Ultimate Web Design

*Start: 2/3/2016 at 8:55 pm*

**Section 1: Intro**  
 Gimp, Pixlr, Photoshop (30 day free trial)  
 Creative Market (free fonts, etc)

**Section 2: Visual Design**

Basic Elements: Lines, shapes, color palette (3 main colors), texture (focus attention), typography, form   
 (add depth)   
Principles: balance, rhythm, proportion (scale), (white)-space, dominance (focal point), hierarchy, unity

C.R.A.P.: contrast, repetition (ex: all headers uppercase), alignment (grid), proximity (groups are related)

Typography (voice, flow based on type face: giving meaning to words)  
Serif: curves at end (ascender/descender, x-height, cap height) Sans-Serif: straight, clean

Science of Color: use color wheel -- analogous (next to each other), complementary (opposite),   
primary, secondary and tertiary colors, triadic   
hue (color on color wheel), saturation (how colorful), brightness (lightness)  
RGB, hexadecimal, HSB (screen colors)  
CMYK, Pantone (print colors)  
General Guidelines: purpose, color basics, dominant color, accent color(s), different shades/hues, simple   
 color, record color palette

Designing with Grids: organize in easy to navigate, create order, save design time b/c limitations  
group related info.  
 single column, multi, modular, hierarchical  
12-column: divided into 3 main sections – wireframe using grid

Wrap Up:   
 Books: Big Brand Theory, Thinking with Type, 100 Ideas that Changed Graphic Design  
 Websites: smashingmagazine.com, ilovetypography.com, thedsgnblog.com

**Section 3: Adobe Photoshop (*View Tutorials on GIMP*)**

Master 20% of photoshop

Creating a New Photoshop Document: 72ppi on web, 300ppi for print, jpeg or psd can open

Use time to learn GIMP (it is free… unless I decide to buy photoshop, educator discount)

*Wrap Up: Didn’t Use Photoshop, need to go back (and or learn to use GIMP)*

**Section 4: Intro to Web Design (and more Photoshop)**

Jargon the Pros Use: Mood Board -- wireframe – a sketch on paper or screen that is blueprint  
 Rough – first draft comp – second draft mockup – final draft of design  
 user/visitor/customer – those that visit our sites

Four Phases of Web Design Project:   
 Discovery Phase (initial contact, structure) about 10 hours ($50 per hr)  
 Creative Phase (user design, wireframing web design mockups) about 40 hours ($50 per hr)  
 Development Phase (covert to html and css additional web dev: wordpress) about 40 hours ($50 per hr)  
 Launch Phase: (review q&a, move to live server, client training) about 10 hours ($50 per hr)

User Exerience (UX) Design:   
 visual design, information architecture, usability, research, content strategy, interface design  
 Strategies: flow charts, site maps, wireframes, screenshots with annotations  
 Tools: omnigraffle pro (mac), visio (pc), balsamiq.com  
 Books: The Design of Everyday Things, Don’t Make Me Think

Anatomy of Website Design:

Header and Navigation: horizontal along top, brand, logo, most common links, global across all pages  
 Call to Action: tells user what to do next, clear/straight forward, 1 more effective than mult (SignUp)  
 Primary Content: main article, about ¾ of page width  
 Secondary/Tertiary Content: lower on page, next article or links  
 Sidebar: left/right, should not attract away from primary content, enhances UX  
 Footer: very bottom, relevant, but not critical, links to less common pages, social media

*Finish: 2/3/2016 at 11:00 pm (2 hours 5 minutes) 12% complete*

*Start: 2/4/2016 at 9:40 pm*

960.gs (grid system) use templates (also has print outs)  
 Design 1 version in a grid (desktop version built in boostrap and let it become responsive)

Project 1: Use Photoshop (landing page)

Project 2: Redesign ‘The Ugly Blog’ (don’t use comic sans)  
 Google Analytics

*Wrap Up: Didn’t Use Photoshop, need to go back*

**Section 5: Advanced Web Design Challenge**

Design your Own  
 Home About Services Blog Contact

Sitemap and Wireframe (on paper, pdf in folder)  
 Design with Photoshop (TAKE CONSIDERABLE TIME TO DO)  
 Feedback: pasteboard.co, cloudApp, dropbox, flatsies (more professional)

*Wrap Up: Didn’t Use Photoshop, need to go back*

**Section 6: HTML**

HTML: HyperText Markup Language (meaning and structure of web content)  
 HTML Syntax: tags <html> <body> content here </body> (opening and closing tags)  
 attributes: extra info <article attribute = “value”> </article>  
 elements: tag and content together  
 HTML Parent/Child Structure (like Tupperware that stacks 🡪 think family tree)  
 parent – child – child (child are siblings, which can become parents, making parent their ancestor)

Basic Structure: File and Folder (keep contents organized)  
 root: top level directory index.html is home  
 images: contains pictures

mywebsite or my\_website [index, images(logo.jpg), downloads(things.html)]  
 The file or folder will eventually become part of url (NO tab, space, “ < >{}/^`|~)  
 You can fake spaces with \_  
 URLs are mostly case sensitive

Code Editor:   
 mac: (text wrangler by barebones [learn from scratch—the hard way], sublime text, coda2 [Hussey favorite, but one time fee], ***brackets*** [might be worth checking, download plugins, free/open source])   
 pc: (notepad++ [learn from scratch—the hard way], sublime text)

*Finish: 2/4/2016 at 11:20 pm (1 hour 40 minutes) 18% complete*

*Start: 2/6/2016 at 10:50 am*

Download Brackets (html\_intro folder)

Wrap Up:

**Section 7: Intermediate HTML**

Basic Structure   
 <!DOCTYPE html>   
 <html><head><title></title></head><body></body></html>  
if you just type text, you can’t format it 🡪 that is why we put it in basic structure

Heading and Paragraphs  
 6 levels of headings <h1></h1> to <h6></h6> best practice to keep them in order  
 paragraphs <p></p>

Emphasis and Strong Emphasis   
 emphasis looks italic but that is just the default in html for <em></em> can be changed with css   
 strong looks bold <strong></strong>

*Finish: 2/6/2016 at 11:30 am (40 minutes) 20% complete*

*Start: 2/6/2016 at 1:00 pm*

Hyperlinks

Absolute URL – full URL that you are telling document to go to  
 Relative URL – navigate to documents within your document

Lists  
 <ul> Unordered List </ul> <ol> Ordered List </ol> <li> List Item </li>  
 <dl> Description List </dl> <dt> Description Title </dt> <dd>Description Description</dd>  
 <!-- Comments in HTML -->

Images  
 <img src= “photo location url” alt= “description here” height= “ “ width= “ “>   
 (speed up loading time if you put in height and width)  
   
 block elements take up an entire line  
 in line elements have no break after  
 in line block is a hybrid (more on these three things later)

Tables  
 in early 90’s people used tables a lot for grids and all kinds of things, but not now, because HTML is not meant to design layout… that’s what CSS is for[also just doesn’t look that good in this way]  
   
 <table> <tr> <th> </th> </tr> </table> table rows and table headings in table  
 <td> </td> table data (must match number of <th></th>)

<hr> works like break tag, but adds a horizontal rule (line)

Forms  
 <form> </form>  
 <fieldset> is a wrapper that gives a box around info  
 <legend> allows us to put a title in top left corner of fieldset  
 <input type=” “> self closing, needs attributes *there are tons of types* already built in!!  
 placeholder = “Text in box will only work in newer browsers, HTML5”

Special Characters  
 &copy; gives you copy right C for example  
 [www.ascii.cl/htmlcodes.htm](http://www.ascii.cl/htmlcodes.htm) will give entire list and table of how to generate (brackets autocompletes them though)

Project: Code a Basic HTML Web Page

See: html\_project.html

Wrap Up:

*Finish: 2/6/2016 at 3:00 pm (2 hours) 25% complete*

*Start: 2/6/2016 at 11:40 pm*

**Section 8: Advanced HTML and HTML5**

IDs and Classes  
 id is unique (can only be used once) called in CSS by #  
 class is global (can be used multiple times) called in CSS by .

Span and Div Tags called meaningless tags b/c there is no meaning, but prob most commonly used  
 can wrap content and used as containers  
 Span inline element (no line break after element) no width, takes width of content, better for wrapping small amounts of text within an element  
 Div block level (takes entire width of element), better for wrapping a whole bunch of content

HTML5  
 easier to use, intro new/better tags that make more sense (ex: header, footer, time, sidebar)  
 HTML5 tags don’t work for older browsers (but most people aren’t using those) but we can make sure that they work on old crappy browsers too

<header>  
 <footer>  
 <nav> for navigation used to navigate to other parts of webpage or (home, about type at top)  
 <article> paired with section, like a blog post  
 <section> can be used in replacement of div for more meaning ???  
 <aside> side note to article, can style with CSS to put on actual side

<time>  
 <small> makes text smaller, adds a little style (can be changed with CSS) but default small w/html   
 W3 Code Schools will give more info on HTML5

HTML5 Project

Code out with comments first to keep things organized  
 comment out after end of div so you can keep track of the id/class of the open div

Wrap Up:

*Finish: 2/6/2016 at 12:40 am (1 hour) 27% complete*

*Start: 2/7/2016 at 1:15 pm*

**Section 9: Expert HTML and HTML5**

Inputs (not everything will work across the web since it is new with HTML5)  
 associate labels to your inputs (good ux!)  
 <label for= “same\_as\_id\_for\_input”> you can now click on label and goes to input on site  
 <input type= “text/search/email/url/number/etc”> change inputs with the type  
 *See example of inputs below in html5\_inputs.html in brackets*  
 search 🡪 puts an x to cancel out  
 email 🡪 required will check in some browsers that it was entered (validation)  
 url 🡪 required will check in some browsers (you can use plugins to do validation if this fails for email or url)  
 number 🡪 counter with min/max/counter  
 date 🡪 built in calendar  
 month 🡪 pick a month and year  
 week 🡪 pick a week  
 time 🡪 built in clock to pick time  
 date time 🡪 kind of pointless now, but will be used later  
 date time local 🡪 set time to local time  
 color 🡪 a color picker  
  
 Internet Explorer with HTML5  
 write a conditional tag that is a js work around to allow IE to use HTML 5 (HTML5 Shim)  
 IE has a comment that only IE can read (because it is so bad)   
 <!--[if IE] <script></script><![endif]--> tells IE to get its act together

HTML5 Data Attribute *(Come back later, after learning some JS and Bootstrap)*  
 use more than id, class, type, placeholder that were pre-built  
 can now create custom attributes, typically preface with data- (angular would preface with “ng-”)  
 working with js or php allows more eloquent way to work with elements, used in bootstrap

Wrap Up:

**Section 10: Introduction to CSS**

CSS Basics  
 Cascading Style Sheets meant for the presentation/how it will look  
 Used to modify any HTML tag  
 Style Rule h1 {font-size: 20px;} selector (h1) curly braces{CSS styles inside}  
 {properties:values}

Inheritance of Styles  
 <h1> Hey, <em> Charlie </em> how are you? </h1> em would inherit from parent h1 unless specified for em (there are some exceptions like border…)

Measurements and the Box Model  
 px pixels  
 % percent  
 em relative to font size of parents element (if parent is 14px 1em=14px, 2em=28px…)  
 rem relative to font size of root element (html element) (if html element is 18px 1rem=18px)

CSS treats all elements as if they were in a box (with margin, padding, and borders)  
 block-level elements have a width property (take up the line)  
 inline-level elements do not (the box is as big as the element)  
  
 margin – border – padding – ***Content*** – padding – border – margin   
   
 Margin: live outside the box (top and bottom margins overlap, so use larger of 2 margins)  
 top has margin-bottom 30px and bottom has margin-top 10px (only 30px not 40px b/n)  
 only true on top and bottom, not left and right  
 Padding: lives inside the box  
 ex: margin: top right bottom left; if you want to write them instead of margin-top..  
 in some cases margin: top/bottom left/right  
 or margin: top left/right bottom can use same in padding&border  
 Border: you can over ride border: give all same border-bottom-color (this over rides border)

Inline CSS  
 type CSS as style= “attribute” but is very rarely used like this in html,this was done in the old days

Internal CSS  
 internal CSS, which is cleaner than inline CSS, however, HTML is best left CSS-free  
 <style type= “text/css”> </style> in the header with the h1 {color:blue;} in the style  
 all in one line, unless there is more than one, then each on their own line in the { }  
 CLOSE < > and { }

External CSS  
 best option, the CSS is in its own file (outside HTML), and linked to the HTML page… keeps HTML cleanest

In <head> <link rel= “(relationship b/n this page and page your linking…) stylesheet” type= “text/css” href= “style.css”>  
 font-stack: list the order you want computer to look for fonts

ID and Class Selectors  
 a lot of developers use dashes with ID and camel case with classes (or both dashes or camel) be consisten  
 CSS will use the more specific tag (ex body#id will over ride the style off just body)

/\* This is a multi-line comment \*/  
 id\_class.html for examples

Descendant Selectors  
 h1 and p and such already have default styles (margins, padding, etc)

ex .secondary-box p { now I’ve selected the paragraph inside the div class sec-box }  
   
 you can get as specific as you want by adding more elements before the { }

Grouping Selectors  
 grouping.html seen below:  
 .primary-box, .secondary-box { }  
 look at #marginContainer (will show how to make boxes centered in this lecture)

*Finish: 2/6/2016 at 3:50 pm (2 hour 35 minutes) 34% complete*

*Start: 2/6/2016 at 8:00 pm*

Specificity the bigger the file, the more likely you’ll have issues  
 More Specific = High Priority cascade means down the document  
 p em {font-weight: bold;} em {font-weight: normal;} bold b/c more specific over em coming last  
  
 Specificity Calculator: inline styles (over take all styles… 1000), ids (100),   
 classes/attributes (10), elements (1)  
 <http://specificity.keegan.st/> Specificity Calculator

CSS Project  
 \* { } selects everything  
 css-project\_final.html  
 blog\_final.css

Wrap Up

**Section 11: Intermediate CSS**

Colors [www.w3schools.com/html/html\_colorvalues.asp](http://www.w3schools.com/html/html_colorvalues.asp) [www.paletton.com](http://www.paletton.com)

Text Formatting  
 note on bold (can be bolder or lighter) “normal” 100 thru 400 “bold” 500 thru 700 “heavier” 800 thru 900  
 letter-spacing: /\* Can be "normal" or a numerical value \*/

text-indent: /\* Indents the first line of the paragraph. Can be numerical value, or percentage based. \*/

line-height: /\* How tall the lines of text are within an element, can be a simple numeric value like "3" which will make the line height 3x the height of the font-size. You may also use a length, like "px" or "em". Percents are also ok \*/

text-decoration: /\* text-decoration -underline, overline, line- through or none... underlines are best left for links \*/

text-transform: /\* -uppercase, lowercase, capitalize, none \*/

Borders  
 see border.html and stylesheets.css

Background Images  
 background-images.html and background-images.css has notes on how to use them!  
 /\* By default, background images repeat X & Y and are positioned top left \*/

Styling Links  
 4 states you can style links in (\*\*\*must be in order: a:link, a:visited, a:hover, a:active\*\*\*)  
 /\* unvisited \*/ /\* visited \*/  
 /\* hovered or mouse-over \*/ /\* active \*/   
 Wrap Up:

*Finish: 2/6/2016 at 10:45 pm (2 hour 45 minutes [with SB50]) 38% complete  
 //* {12hours 45minutes over 5 days total}

{spent night helping Sam with his Javascript homework, code on github 🡪 then finish javascripting on node school}

*Start: 2/9/2016 at 10:30pm*

**Section 12: Advanced CSS**

Styling Web Forms (styling forms.html)

Block, Inline, and Inline-Block  
 Majority are block (take up browser from edge to edge with line break)  
 or inline (do not take up whole line, only as wide as element, no line break)

Inline-block (box is only as wide as content (like inline), but content sits on own line (like block)  
 Call for block, inline or inline-block by using display:

Understanding Float and Clear  
 float works like a magazine (text floats around image, like Microsoft ‘square’)  
 must have a set width in order to float (if it is full it can’t float)  
 gutter (space between main content and sidebar)

*Relative, Absolute and Fixed Position*  
 position: **relative**; move relative to top left corner top: px left: (is moved)  
 position: **absolute** takes element out of flow of page more flexibility on where you want to move that element… **moved based on parent container** quick find top: 0 or left: 0 to find   
 position: **fixed** stays on the page exactly where it is hard coded even if scrolled

*Finish: 2/10/2016 at 12:10 am (1 hour 40 minutes) 40% complete*

*Start: 2/10/2016 at 10:00 pm (1 hour) work on Porfolio until 11:00 pm*

*Start: 2/10/2016 at 11:00 pm*

**CSS Project: Build the Google Home Page**

Background image needs to be set 2x the size (so have to cut px in half when taking them from internet)  
 shift command 4 measures things in pixels

fake-google.html in css folder

*Finish: 2/11/2016 at 12:15 pm (1 hours 15 minutes) 41% complete*

*Start: 2/11/2016 at 10:00 pm*

**Section 13: Expert CSS & CSS3**

Box Sizing and Round Corners  
 normalize.css

box-sizing: border-box; doesn’t add padding to height and width put puts padding inside like you’d think  
 border-radius: px; create rounded corners (can do border-left-radius: 100px to just do one corner… or multiple)

-webkit-border-radius browser prefix allows css3 to work in older browsers  
 -moz-border-radius

CSS3 Colors & Gradients  
 rgba red, green, blue, alpha alpha 🡪 how transparent, opacity (1 is 100% fully visible, 0 is fully transparent) can see image thru   
 hsla hue, saturation, lightness, alpha

Gradients 🡪 background: linear-gradient(blue, yellow); default, top… side is   
 background: linear-gradient(to left blue, yellow); to top right or 45deg, blue, yellow  
 background: radial-gradient(blue, yellow); can add as many colors as you want  
 <http://w3schools.com/css/css3_gradients.asp>

CSS3 Shadows  
 can be used to add contrast  
 text-shadow: vertical shadow horizontal (blur is optional 3rd) shadow color   
 can use rgba for color   
 text-shadow: 2px 2px black; can put a second shadow , after color and do again

Box-shadow: vertical horizontal (blur) (spread) color; spread 🡪 how far it spreads out

CSS3 Columns can put divs and float each column but CSS3 can do it also   
 (aren’t compatible with browsers, need to use prefixes, –webkit- and –moz-)  
 column-count: #of columns; -webkit-column-count: -moz-column-count  
 column-gap: pixels you want in between each gap (use prefix again)  
 column-rule-style: solid, dotted, dashed (puts a line between columns)  
 column-rule-width: how thin/thick do you want this line between columns to be  
 column-rule-color: change color of the line between columns

CSS3 Animations & Transitions

@keyframes box { /\* what you want animation to do \*/  
 from { }  
 to { } (can also use percents 0% { } 50% { } 100% { }

#animation #box {  
 animation-name: (id/class)  
 animation-duration: how long do you want it to run (ex: 10s;)  
 animation-timing-function: linear (ease-in…. lots of choices)  
 animation-delay: wait to start animation (ex: 3s;)  
 animation-iteration-count: how many times do you want it to run (ex: 7;)  
 animation-direction: reverse; would go from last to first keyframe (alternate back&forth)  
 }  
 /\* shorthand in order as above \*/ animation: box 10s linear 3s 7 reverse;

Transitions used for hovering over objects and increase user experience  
 transition: property(ex: width/height/all) duration(ex: 5s)  
 transition: width 8s, height 10s, background 20s; can do multiple properties at once!

Project #1 Create Animated Buttons  
 buttons.html with animation

Project #2 An Orbiting Planet Animation  
 must set postion to use z-index  
   
 Reminder on Positioning:   
 position: **relative**; move relative to top left corner top: px left: (is moved)  
 position: **absolute** takes element out of flow of page more flexibility on where you want to move that element… **moved based on parent container** quick find top: 0 or left: 0 to find   
 position: **fixed** stays on the page exactly where it is hard coded even if scrolled

<https://projects-codecuddy.c9users.io/earth-orbit/index.html>

*Finish: 2/11/2016 at 12:30 am (2 hours 30 minutes) 45% complete*

*Start: 2/13/2016 at 2:00 pm*

**Section 14: Advanced HTML and CSS Challenge**

Gimp paths tool (800%) add an alpha channel in layers   
 cross grove on last connection right click inside path “select 🡪 from path” ctr+I 🡪 delete

*Finish: 2/13/2016 at 8:30 pm off and on (4 hours) 46% complete*

*Start: 2/14/2016 at 12:00 pm*

Website is not yet responsive, but it does fit on a normal sized computer screen…

*Finish: 2/14/2016 at 1:15 pm (1 hour 15 minutes) 46% complete*

*Start: 2/14/2016 at 1:30 pm*

**Section 15: Introduction to Javascript (80-20 Principle… focus on 20% that gets 80% results)**

Intro:  
 Object Oriented Language, used to enhance behavior of websites or applications  
 JS

Internal and External Javascript:  
 3 ways just like css 🡪 in style (never), internal or external  
 Internal: Add Javascript within the HTML, best practice to add script at the bottom, just before load last </body>

External: Link an external javascript at the begin in the <head> just like css (This is best practice)  
 still load at bottom of page

Javascript Syntax  
 has a lot in common with other major programming languages (makes learning others easier)  
 Javascript is not short for Java 🡪 entirely different  
 Computer Programs are created by languages like javascript, and give instructions through statements which are separated by semicolons… like… var funMessage = “Hello!”;  
  
 Statements have 5 separate elements:  
 Values: 2 types: **fixed** (**literals**) doesn’t change ever (like numbers, strings, expressions)  
 **variables** value can change, used to store data… uses var = to assign value to variable… ex var myVariable; myVariable = 2\*4;  
 Keywords:  
 Expressions:  
 Operators: 2 most common are:  
 Arithmetic Operators (+ - \* / )  
 Assignment Operators ex: a += b 🡪 a = a + b  
 = assigns value  
 += adds value to variable \*= multiplies  
 -= subtracts value from variable /= divides  
 Comments: //single line comments  
 /\* multi line comments \*/  
 Javascript is CASE SENSITIVE  
*Finish: 2/14/2016 at 2:30 pm (1 hour) 47% complete*

*Start: 2/14/2016 at 4:00 pm*

Javascript Outputs  
 window.alert(); alter window on refresh  
 document.write(); best for quickly testing if something works, but nothing else  
 document.getElementById().innerHTML =   
 console.log(); logs to the console, not really for the user, but for the coder

Javascript Variables  
 containers for storing data  
 declare variable with var //name

“string”

Variables must start with a letter, can’t use some words 🡪 [www.w3schools.com/js/js\_reserved.asp](http://www.w3schools.com/js/js_reserved.asp)

Concatenation  
 used when you don’t know what you want to put together  
 ex: userFirstName userLastName

Javascript Arrays  
 can store multiple values, unlike a variable that can only store one value  
 index 🡪 number associated with array 🡪 always starts with 0  
 arrays are very powerful… you can select each individual element

For ( var i = 0; i < favFoods.length; i++) //start at index = 0; check if i < the array, and as long //as it is; keep adding an i

{  
 document.write( favFoods[i]); //displays all the index  
 document.write(<br>); //puts each on a new line  
 }

Variable is one value  
 Array is multiple values  
 MultiDimensional Arrays have multiple values within the values

Javascript Functions  
 functions are similar across the board in programming languages  
 it is a block of code built to perform a specific task, runs when invoked

Function customName () { //paramaters in () separated by ,  
 //run some code  
 }

return tells function to stop operating that function

If / Else Statements

if(condition to be met) {  
 //execute code if true  
 }

if… else if… else example

var age = 72;

if(age < 30) {

document.getElementById("box1").innerHTML = "You're younger than 30!";

document.getElementById("box1").style.color = "red";

} else if (age > 70) {

document.getElementById("box1").innerHTML = "You're getting old.";

document.getElementById("box1").style.color = "blue";

} else {

document.getElementById("box1").innerHTML = "You must be between 30 and 70 years old.";

document.getElementById("box1").style.color = "green";

}

**Javascript Tip Calculator Project** <sup></sup> //superscript can also use <sub></sub> **= =** is equal to **= = =** is exactly equal to

Wrap Up: http://jsbooks.revolunet.com/

*Finish: 2/14/2016 at 7:00 pm (3 hour) 51% complete*

*Start: 2/15/2016 at 9:30 am*

**Section 16: Introduction to jQuery** Download and Install  
 does everything javascript can  
 Production version compacted, faster to load, but unreadable  
 Development version easier to read, but never really need to do this

put scripts at the bottom to help with load time before </body>

don’t need type with html5 because it defaults to javascript

Simple Syntax of jQuery  
 invented to give developers and easier/powerful way to use javascript

$document.ready(function() { }); can use $(function() { });  
 called document ready, waits until website is ready

$(selector).action () $calls jQuery selector(s) action can have parameters

\*\*\*Help build Keegan an app for her diet and post them to GitHub and CodePen\*\*\*

*Finish: 2/15/2016 at 11:00 am (1 hour 30 minutes) 51% complete*

*Start: 2/15/2016 at 8:45 pm* Selectors  
 jQuery can select any element, any and every, not just html tags/id/etc…

Event Methods  
 user interacts with browser by clicking/hovering (events) jQuery recognizes this and then does something with it

Google or use API to find what jQuery can do… because it probably exists what you want

*Finish: 2/15/2016 at 10:30 pm (1 hour 45 minutes) 53% complete*

*Start: 2/16/2016 at 7:30 pm*

Did codecademy.com ‘new build a website’ 1 hour  
 Download a few things from creative market 30 min  
 read some articles online about bootstrap 30 minutes

*Finish: 2/16/2016 at 9:30 pm (2 hours) 53% complete*

*Start: 2/17/2016 at 9:15 pm*

**Section 17: Intermediate jQuery**

Chaining is efficient and faster for the computer

Hiding, Showing and Fading Content with jQuery   
 fadeIn fadeout fadeTo slideUp slideDown //takes something hidden and makes it appear

slideToggle //if hidden slideDown if not then slideUp

jQuery Animate you can create anything with code

Modifying CSS

You do not want to add any VITAL information in this way because some people might have javascript (jQuery) disabled and won’t see it so don’t use this for your logo, buy-now button, blog post, etc!! Do not use it to style entire site! Should only be used to enhance site!!

*Finish: 2/17/2016 at 10:45 (1 hour 30 minutes) 55% complete*

*Start: 2/19/2016 at 9:00pm*

jQuery Care Race

tiny.cc/raceme

*Finish: 2/19/2016 at 11:00 (2 hours) 56% complete*

*Start: 2/22/2016 at 9:45 pm*

**Section 18: jQuery UI (Advanced jQuery)**

[www.jqueryui.com](http://www.jqueryui.com)

draggable [www.jqueryui.com/draggable](http://www.jqueryui.com/draggable)  
 droppable [www.jqueryui.com/droppable](http://www.jqueryui.com/droppable)  
 sortable [www.jqueryui.com/sortable](http://www.jqueryui.com/sortable)  
 accordion [www.jqueryui.com/accordion](http://www.jqueryui.com/accordion) bootstrap accordion is better  
 heightStyle: "content" resize each accordion to size of content instead of biggest  
 date-picker [www.jqueryui.com/datepicker](http://www.jqueryui.com/datepicker)  
*Finish: 2/22/2016 at 11:00pm (1 hour 15 min) 58% complete*

*Start: 2/23/2016 at 11:00 pm*  
 To-Do-List  
*Finish: 2/23/2016 at 11:30pm (30 min) 59% complete*

*Start: 2/25/2016 at 10:00 pm*

**Section 19: Responsive Design and Development using Bootstrap**

Responsive uses media queries to create mobile optimize websites 🡪 bootstrap saves us time

Boostrap updates pretty regularly so keep it up to date

Getting Started  
 Bootstrap prob the one you want  
 Source Code source with less (nicer use of css… will talk about later) require less compiler|  
 Sass easy inclusion in rais and sass-only projects

Bootstrap CDN can copy the link without having to download it

Basic template  
 you can copy to clipboard and get rid of what you don’t need

Recommend not putting bootstrap into into your own css folders (so it is easier to update to newer bootstrap… just copy and paste the new basic template)

Quickly Prototype   
 Bootstrap gets you up and running fast… lets you work from something and ignore little details  
 In his experience… speeds up time by 50%  
 Using Bootstrap examples

Pick and Example view source and view page source, copy and paste  
 make some minor adjustments in the header for things you don’t need  
  
 favicon 16x16 graphic save as an ico file

Can attach in the theme.min.css files and play around with the themes

Understanding Bootstrap Grid System  
 CSS 🡪 Grid System

<http://getbootstrap.com/css/#grid>

Rows must be placed within a .container (fixed-width) or .container-fluid (full-width) for proper alignment and padding.

Use rows to create horizontal groups of columns.

Content should be placed within columns, and only columns may be immediate children of rows.

Predefined grid classes like .row and .col-xs-4 are available for quickly making grid layouts. Less mixins can also be used for more semantic layouts.

Columns create gutters (gaps between column content) via padding. That padding is offset in rows for the first and last column via negative margin on .rows.

The negative margin is why the examples below are outdented. It's so that content within grid columns is lined up with non-grid content.

Grid columns are created by specifying the number of twelve available columns you wish to span. For example, three equal columns would use three .col-xs-4.

If more than 12 columns are placed within a single row, each group of extra columns will, as one unit, wrap onto a new line.

Grid classes apply to devices with screen widths greater than or equal to the breakpoint sizes, and override grid classes targeted at smaller devices. Therefore, e.g. applying any .col-md-\* class to an element will not only affect its styling on medium devices but also on large devices if a .col-lg-\* class is not present.

Look to the examples for applying these principles to your code.

Grip Options <http://getbootstrap.com/css/#grid>  
 can combine class=“col-md-6 col-sm-8”  
 container – row – then columns

Class=”container-fluid” will center on full screen

Responsive Forms  
 create forms that don’t get in the way of the user  
 can be done with css, but often come across errors 🡪 bootstraps can help here

div   
 Class = container bootstrap class allows center and use grid based layour  
 class = page header bootstrap class with some padding and thin border  
 class = lead increase font size of paragraph  
 class = row bootstrap row to allow us to access columns  
 class = col- - bootstrap grid

class = form-group bootstrap that has label and input grouped together  
 label for = //id of input will allow you to click on label for input  
 input always needs class = “form-control” gives them the style

wrap a checkbox in the label so you can click the words and not just the little box

wrap a radio in its own label but give them but the same name= so you only can select 1  
 if you don’t want them side by side add a break tag

button class= “btn btn-inverse (or btn- default primary warning success info)  
 way easier to do buttons in bootstrap than in css (saves a ton of time!)

form class= “form-inline” lets bootstrap know its inline

class of “sr-only” in label will only be seen in a screen reader (so removes from site)  
 to counter this put a placeholder in the input so user knows what is expected

can wrap an input in a div with class= “input-group” to make it more user friendly…  
 then another div class= “input-group-addon”

<div class="input-group"> //can also be done at the end

<div class="input-group-addon">@</div> so there is a solid @ in front of input

Form class=”form-horizontal” allows us to use grids in forms form-group acts as row

Add disabled attribute to any input to not allow user to type in it

Additional classes to add to form-group has-success has-warning has-error

Additional input css input-sm input-lg change size of input

<http://getbootstrap.com/css/#forms>

*Finish: 2/26/2016 at 12:00am (2 hours) 61% complete*

*Start: 2/27/2016 at 10:45 pm*

Responsive Tables, Buttons, Images  
 tables shouldn’t be used for layouts 🡪 meant for tabular data  
 <table class = “table”> </table> can use a lot more classes

<http://getbootstrap.com/css/#tables>

buttons-images.html

<http://getbootstrap.com/css/#buttons> button over input when using a button

images by default non-responsive class= “img-responsive”  
 class for shapes

<http://getbootstrap.com/css/#images>

Bootstrap Helper Classes  
 text - or bg- (text or background)  
 muted mutes text, less important  
 primary main color, stands out  
 success they did it correctly  
 info info usually blue  
 warning something is wrong  
 danger last step before something bad, ‘are you sure’  
 clear fix class will clear floats so content after doesn’t break it  
 pull-left pull-right  
 class center-block  
 class of show or hidden to show or hide

<http://getbootstrap.com/css/#helper-classes>

Responsive Utilities

Meant for faster responsive can be used for printing

<http://getbootstrap.com/css/#responsive-utilities>

Glyphicons (Font Icons)

These are actually text/fonts (not pictures) so they scale infinitely like text  
 <span class= “glyphicon //copy straight from website below”></span>  
 ex: <span class="glyphicon glyphicon-trash"></span>

<http://getbootstrap.com/components/#glyphicons>

Bootstrap Navbar

If you aren’t going to use <nav> then use <div role=”navebar”>  
 navbar-header built in for responsive hamburger

<http://getbootstrap.com/components/#navbar>

*Finish: 2/28/2016 at 12:45am (2 hours) 63% complete*

*Start: 2/28/2016 at 12:45 pm*

Bootstrap Modal Window  
 <http://getbootstrap.com/javascript/#modals>

Bootstrap ScrollSpy

Data-spy = “scroll” is what makes it scroll

<http://getbootstrap.com/javascript/#scrollspy>

Togglable Tabs  
 <http://getbootstrap.com/javascript/#tabs>

Sliding Carousel  
 first div class in the wrapper must be active or it won’t work  
 <http://placekitten.com>  
 <http://placehold.it>  
 prob want to hide text <hidden-xs> in the div class or just the paragraph etc

<http://getbootstrap.com/javascript/#carousel>

*Finish: 2/28/2016 at 2:45am (1 hour 30 min (break when josie woke up)) 64% complete*

*Start: 2/28/2016 at 10:15 pm*

Startup Website HTML and Bootstrap

*Finish: 2/28/2016 at 11:45am (1 hour 30 min during Oscars with Jenny) 65% complete*

*Start: 2/29/2016 at 8:45 pm*

Startup Website CSS, HTML, and Bootstrap  
 Media Queries find out how wide browser is and then serves up css to adapt to that browser size.

**Section 20: Advanced Bootstrap Challenge**

<https://github.com/codecuddy/mbp_boot>

*Finish: 2/29/2016 at 12:30am (3 hour 45 min) 65% complete*

*Start: 3/1 /2016 at 11:00 pm*

**Section 21: Web Hosting & Domains**

What is Web Hosting? Brad has a free course that goes more in depth on web hosting

Purchasing a Bundled Domain Name & Hosting Package  
 [www.justhost.com](http://www.justhost.com) bradhussey.ca/justhost  
 book site package info for as long as you can (will go up when it expires)  
 domain privacy, site lock are good ideas  
 save your account information that is sent to you (will need FTP later)

The Hosting Control Panel (a.k.a cPanel)  
 <https://my.justhost.com>

Understading FTP & How to Upload a Website to Your Live Server

FTP – file transfer protocol  
 coda2 has FTP transfer built in ($100 to get coda)… there are free options out there

FTP Clients

Flow (costs $5)… you can also edit files, only for mac  
 Cyberduck (it is free, you can donate) available for mac and windows  
 FileZilia (it is free) doesn’t look as pretty, but does the trick

*Finish: 3/2/2016 at 12:00am (1 hour) 68% complete*

*Start: 3/4/2016 at 12:30 pm*

**Section 22: Introduction to PHP**

What is PHP & What Does it Do?   
 server side processor that works nicely with html which makes it popular  
 Basic PHP SYNTAX  
 Speaks directly to the server then spits back html to browser

MAMP (manage your websites locally)  
 localhost:8888

<?php

echo (“Hello World!”);  
 // single line comment  
 /\* multi  
 line here \*/  
 $name = “Charlie”;  
 echo “Hello ” . $name . “<br>”;  
 ?>

Summary  
 php is script can put anywhere, executed on server, html is result sent to browser  
 put php with script tags in basic html  
 virtual server must be up and running in order to see it (MAMP)

*Finish: 3/2/2016 at 2:00pm (1 hour 30 minutes) 68% complete*

**{46 hours 45 minutes over 30 days total \* 68% complete}**

**//Taking break to take Rails Course with Sam**

*Start: 3/4/2016 at 12:30 pm*

PHP Variables & Constants

GIMP

Path Tool  
Right Click on Image 🡪 Add Alpha Channel  
Blow up to 800%  
Left click to drop anchor (ctrl + z)  
after finish…

Right click inside path select from path ctrl+I (to invert) delete  
Save as .png

Export

Business Cards

#F04D37  
#CECFD1

#FEFF00  
#734D26