Metadata – Mastering CSS

**Section 1: CSS Foundations**

Section Description (from the outline): This will introduce the course and re-familiarize the viewer with CSS Foundations that are necessary to understand before they can master CSS.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Problem / Solution | Step 1 | Step 2 | Step 3 |
| 1.1 | Course Overview | The course needs a summary describing what will be learned over the course of each section / Provide a course overview, describe what will be learned | Understand what CSS is and what are we going to learn | Know the expected pre-requisite knowledge like how to write basic html, and even some basic CSS which are required to follow along with this course. | Go over different software such as Sublime Text, Mac, Live Reload I’m using and basics for moving through the course. |
| 1.2 | Anatomy of a Rule Set and the 3 Types of Style Sheets | In order to style websites with CSS, you must understand the syntax for writing CSS rules. | Review a sample rule set and see how it affects a web page | View the rule set in an embedded style sheet and external style sheet | View an inline rule set |
| 1.3 | The box model and block vs. inline elements | All CSS elements conform to a box model. If the box model isn’t understood completely, CSS cannot be mastered. Same goes with block and inline elements | Review of the box model, and using box-sizing border box to change the box model | Block level elements | inline elements |

**Section 2: Ramping Up**

Section Description (from the outline): Before we deep dive into CSS we need to think about where we’re going to write our CSS, understand the design/build process, and set up a reset stylesheet. Once those items are in place, we can start styling text, using classes to rename elements, and learning about compound selectors.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Problem/Solution | Step 1 | Step 2 | Step 3 |
| 2.1 | Text Editors | Writing code is error prone and hard, a good text editor like Sublime Text 3 solves this problem. | Snippets | Syntax Highlighting | Code completion and other cool features |
| 2.2 | CSS Reset | Browsers add a lot of default styling, especially margins and padding. A nice CSS reset can solve this and allows you to provide the styling. | Adding in the reset | What’s the reset doing? | Customizing the reset to fit our needs |
| 2.3 | Chrome Dev Tools | Without fail, CSS doesn’t work. Usually its a tiny overlooked mistake that causes something not to work and is hard to find and fix. Enter the Chrome Dev Tools. | Opening up the Element Inspector | Changing CSS inside the dev tools | Are there any errors in the console? |
| 2.4 | Renaming Elements: Classes and ID’s | CSS allows you to position, style, and control elements on a page. What if you want to style one div differently than another. You can name an element with a class. | Renaming with Classes | Renaming with ID’s |  |
| 2.5 | Descendant Selectors | Renaming elements with classes is extremely powerful feature in CSS. However that’s not the only way to target a specific type of element. Descendant selectors allow you to target elements on a page based on their parent element. | What is a parent and child element? | Creating a descendant selector | Calculating weight of descendant selectors |

**Section 4: Creating Buttons w/Modular, Re-Usable CSS Classes and CSS3**

Section Description: Creating reusable CSS eliminates style sheet “bloat” and allows you to create sites faster without duplicating styles that have already been written. To help with keeping an organized style sheet I’ll introduce modular classes also sometimes referred to as ‘Object-Oriented CSS’. This is also a good time to introduce the CSS3 properties like transforms, transitions, and gradients. And again it also happens to be a good time to introduce CSS specificity, which is the precedence of how certain CSS selectors override others.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Problem/Solution | Step 1 | Step 2 | Step 3 |
| 4.1 | Creating buttons with Modular CSS | Buttons appear all over a site (Go Premium, Learn More, Submit, etc.) and usually have the same general style, but vary in things like color, width, and position. Creating modular light-weight classes to handle this variation can be very efficient. | Explore the different button types we are going to create throughout the final site. | Create Button variation 1 | Create Button variation 2 |
| 4.2 | Multiple Classes | Buttons vary throughout the site, but a good portion of the button style remains the same. Multiple classes on an element will allow variation in our style with an object-oriented approach. | Create multiple classes to vary our buttons | Add the desired classes to our html. |  |
| 4.3 | Specificity Rules | Understanding which selectors overrule other selectors can be confusing. Throughout a site overriding styles tends to be common so understanding specificity and weight is important. | Explanation of the various weights and point system. | Discuss universal selector and !important declaration. |  |
| 4.4 | Transitions | Creating an added experience layer transitioning elements from default to hover state and vice versa | Create a hover state | Create a transition | Discuss vendor prefixes |
| 4.5 | Transforms | CSS3 allows us to transform elements like never before. | Scale | Translate | Rotate |
| 4.6 | Styling a Call to Action button | Most sites need a call to action button to drive users towards a desired result. | Adding the HTML | Styling the button | Adding the hover affect |
| 4.7 | Gradients | Writing Gradient syntax by hand is lengthy and difficult. Use an online tool to automate it. | Plug in start and stop colors in to the gradient generator tool. | Copy the output to the css file. | Explanation of the output. |

**Section5: Creating the Main Navigation and Drop**

Section Description: The navigation is a critical component of any website and can be a veary tricky thing to build with CSS. Throw in a drop down menu and the difficulty level increases. This section simplifies building the navigation, reinforcing the use of floats, while introducing pseudo classes, absolute, relative, and fixed positioning, as well as a few more CSS3 properties, like animations.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Problem/Solution | Step 1 | Step 2 | Step 3 |
| 5.1 | Starting the Nav | How to build a site’s primary navigation? | Create an unordered list in the html to be used as the navigation. | Add in css to make it look like the final product. |  |
| 5.2 | Using Pseudo Classes | How to target elements based on their order of appears without adding a class to the html. | The First-child | The last-child | The nth-child |
| 5.3 | Absolute Positioning | How to get the shark positioned properly? And how to get the menu to be “sticky” to the top and always visible? | Warm up by absolutely positioning the shark. | Use fixed positioning for the nav bar. |  |
| 5.4 | Building the drop down | How do we build the drop down menu? | Create a UL inside of an LI. | Add in the CSS for the drop down menu. |  |
| 5.5 | CSS Animations | Need to smoothly animate the drop down menu sliding down | Define animation-name, duration, timing function. | Add in keyframes | Create more robust animation for the shark logo.  Add in all necessary vendor prefixes. |
| 5.6 | Finalizing the nav | We have a z-index bug, need the box shadow, and need the nav on all pages | Fix the z-index bug | add the drop shadow using the box-shadow property | Copy the nav html to all other pages. |

**Section 6: Becoming Responsive**

Section Description: Responsive Web Design allows you to provide a great experience on all device sizes. This section explains the 3 foundations: fluid grids, flexible images, and media queries. Since mobile is such a big part of responsive web design, we’ll go a step further and really account for some of the differences between mobile and desktop experiences.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Problem/Solution | Step 1 | Step 2 | Step 3 |
| 6.1 | Fluid Grid | Create a fluid percentage based layout | Learn the formula | Apply formula to widths | Apply formula to margins and padding |
| 6.2 | Flexible Images | How to make images squishy like our content | Wrap an image in a container | Make the container fluid | add max-width 100% to the image |
| 6.3 | Media Queries | How to fix remaining issues with site at narrower widths | Add in a media query discuss anatomy | Fix obvious design problems at narrower widths inside the media query |  |
| 6.4 | Mobile Menu | The navigation really is terrible looking at mobile widths and we need a design pattern that will withstand adding more menu items | Create the mobile menu layout | Hide the mobile menu and add in the mobile menu icon. Hide the mobile menu icon for deskop | Add in jQuery to activate mobile menu menu on click or touch |
| 6.5 | Viewport Meta Tag | Need to test device on a phone | Use chrome’s emulator | Add in viewport meta tag | discuss anatomy of viewport tag. |

**Section 7: Web Fonts**

Section Description: For a very long time the web was limited to a small number of web-safe fonts. The alternative was using images to achieve a more unique font family. Web fonts over the last few years have changed the game. This section provides a number of different methods to obtain, and add, web fonts to your site. Also covering icon fonts as an alternative to images.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Problem/Solution | Step 1 | Step 2 | Step 3 |
| 7.1 | @font-face property | There were only so many “web safe” fonts out there. Until Web Fonts came along | Add OTF file to site folder | Define new fonts in css | Apply font to elements |
| 7.2 | Font Kits and Font Services | Web Fonts are kind of hard to make work in all browsers by yourself. Here’s how font kits solve that when hosting your own web fonts. | Download a font kit | Implement css to make it work |  |
| 7.3 | Google Web Fonts | Where to get free quality web fonts? | Find the fonts we want from Google fonts | Add link to font in the <head> of the document | Add the font to different parts of our website |
| 7.4 | Subscription Font Foundries | How to obtain fonts that aren’t free? Subscription fonts foundries | Select a font from Typekit | Add javascript to our site | Apply the font to different parts of our website |
| 7.5 | Icon Fonts | Have a lot of solid color images or icons on your site and want to reduce the number of http requests? Icon Fonts can help. | Build the footer in to the html where the icons will go. | Download Zurb Icon fonts and add the font files to our website. | Add the icon css file to our website and apply the icons to our footer.  Add in an icon hover state. |

**Section 8: Workflow for HiDPI Devices**

Section Description: With the introduction of Apple’s “Retina” screens that cram twice the number of pixels in the same amount of space, as web developers we need to account for these HiDPI displays when it comes to serving images. This section explains how to display “2x” images to HiDPI devices and other techniques for tackling Retina.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Problem/Solution | Step 1 | Step 2 | Step 3 |
| 8.1 | 2x Images | Retina devices make our images look blurry! We need to supply larger images. | Create an image that is twice the size. | Add the image to our html. | Update CSS to size the 2x image down to the intended display size. |
| 8.2 | The Javascript Approach | Non-retina devices are getting a huge retina-sized image when they don’t need it. javascript can help serve the appropriate sized image based the devices capabilities | Check out retina.js and what it can do and grab the script. | Add the script to the js folder and add a script tag that links to it in our html. | Test to make sure its working using Chrome’s Developer Tools. |
| 8.3 | 1.5x images | Double sized images can be 3-4 times as larger as their regular sized counterpart. Creating two images to account for retina and non-retina is a lot of work. There is a middle ground: 1.5x images. | Create an image that is 1.5 times the size it will be displayed at. | Add the image to the html. | Test to make sure it works. |
| 8.4 | Background Images | How do we account for background images on retina? | Create a special media query to determine the device’s pixel ratio | Update the background image inside the media query to serve the 2x image only to retina devices. |  |
| 8.5 | SVG | Is there a perfect solution to the retina trials and tribulations? No, but SVG is pretty darn good. | Save an Illustrator file as SVG. | Add in the SVG file as a background image.  Add in SVG file as a regular image. | Create an inline SVG and use instead of an image. |

**Section 9: Wrapping Up**

Section Description: Next steps and Conclusion

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. | Title | Problem/Solution | Step 1 | Step 2 | Step 3 |
| 1 | Next Steps | We’re done learning css, what to do next? | CSS Preprocessors | Javascript and jQuery |  |
| 2 | Conclusion and Links | Recap of site and recommended links for further learning | Recap of website | Links to my favorite learning resources that have helped me learn web development | Thank you! |