WEEK 2 – BOX MODEL AND PAGE LAYOUT

GENERAL ASSEMBLY

FEWD

Joe Bliss
Will Code for Food

AGENDA

Github, Personal Website Review
Google Chrome Inspector
The Box Model: Padding, Margin, Border
Browser Variances: Simple CSS Reset, Normalize.css
The Display Property: Block vs. Inline
 and
Embedded Fonts
Classes and ID's
Nested Selectors
Semantic Elements: <header>, <aside>, <footer>, etc.

LET'S GIT DOWN TO BUSINESS!

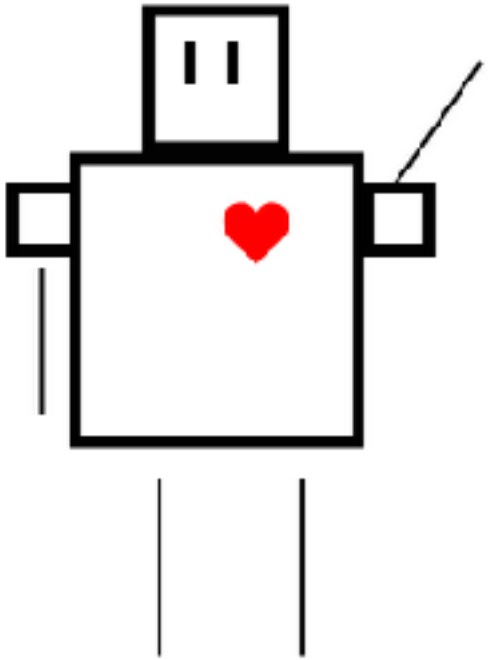
GIT / GITHUB

HOMEWORK WORKFLOW

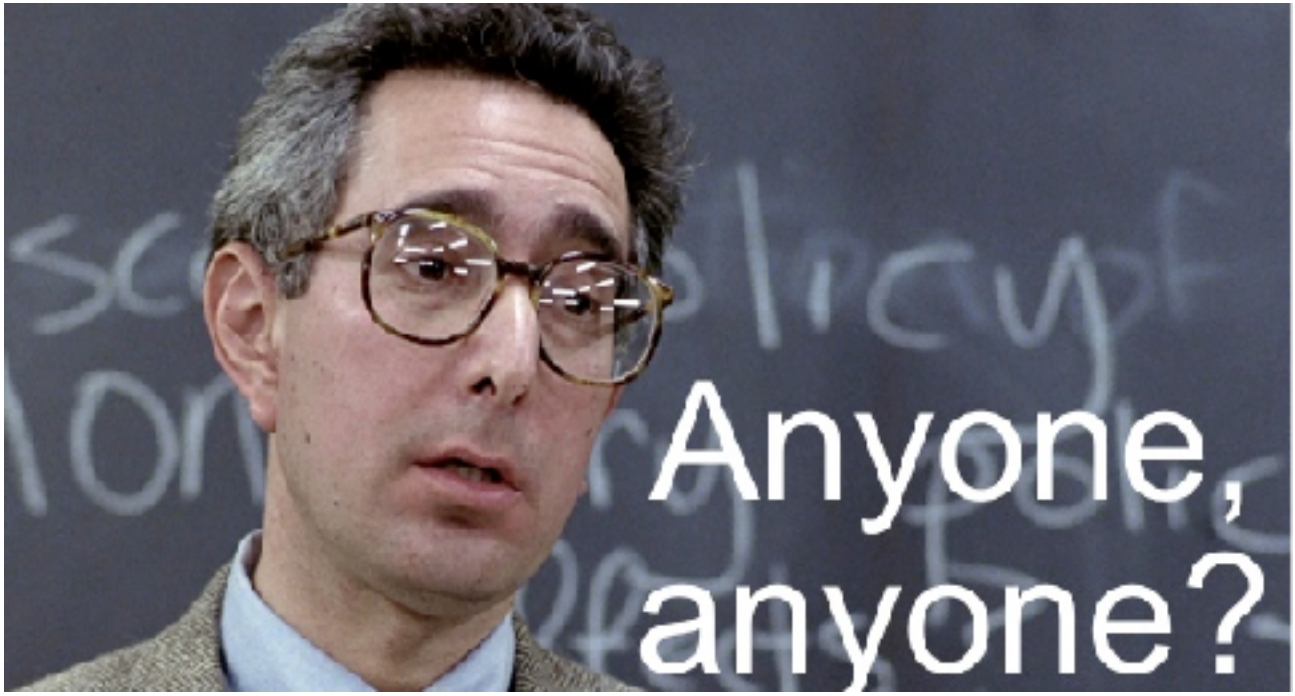
fork, clone, commit, sync, pull request, (and branch and publish … oh my!)

DEPLOYING SITE VIA GITBOT

http://gitbot.co/



PERSONAL WEBSITE



If you're comfortable, please post a link in Slack!

GOOGLE CHROME INSPECTOR

GOOGLE CHROME INSPECTOR

Allows us to “peek under the hood”.

Allows you to see the style of specific elements on a page on-the-fly.

EXERCISE - SCAVENGER HUNT

Pick any (not too complex) site.

What color are the links (default <a> tag)?

Is an <h1> tag being used?

Are other headings being used (h2-h6)?

Find the src of a particular image? Is it a relative or absolute?

How many stylesheets are <link>ed in the <head>?

THE BOX MODEL

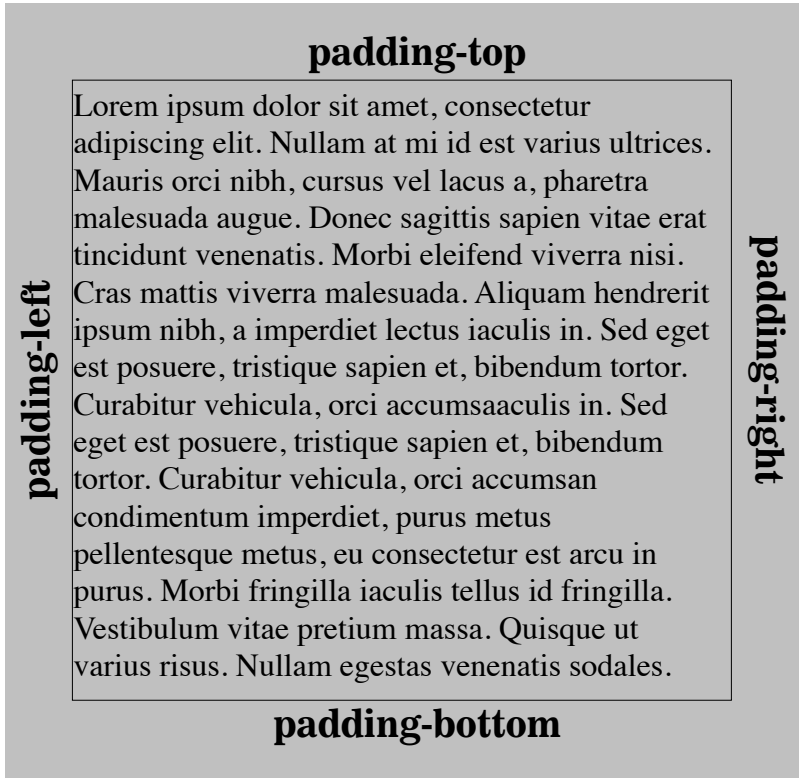
Every element on your website is a box. Most of these boxes get stacked on top of one another.

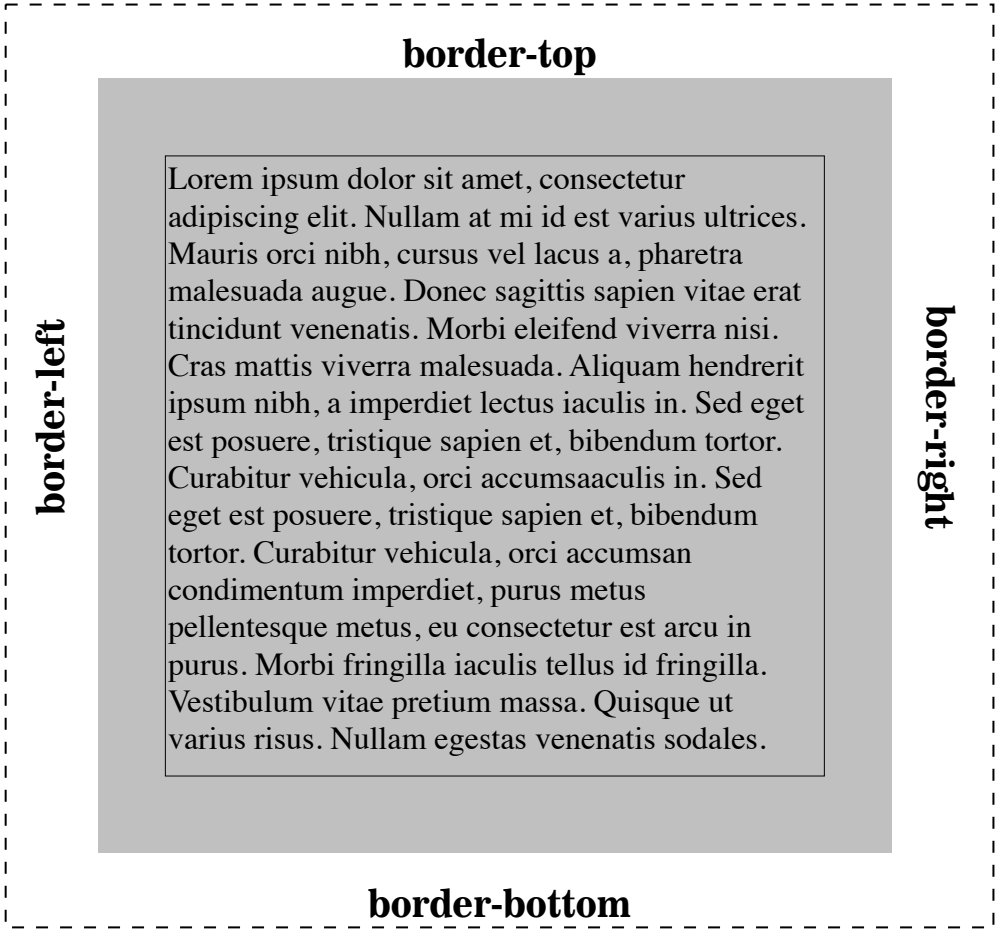
The layout of each “box” is a combination of the object’s content, padding, border, and margin.

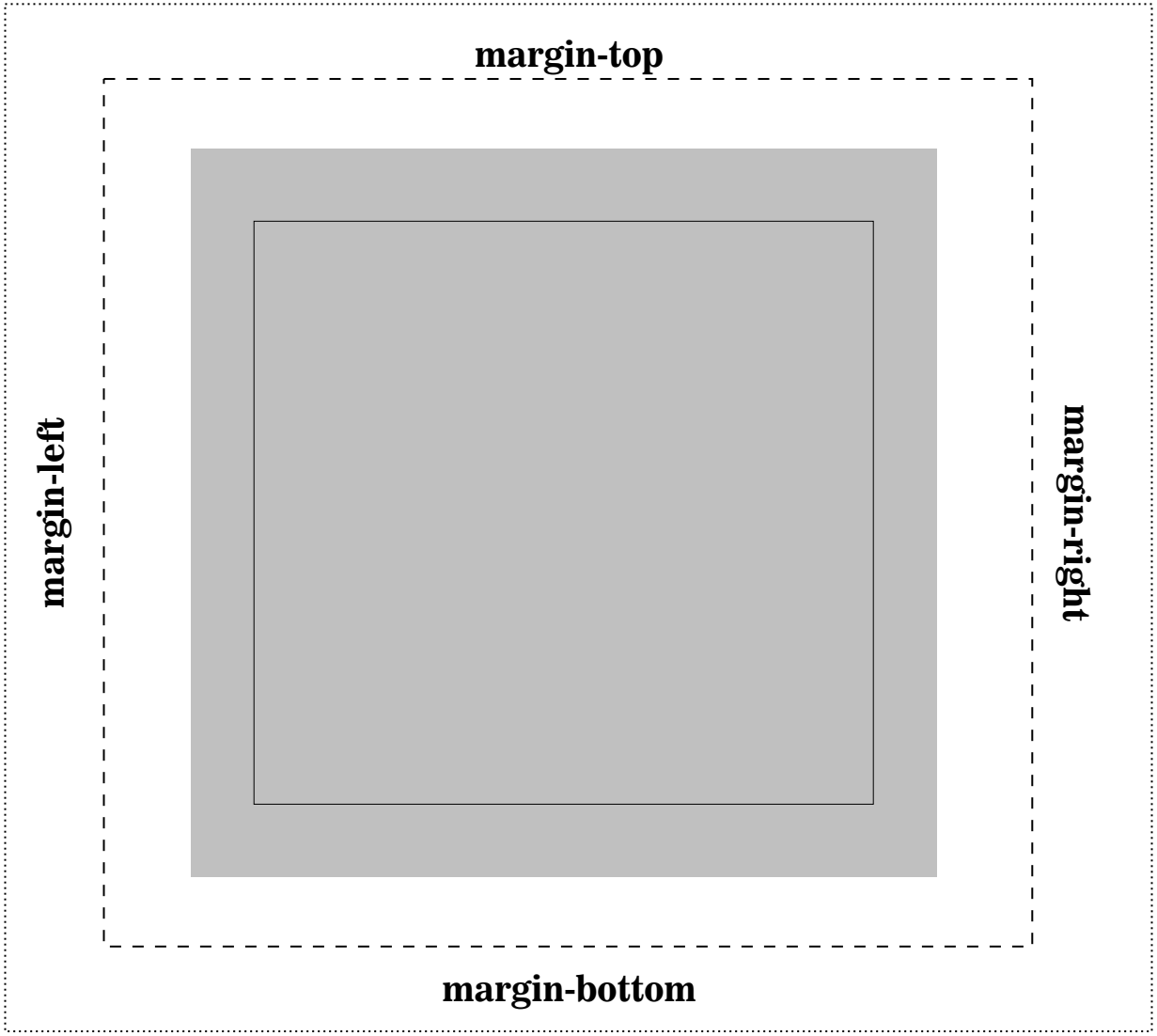
Let’s look at some sites.


Hello World
Content
Padding
Border
Margin

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam at mi id est varius ultrices. Mauris orci nibh, cursus vel lacus a, pharetra malesuada augue. Donec sagittis sapien vitae erat tincidunt venenatis. Morbi eleifend viverra nisi. Cras mattis viverra malesuada. Aliquam hendrerit ipsum nibh, a imperdiet lectus iaculis in. Sed eget est posuere, tristique sapien et, bibendum tortor. Curabitur vehicula, orci accumsaaculis in. Sed eget est posuere, tristique sapien et, bibendum tortor. Curabitur vehicula, orci accumsan condimentum imperdiet, purus metus pellentesque metus, eu consectetur est arcu in purus. Morbi fringilla iaculis tellus id fringilla. Vestibulum vitae pretium massa. Quisque ut varius risus. Nullam egestas venenatis sodales.


padding-top
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam at mi id est varius ultrices. Mauris orci nibh, cursus vel lacus a, pharetra malesuada augue. Donec sagittis sapien vitae erat tincidunt venenatis. Morbi eleifend viverra nisi. Cras mattis viverra malesuada. Aliquam hendrerit ipsum nibh, a imperdiet lectus iaculis in. Sed eget est posuere, tristique sapien et, bibendum tortor. Curabitur vehicula, orci accumsaaculis in. Sed eget est posuere, tristique sapien et, bibendum tortor. Curabitur vehicula, orci accumsan condimentum imperdiet, purus metus pellentesque metus, eu consectetur est arcu in purus. Morbi fringilla iaculis tellus id fringilla. Vestibulum vitae pretium massa. Quisque ut varius risus. Nullam egestas venenatis sodales.
padding-left
padding-right
padding-bottom


border-top
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam at mi id est varius ultrices. Mauris orci nibh, cursus vel lacus a, pharetra malesuada augue. Donec sagittis sapien vitae erat tincidunt venenatis. Morbi eleifend viverra nisi. Cras mattis viverra malesuada. Aliquam hendrerit ipsum nibh, a imperdiet lectus iaculis in. Sed eget est posuere, tristique sapien et, bibendum tortor. Curabitur vehicula, orci accumsaaculis in. Sed eget est posuere, tristique sapien et, bibendum tortor. Curabitur vehicula, orci accumsan condimentum imperdiet, purus metus pellentesque metus, eu consectetur est arcu in purus. Morbi fringilla iaculis tellus id fringilla. Vestibulum vitae pretium massa. Quisque ut varius risus. Nullam egestas venenatis sodales.
border-left
border-right
border-bottom


margin-top
margin-left
margin-right
margin-bottom

BOX MODEL

content

- The text and images that are within the element <tags>

padding

- The space between the content and the border of the element.

border

- The border around the element.

margin

- The space between the border and the other objects on the page.

PADDING

Often specified via pixels, it adds space around an objects content.

Padding is considered part of the overall width of the object.

Backgrounds extend to padding.

BORDERS

Usually specified via pixels, it adds a border to an object outside of the background

Borders is considered part of the overall width of the object.

Backgrounds do extend to borders.

A NOTE ON BORDERS

Borders consist of 3 properties:

- **border-width**: a number of pixels indicating how wide or "heavy" the border should be
- **border-style**: what type of border it is (common value are dotted, dashed, solid, and double)
- **border-color**: what color the border is

BORDER SHORTHAND

```
p {
  border-width: 2px;
  border-style: dashed;
  border-color: red;
}
```

OR

```
p {
  border: 2px dashed red;
}
```

BORDER SHORTHAND

This is the first instance we've seen of CSS Shorthand properties. Sometimes, related CSS properties can be grouped-together, rather than listing-out.

We will use shorthand values of many properties.

MARGIN

Often specified via pixels, it adds space around an object between the border and other objects on the page.

Margin is NOT considered part of the overall width of the object.

Backgrounds do not extend to margins.

I CAN HAZ BOX?


(PADDING)
(CONTENT)
(MARGIN)
(BORDER)
IZ IN YUR BOX MODEL MAKUN NAPS

BROWSER VARIANCES

Every web browser has a different default style, what is referred-to as that browser's "User Agent Stylesheet". It is in place to attempt to make a website rendered without CSS more legible. (underlined and different colored links, variable sizes for h1-h6, etc.).

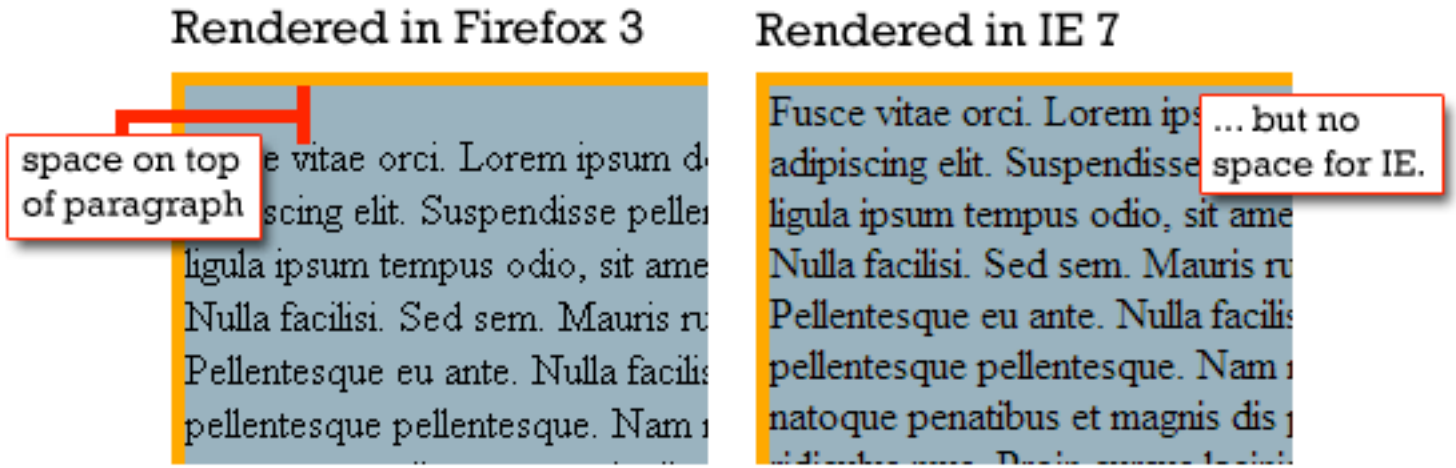
However, every browser's user agent stylesheet is subtly DIFFERENT (or not-so subtly different). This makes coding consistent Front-End code a headache.

FOR EXAMPLE

Browser	font-size	margin
W3C Recommended	2em	0.67em 0
IE7	24pt	14.25pt 0
IE8	2em	0.67em 0
FF2	32px	21.4667px 0
FF3	32px	21.4333px 0
Opera	32px	21px 0
Safari	32px	21px 0

http://www.vanseodesign.com/css/css-resets-pros-cons/

FOR EXAMPLE


Rendered in Firefox 3
Rendered in IE 7
space on top of paragraph
... but no space for IE.
Fusce vitae orci. Lorem ips
adipiscing elit. Suspendisse
ligula ipsum tempus odio, sit ame
Nulla facilisi. Sed sem. Mauris ru
Pellentesque eu ante. Nulla facilis
pellentesque pellentesque. Nam
natoque penatibus et magnis dis

DEALING WITH BROWSER VARIANCES

Way 1: Reset It

- A simple CSS reset that removes all default padding and margin from an object:

```
* {
    margin: 0;
    padding: 0;
}
```

CODEALONG – BROOK+LYN

We will use the simple CSS reset in this exercise.

DISPLAY: BLOCK; VS. DISPLAY: INLINE;

All of our HTML elements fall into one of two categories: Those that are “inline” and those that are “block”. More specifically, all HTML elements have a CSS property - “display” - set by default to be one of two things: display: block; or display: inline; We will refer to these, shorthandedly, as being “block” elements and “inline” elements.

BLOCK ELEMENTS …

… expand to fill their parent container (unless an explicit width is set).

… can have margin and padding assigned.

… will expand naturally to the height of whatever content is within them (unless an explicit height is set).

.. will be placed below previous elements in the markup and break to a new line after.

BLOCK ELEMENTS

BLOCK ELEMENTS

Examples of Block Elements

`<p>`, `<h1>...<h6>`, `<ul>`, `<li>`

INLINE ELEMENTS

… flow along with text content, thus will not clear previous content to drop to the next line like block elements.

… will ignore top and bottom margin settings, but will apply left and right margins, and any padding.

… will ignore the width and height properties.

INLINE ELEMENTS

INLINE ELEMENTS FLOW WITH TEXT

PELLENTESQUE HABITANT MORBI TRISTIQUE SENECTUS ET NETUS ET MALESUADA FAMES AC TURPIS EGESTAS. VESTIBULUM INLINE ELEMENT VITAE, ULTRICIES EGET, TEMPOR SIT AMET, ANTE. DONEC EU LIBERO SIT AMET QUAM EGESTAS SEMPER. AENEAN ULTRICIES MI VITAE EST. MAURIS PLACERAT ELEIFEND LEO.

INLINE ELEMENTS

Examples of Inline Elements:

`<a>`, `<strong>`, `<em>`, and `<code>`.

BLOCK AND INLINE

The easiest way to see these behaviors is by setting the background-color of several elements.

http://codepen.io/ga-joe/pen/BzWNgR?editors=110

HORIZONTAL CENTERING

We can center an inline object by setting text-align: center; on its parent.

We can center a block object by setting a width on it and margin-left: auto; and margin-right: auto;

SPAN AND DIV

 is a generic inline element.

 is a generic block element.

They have no inherent meaning.

We use them to provide structure for styling other elements that they wrap.

SPAN

We use to add hooks onto specific words / phrases within flowing content.

http://codepen.io/ga-joe/pen/gMmpNL

THE ALMIGHTY DIV!

The most basic building block of HTML.

By itself, it does nothing and has no dimensions. It is used to organize other tags into blocks of content.

We use it for layout EVERYWHERE.

CODEALONG - USING THE DIV

Joe's Bistro

Appetizers
Olive Tapenade
Babaganoush
Chackchouka

Entrees
Leg Of Lamb
Wild Mushroom Risotto
Wild Salmon Tomato Confit

Desserts
Pastilla Au Lait
Strawberry & Cream Caramel
Chocolate Mousse

EMBEDDING FONTS

EMBEDDING FONTS

If we want to use a font not considered “Web Safe” or want to pick a custom font for our pages, we can embed a font file with our site. With embedded fonts, the browser is “temporarily installing” the font and rendering all the type with the given font. Yay!

Unfortunately different browsers understand different font file types. Yes. Seriously.

EMBEDDING FONTS

There are four ways to embed fonts:
- Use Google Fonts
- Use another embedding service (e.g. https://typekit.com/)
- Download a webfont kit(e.g. http://www.fontsquirrel.com/)
- Create a webfont kit (advanced)

GOOGLE FONTS

https://www.google.com/fonts

Pick a font you like.
Click “+”.
Copy <link> tag provided.
Paste into <head> tag before your own CSS.

Use the “font-family” shown in your CSS.

Boom!

@FONT-FACE

@FONT-FACE

Download the webfont kit.

Create a folder for “fonts” in your project folder, copy all font files from the webfont kit into your fonts folder.

Copy the @font-face rule from the enclosed stylesheet to yours, making appropriate changes to filepaths.

CODEALONG - EMBEDDED FONTS

Add a Google Font

Add a Webfont Kit

DEALING WITH BROWSER VARIANCES

Way 2: Normalize It

- A way to make all browsers the same while maintaining some default styles.
- https://necolas.github.io/normalize.css/

NORMAL EYES VS. STEVE BUSCEM-EYES

http://chickswithstevebuscemeyes.tumblr.com/

DEALING WITH BROWSER VARIANCES

There are many different ways to do it, and there is spirited debate among Developers which is right, the best, and how to use them.

http://sixrevisions.com/css/should-you-reset-your-css/

http://www.vanseodesign.com/css/css-resets-pros-cons/

http://cssreset.com/which-css-reset-should-i-use/

https://www.sitepoint.com/css-resets-useful-or-useless/

PARTNER EXERCISE – ECARDLY

We will start it together.

If you're feeling up for a challenge (and have worked with "float" or "flexbox" before), try to tackle Ecardly Two Column.

RIDDLE ME THIS ...

What if we want to set the width of one of our 's without affecting the rest?

What if we have 3 <p>'s on our site that we want red and two we want blue?

What would we do?

CLASSES AND IDS

Classes and ID's are a way, in the HTML, to provide "flags" on content that we can then style in the CSS.

IDS

```
#someid {
  //css goes here
}

<!-- This content will be styled accordingly -->
<div id="someid">
</div>
```

CLASSES

```
.someclass {
//css goes here
}

<!-- This content will be styled accordingly -->
<div class="someclass">
</div>
<div class="someclass">
</div>
```

CODEALONG – COFFE SHOP

http://codepen.io/ga-joe/pen/LZWVKP

Add a “milk” class to the drinks that contain milk.

Add an “order” id to the “For: Joe” line.

RULE OF THUMB

Use an ID when you are styling one specific item.

Use a CLASS when you are styling a group of items.

HTML5 STRUCTURAL ELEMENTS

The following behave exactly like 's, but add semantic value to your code:

<header>

<footer>

<nav>

<section>

<aside>

<article>

HTML5 ELEMENTS

<header>

- Defines a header for a document or section

<footer>

- Defines a footer for a document or section

<nav>

- Defines navigation links

HTML5 ELEMENTS

<section>

- Defines a section in a document

<aside>

- Defines content aside from the page content

<article>

- Defines an article (as in a blog or news site)

CODEALONG – HTML5 ELEMENTS

Let's incorporate these into Ecardly.

Which elements that we are already using will be replaced with HTML5 elements with more semantic value than just a ?

EXERCISE – FASHION BLOG

If you're feeling ambitious, tackle the Fashion-Blog Part 2.