123 Main St, Greenville NC 28701	(803)555-1234	work	mthompson@blackhills.com	Greenville	Sales Manager
504	David White	456 Elm St, Atlanta GA 30303	(404)555-1234	work	dwhite@blackhills.com	Atlanta	Regional Account Manager
505	Emily Black	567 Peachtree St, Atlanta GA 30303	(404)555-1234	work	eblack@blackhills.com	Atlanta	Product Sales Manager
506	Alice Green	123 Main St, Greenville NC 28701	(803)555-1234	work	agreen@blackhills.com	Greenville	Technical Support Engineer
507	Steve White	456 Elm St, Atlanta GA 30303	(404)555-1234	work	swhite@blackhills.com	Atlanta	Customer Support Representative
508	Danny Black	567 Peachtree St, Atlanta GA 30303	(404)555-1234	work	dbrown@blackhills.com	Atlanta	Graphic Artist
509	Amy Black	123 Main St, Greenville NC 28701	(803)555-1234	work	abrown@blackhills.com	Greenville	Graphic Artist

Mobile View:

Contact ID	Name	Address	Phone Num	Phone Type	Email	Company	Job Title
500	Red Victors	4517 Elm St, Riverside NJ 08075	(301)356-8921	home	fthompson@yahoo.com	ABC Inc.	President
501	Bob Green	2101 Eucalyptus Ave, Philadelphia PA 09119	(202)901-2121	home	dbreal@hotmail.com	Tupelo Industries	Product Manager
502	John Brown	331 Birch Cir, Black Hills SD 82101	(307)555-1234	work	jbrown@tupelo.com	Black Hills	Customer Service Representative
503	Mike Thompson	123 Main St, Greenville NC 28701	(803)555-1234	work	mthompson@blackhills.com	Greenville	Sales Manager
504	David White	456 Elm St, Atlanta GA 30303	(404)555-1234	work	dwhite@blackhills.com	Atlanta	Regional Account Manager
505	Emily Black	567 Peachtree St, Atlanta GA 30303	(404)555-1234	work	eblack@blackhills.com	Atlanta	Product Sales Manager
506	Alice Green	123 Main St, Greenville NC 28701	(803)555-1234	work	agreen@blackhills.com	Greenville	Technical Support Engineer
507	Steve White	456 Elm St, Atlanta GA 30303	(404)555-1234	work	swhite@blackhills.com	Atlanta	Customer Support Representative
508	Danny Black	567 Peachtree St, Atlanta GA 30303	(404)555-1234	work	dbrown@blackhills.com	Atlanta	Graphic Artist
509	Amy Black	123 Main St, Greenville NC 28701	(803)555-1234	work	abrown@blackhills.com	Greenville	Graphic Artist

Chapter 8

CSS Pre-processing Systems

Using SASS and LESS to make more



Overview

What is a CSS Pre-processing System?

Implementing with SASS

Using LESS



CSS Pre-processing

- Standard CSS has limitations
 - Not very re-usable
 - Changes must be made throughout the CSS individually
- CSS 3 did little to improve this
- Enter the CSS Pre-processing framework...



Pre-processing Frameworks

CSS Pre-processing Frameworks

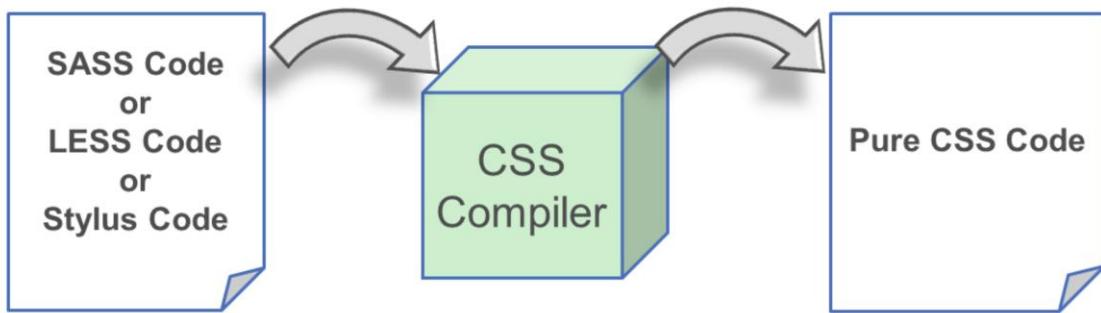
SASS

LESS

Stylus

Advantages
Less Redundancy
Variables and Mixins

Disadvantages
Updating CSS is
one layer removed





Syntactically Awesome Stylesheets

Homepage: <http://sass-lang.com>

- Requires Ruby to be installed
 - After installing Ruby, issue the command:

`gem install sass`

- SASS Stylesheets are written in 2 formats
 - SCSS – "Sassy CSS" (.scss files) currently used format
 - SASS – (.sass files) older format, uses indentation, less encountered now



SASS requires Ruby to be installed first. Afterwards, simply executing the `gem install sass` command will cause SASS to be installed.

SASS stylesheets can be written using the classic (original) format, which uses files proper indentation instead of curly braces to indicate ownership. SCSS file formats support standard CSS3 syntax.

To convert .sass files to .scss files use the conversion tool `sass-convert myfile.sass myfile.scss`



SASS can "Watch" and Convert

- Issue the following command to have SASS automatically update your .css files when changes to .scss files occur:

```
sass --watch myfile.scss:myfile.css
```

SASS runs in the background. It actively monitors the files (directories) you specify. For example, in slide, issuing the described command will cause sass to "watch" for changes to your .scss file. When you save it, sass automatically rebuilds to CSS file.

SASS can also monitor entire directories and convert .scss files within the directory as well.



SASS Nesting (DRY)

```
#empDetails {  
    width: 80%;  
    margin: 0 auto;  
    font: normal 16px Arial;  
}  
  
#empDetails label, #empDetails input[type="text"] {  
    display: block;  
}  
  
#empDetails input[type="text"] {  
    margin-bottom: 15px;  
}
```

Take this original CSS...



SASS (like Ruby) encourages the DRY principle, don't repeat yourself. To this means, it uses the concept of "Nesting" rules.



SASS Nesting (*continued*)

...the SASS way

```
#empDetails {  
    width: 80%;  
    margin: 0 auto;  
    font: normal 16px Arial;  
  
    label, input[type="text"] {  
        display: block;  
    }  
  
    input[type="text"] {  
        margin-bottom: 15px;  
    }  
}
```



(style with attitude)

SASS (like Ruby) encourage the DRY principle, don't repeat yourself. To this means, it uses the concept of "Nesting" rules.

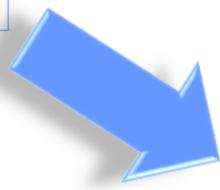


SASS Variables

```
$width: 80%;
```

```
#wrapper {  
width: $width;  
min-width: 800px;  
margin: 10px auto;  
}
```

emp.scss



```
#wrapper{  
width: 80%;  
min-width: 800px;  
margin: 10px auto;  
}
```

emp.css



Sass.

(style with attitude)



SASS Mixins

```

@mixin gradient-mixin {
background-color: #1d2fcf;
background-image: -webkit-gradient(linear, left top, left bottom, from(#1d2fcf), to(#faf5fa));
background-image: -webkit-linear-gradient(top, #1d2fcf, #faf5fa);
background-image: -moz-linear-gradient(top, #1d2fcf, #faf5fa);
background-image: -ms-linear-gradient(top, #1d2fcf, #faf5fa);
background-image: -o-linear-gradient(top, #1d2fcf, #faf5fa);
background-image: linear-gradient(to bottom, #1d2fcf, #faf5fa);
}

#empSearch {
-webkit-box-shadow: 0px 0px 4px 0px #c23cc2;
box-shadow: 0px 0px 4px 0px #c23cc2;

@include gradient-mixin;

a {
position: absolute;
bottom: 10px;
right: 20px;
}
}

```



Mixins allow for groups of properties to be "mixed in" or reused in various parts of CSS code. Declare a mixin using `@mixin` and then use it with an `@include` declaration.

An additional reference for more you can do with SASS can be found here:
<http://sass-lang.com/tutorial.html>



LESS

- LESS is another common CSS-Preprocessing framework
 - LESS requires **nodejs** to be installed first
 - Install LESS using **npm install -g less**
 - Run the LESS compiler using
lessc options main.less main.css



Homepage: <http://lesscss.org/>



LESS Variables

- Like SASS, constants can be created in LESS

```
@company-green: #072;  
@company-green2: @company-green + #040;
```

```
#header {  
  color: @company-green2;  
}
```



```
#header {  
  color: #00bb22;  
}
```

LESS variables, which are really more like constants can then be used throughout the LESS CSS.



LESS Mixins

- Mixins are handled like classes in LESS:

```
@company-green: #072;  
@company-green2: @company-green + #040;  
.laf {  
    border: 1px solid black;  
    background-color: #855e85;  
    background-image: -webkit-linear-gradient(top, #855e85, #615561);  
    background-image: linear-gradient(to bottom, #855e85, #615561);  
}  
  
#header {  
    color: @company-green2;  
    .laf  
}
```



```
#header {  
    color: #00bb22;  
    border: 1px solid black;  
    background-color: #855e85;  
    background-image: -webkit-linear-gradient(top, #855e85, #615561);  
    background-image: linear-gradient(to bottom, #855e85, #615561);  
}
```

LESS mixins follow the class creation format. Since classes are re-usable, this is also how LESS defines them. You may reuse classes in your CSS. Be sure to put them last. If something follows the class, it will be interpreted as being nested inside the class.



Nested LESS Rules

- Like SASS, rules may be nested:

```
#header {  
  h1 { font-size: 2rem; }  
  color: @company-green2;  
}
```



```
#header {  
  color: #00bb22;  
  border: 1px solid black;  
  background-color: #855e85;  
  background-image: -webkit-linear-gradient(top, #855e85,  
#615561);  
  background-image: linear-gradient(to bottom, #855e85,  
#615561);  
}  
#header h1 {  
  font-size: 2rem;  
}
```

Here, as in SASS, rules may be nested and will compile into separate rules.



Nested Media Queries

```
@media screen and (min-width: 450px) {  
  .headercolor {  
    color: #007722;  
  }  
}  
  
@media screen and (min-width: 450px) and (min-width: 768px) {  
  .headercolor {  
    color: #009922;  
  }  
}  
  
@media screen and (min-width: 450px) and (min-width: 960px) {  
  .headercolor {  
    color: #00bb22;  
  }  
}
```

```
.headercolor {  
  @media screen and (min-width: 450px) {  
    color: @company-green;  
    @media (min-width: 768px) {  
      color: @company-green + #020;  
    }  
    @media (min-width: 960px) {  
      color: @company-green + #040;  
    }  
  }  
}
```



Care should be given to some of the rule systems within LESS. Readability should be kept in mind. Is the LESS version more readable in this case?



Summary

- CSS Pre-processing frameworks are becoming more and more popular, particularly in enterprise application development environments
- Neither SASS nor LESS enjoys any particular advantage over the other
- Choice of one of these systems is generally made based on current operating environments (does another framework you are using use SASS or LESS?, do you already use NodeJS?, etc.)



Exercise 8 - It's Your Turn

A Little Bit LESS

Perform Exercise 8 Using LESS

```
#contactDetails {  
    width: 85%;  
    margin: 0 auto;  
    font: normal 16px Arial;  
  
    label, input[type="text"] {  
        display: block;  
        width: 90%;  
    }  
    input[type="text"] {  
        margin-bottom: 15px; padding:  
    }  
    input[type="button"] {  
        font: bold 11px Arial;  
        padding: 2px;  
    }  
}
```

- Convert contacts.css to use LESS
- Install LESS as described in the chapter (npm install -g less)
- Rename your contacts.css file to contacts.less

Compile it:

(lessc contacts.less contacts.css)

from a command prompt and test your app to ensure it still works

Nest the header h1 statement inside of the header element. Compile again, verify the CSS is still valid

See if you can find other nestings, such as those for .contactdetails, .contact, or .contactdata

- Compile again and verify it is valid

Note: the LESS install requires an internet connection.

Course Summary



What Did We Learn?

- Layered Architectures
- Semantic HTML
- Browser Developer Tools
- CSS/CSS3 Selectors
- Specificity
- Absolute/Relative Positioning
- Floating, Clearing
- Laying out Pages
- CSS3 Colors, Gradients,
Rounded Corners, Transforms
- Responsive Web Design
- CSS Media Queries
- Scalable Fonts
- Scalable & Adaptable Images
- Flexible Grids
- Unit Systems (relative, root-em)
- Transforming Tables
- CSS Fluid Grid Systems
- Working with Bootstrap
- CSS Pre-processing Solutions
- SASS and LESS



Evaluations

- Please take the time to fill out an evaluation
- All evaluations are read and considered

Questions



Modern CSS Techniques

Exercise Workbook

Exercise 1 - Browser Developer Tools

Overview

This exercise is designed to give you practice using either Chrome Developer Tools or Firebug.

Objectives

At the conclusion of this exercise, you should:

- Be able to select, view, modify/edit various CSS elements on a page

Step 1. Open Your Developer Tools (as needed)

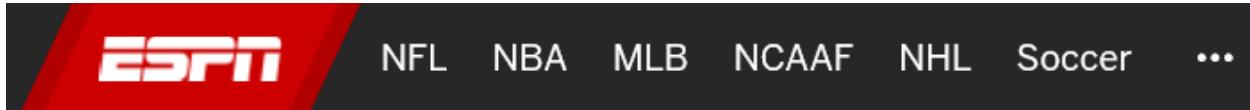
Open either a Chrome Browser or Firefox Browser. Browse to **ESPN.com** (it will redirect you, that's okay).

Press F12 (or CTRL-SHIFT-I will also work in Chrome) to bring up Firebug or Chrome Developer Tools.

Note: If you do not have Firebug installed, you may either: 1) install it from getfirebug.com, or 2) switch to Chrome. Optionally, you may also use Firefox Developer Tools.

Step 2. Locate Elements

On the ESPN.com page, locate the menu bar that looks like the following:



Use the *inspector tool* in Chrome or Firebug to **select the MLB item**. At its innermost level, what kind of element is this?

Mouse over the parent element (the anchor) in the HTML source tree at the bottom in the developer tools. It should highlight in the page up above.

What color(s) do you see when it highlights in the page above?

You should see 2 colors. In Firebug these will be light blue, purple (there is also yellow--not present in this case).

In Chrome these colors are light blue, green(ish) (and orange, not present in this case).

What do these colors mean?

Step 3. Obtain Element Information

With the MLB anchor still selected, add a *margin-right* property to the first selector. Change it to **10px** and note the effect in the page.

What is the value of the padding? Why are there 2 values?

Add a font-size property to the first selector. Give it a value of 150%.

In the HTML source, select the element (parent) that contains the anchor.

Turn off the *float: left* property associated with it. What happens? (floating is discussed later).

Turn the floating back on.

With the still selected, view the Style(s) Tab opposite the HTML source in your developer tools.

In the Style(s) Tab, you should see CSS selectors that contain rules for the . Find the selector that sets the *float: left* and *line-height: 50px*.

A selector is a shorthand notation for identifying one or more nodes. What selector did you see? What does the # (hash) mean? What does the . (period) indicate?

You should have answered an id attribute for the hash (#) and a class attribute for the period (.). Selectors will be discussed shortly.

Verify in the HTML source that the element does indeed match the selector by checking the id and class attributes.

Step 4. The Logo: A Last Task

Using the inspector tool, select the **ESPN logo** at the top of the page. What kind of element did you find?

The element should be an anchor <a> with the text of ESPN. Three questions:

1. Why don't we see this text (instead we see an image)? Where is the text? Hint: look at the CSS properties.
2. Why did they hide this text this way ?
3. Why did they wrap it inside of an <h1> ?

Also, note: the <h1> has a transform property. Turn this off. What happens?

Answers to the previous questions:

1. Check the color (transparent) and font property (0px).
2. It is not intended to be seen the screen. It is there so that devices and machines (including screen readers) can understand the content.
3. Semantics. A text logo is typically the most important text. So, semantically, they want devices to understand what the most important text is on their page.

The logo is represented using an image. Yet there is no tag here. Where is the logo image? Can you find it?

Hints:

- Look for background-images on the anchor

This concludes our Browser Developer Tools exercise. Feel free to use these newfound skills to explore further this page or other pages.

As you progress through this course, you should rely more and more on help from these tools and less on help from your instructor. Therefore, proper continued practice using these tools is essential!

Exercise 4 - Multi-column Layout

Overview

This exercise is designed to give you practice building a multi-column layout.

Objectives

At the conclusion of this exercise, you should:

- Be able to create any CSS layout using basic CSS properties

Step 1. Test the Solution First

This application provides actual data from the server. In order for this application to run properly, you must have the server running before testing it.

To run the server, open a command window, browse to the
<student_files_location>/nodejs folder.

Type the command: **node server.js**

You should see a message that your server is running on port 8005. (Assumes nodejs has been properly installed, a setup requirement of this class).

Now, in your browser, browse to localhost:8005/labs/lab04/solution/index.html.

You should see an approximation of what your solution should look like. It is fully functional in terms of the search and the table is populated with data from the server. The JavaScript used in this exercise is irrelevant to the presentation and can be ignored (but don't remove it!).

Step 2. Complete Your Solution

Your task is to create a 3 column layout such that the leading, trailing, and main divs are all columns.

Your choice of colors is up to you and largely unimportant.

Do not change the HTML or the order of this HTML as this might break the functionality.

You should only have to edit lab04/starter/contacts.css.

Some CSS already exists within this file, it styles the dialog box that appears at the bottom of this page. Don't worry about this dialog box, it will be styled in the NEXT exercise.

Tasks:

- 1. Begin by getting your columns in place, don't forget about clearfix**
- 2. Style the form (contactDetailsForm) in the trailing column.**
- 3. Style the results div that displays search results.**
- 4. Style the table, try to use alternating row colors. Hint: use tr:nth-child(odd)**
- 5. Style the title in the <header>**
- 6. Since our header and footer don't have any real content, you may artificially give them a height (100px is fine).**

Note: to test the search box, use contact ID values from 500 - 509.

Exercise 5 - CSS 3 Features

Overview

This exercise provides a chance to manipulate several CSS 3 features to observe their effects in an application.

Objectives

At the conclusion of this exercise, you should:

- Be able to create use gradients, rounded corners, and other CSS 3 features

Step 1. Style the Dialog Box

Note: if you wish to work from your previous solution, see the help file in the lab05/starter directory. It will help you with steps on how to use your lab04 solution that you created.

In this exercise, you should test the search dialog popup box. With the server still running, browse to <http://localhost:8005/labs/lab05/starter/index.html> and click the new "Contact Search" link. An ugly-looking dialog should pop up in the middle of the screen.

Your task: Using the provided contacts.css, style this dialog. You should incorporate a gradient for a background, rounded corners, and even a box-shadow if you wish. Your solution should vary from the solution found in the student files.