<https://github.com/mattchatterley/coding-programme/blob/master/2.%20Intro%20to%20HTML%20%26%20CSS/notes.md>

**CSS Resources**

* [MDN (Mozilla Developer Network)](https://developer.mozilla.org/en-US/): Your one stop shop for when you need to know ALL information about a particular HTML tag or CSS selector
* [CSS-Tricks Almanac](https://css-tricks.com/almanac/): Well thought explanations of various CSS selectors etc by an industry specialist. Check out his blog articles, videos and snippets as well
* [Can I Use?](http://caniuse.com/): Thinking of using a certain CSS selector or property? This site will let you know what browsers support it.
* [CSS - What not to do](http://www.creativebloq.com/css3/avoid-css-mistakes-10135080)

**HTML Resources**

* [Intro to Semantic HTML](https://internetingishard.com/html-and-css/semantic-html/)
* Know HTML Semantics. [Dive into HTML5](http://diveintohtml5.info/semantics.html#new-elements) - A massive article on HTML5. Perhaps just look at the small section discussing some of the new HTML5 elements and when to use them.
* [Wtf HTML & CSS](http://wtfhtmlcss.com/) - Some common problems that you might encounter when starting out with HTML and CSS

[**CSS-Tricks**](http://www.css-tricks.com/)**is the best site ever for CSS information. No questions.**

* If you're looking for information on CSS, this is the site you should be looking for in your search results
* Sign up for the weekly email, it contains some good stuff and keeps you up to date with new technologies / methodology

**How CSS rules are interpreted**

* How we write CSS (Selectors, Declarations, properties and values. Don't forget the semi-colon!)
* Specificity, what it is and how its calculated
* [CSS-Tricks - Specificity](https://css-tricks.com/specifics-on-css-specificity/)
* For rules with the same specificity, the last one takes precedence
* Be as specific as you need to when defining rules. Try to use a flat CSS structure that doesnt mimic your HTML.
* Use classes as much as possible. Never use IDs (unless you need to reference the element in Javascript)

**Everything is a box.**

* Important to know how the box model works
* [CSS-Tricks - The Box Model](https://css-tricks.com/the-css-box-model/)
* [Treehouse - Box sizing explanation](http://blog.teamtreehouse.com/box-sizing-secret-simple-css-layouts)

**Margin and Padding**

* Think of it as => padding goes inside, margin outside.
* Margin spaces elements.
* Vertical margins collapse. Ie largest of two touching vertical margins will be the space between them.
* Margins need something to push against. They may extend outside of their parent if the parent has no padding.

**Floats**

* Floats are confusing. Re-read the following article 10 times if you have to.
* [All About Floats](https://css-tricks.com/all-about-floats/)
* Understand the clearfix hack used to stop parent height collapse when all children are floated.
* [CSS Tricks Screencast on floats - 27 mins](https://css-tricks.com/video-screencasts/42-all-about-floats-screencast/)

**Positioning**

* Understand how positioning works, specifically relative, absolute and fixed.
* [CSS Tricks - position](https://css-tricks.com/almanac/properties/p/position/)
* Realise that you should probably use absolute and fixed positioning sparingly if you want a responsive design.
* [CSS Tricks Screencast on positioning - 13 mins](https://css-tricks.com/video-screencasts/110-quick-overview-of-css-position-values/)

**Display types**

* Know the main ones and what they mean: block, inline-block, inline, and none.
* [CSS Tricks - display](https://css-tricks.com/almanac/properties/d/display/)

**Other**

* Most CSS issues in group projects come from overlapping styles.
* Break CSS rules down into many files that each relate to a specific element or group of elements. Ordered CSS rules are much easier to work with. A good rule of thumb is max 100 lines per file.
* Don't style bare elements. E.g applying a default font-size of 24px to a general 'p' tag will likely cause problems.
* Vertical positioning is a nightmare
  + Try setting all elements to be inline-block and have the same line height
  + Try Flexbox (more on this later)
* Height 100% will require height: 100% on all parents (including body and html tags) in order for it to work

**Style Guides**

* [AirBnB CSS Style Guide](https://github.com/airbnb/css)

**Other things that we will cover in more detail later**

* Flexbox
* Pseudo Classes
* Transitions
* Transforms
* Sizing of elements using px, %, em, rem

**Before we start to code**

* Does this have to be responsive? (What screen size are we designing this for and whats the target audience)
  + Yes it should be responsive.
  + Try to use widths as percentages.
  + An easy responsive design can be made by defining a content wrappers with a fixed width. Alter this with media queries (more on this later).
* Do we need to worry about users using touch?
  + This might affect sizing of elements that we want them to click.
* Are we working from a design?
  + If so, pay attention to detail. Designers may have put a lot of work into getting things just right.

**Starting to code**

* Understand browsers default styles (specifically font size and margin on the body)
* My preference is to code up most HTML first, then do basic styles for layout before concentrating on specific sections.
* Use developer tools to help you when you have issues
* Naming conventions
  + Do not use classes that describe the content of an element. ie 'blue-background' is bad
  + Give abstract names like 'entries' rather than 'products-list', this way the class is more reusable and can accommodate change
  + Names should be lowercase with words separated by a hypen (dash)
  + Try not to abbreviate, full descriptions are usually best for readability ie 'main-nav' is preferred to 'mv-nv'