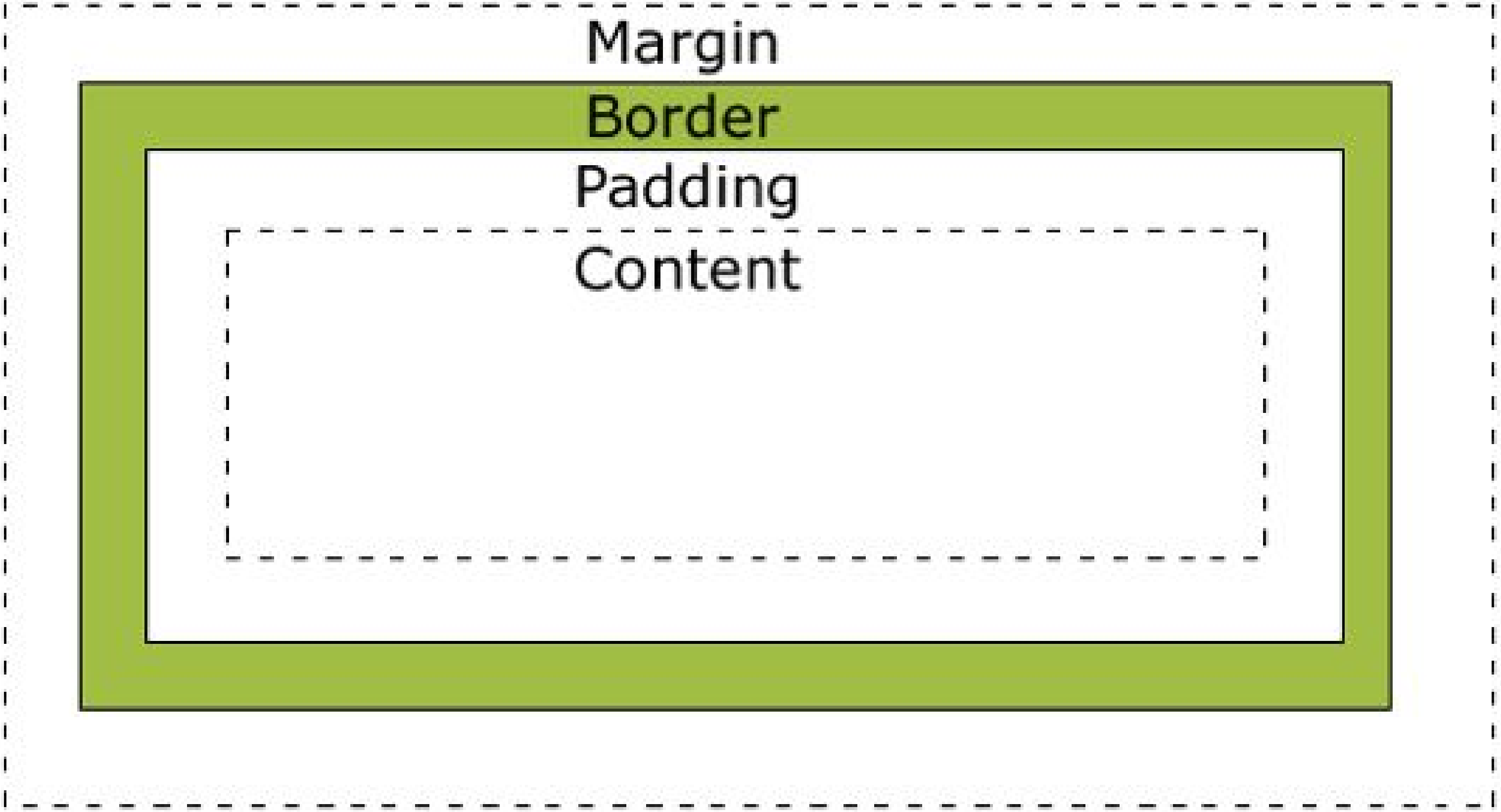
CSS Positioning && GitHub Pages

The Box Model

In CSS, every element rests within a series of boxes.
Each box has customizable space properties:
margin, border, and padding.


Margin
Border
Padding
Content

Activity! Box Model (10 min)

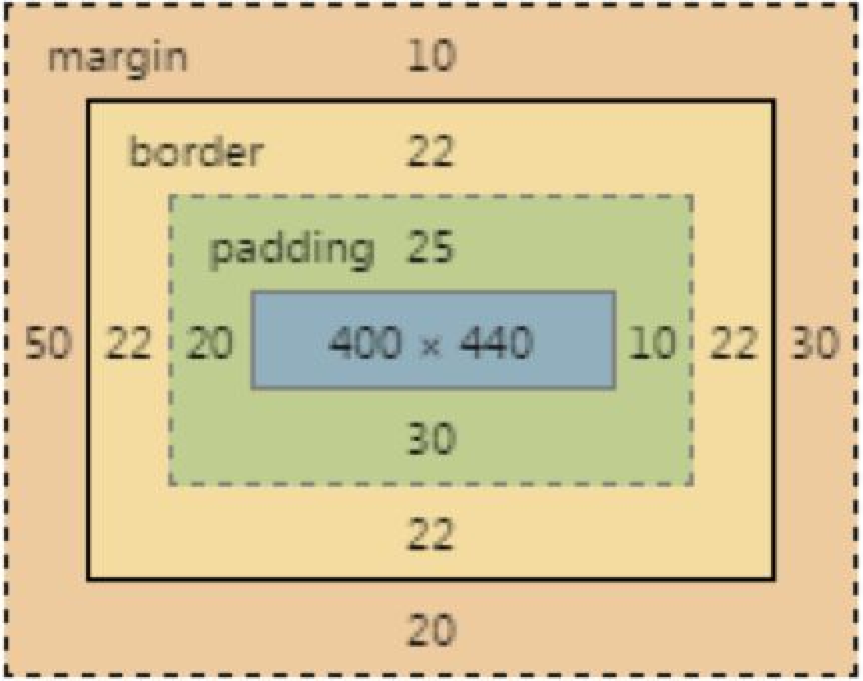
```
#box {
    width: 400px;
    height: 440px;
    margin: 10px 30px 20px 50px;
    padding: 25px 10px 30px 20px;
    border-style: solid;
    border-width: 22px;
    border-color: #113152;
}
```

How wide is the box?

How tall is the box?

Review: Box Model

- Width:
  - 474 px (no margin)
  - 554 px (with margin)
- Height:
  - 539 px (no margin)
  - 569 px (with margin)



Positioning

Demo: CSS Positioning

- relative
  - Positions elements relative to their static location in the document. These elements behave with and interact with other elements the same way they would as if they were positioned statically, except that you can use the top/right/bottom/left properties to move the elements after they have been placed into the document flow.

Demo: CSS Positioning

- absolute
  - Positions elements relative to the nearest positioned ancestor (non-static). They are taken out of the flow of the document, taking up no space when placing other elements. These elements will move in the viewport as you scroll (unlike fixed).

Demo: CSS Positioning

- fixed
    - Positions elements relative to the top left of the browser window. Similar to absolute, except the containing block is the whole viewport. These elements will remain in the same place in the viewport as you scroll.

Demo: CSS Positioning

- z-index
    - Allows us to position elements on top of one another.

Activity! CSS Positioning (30 min)

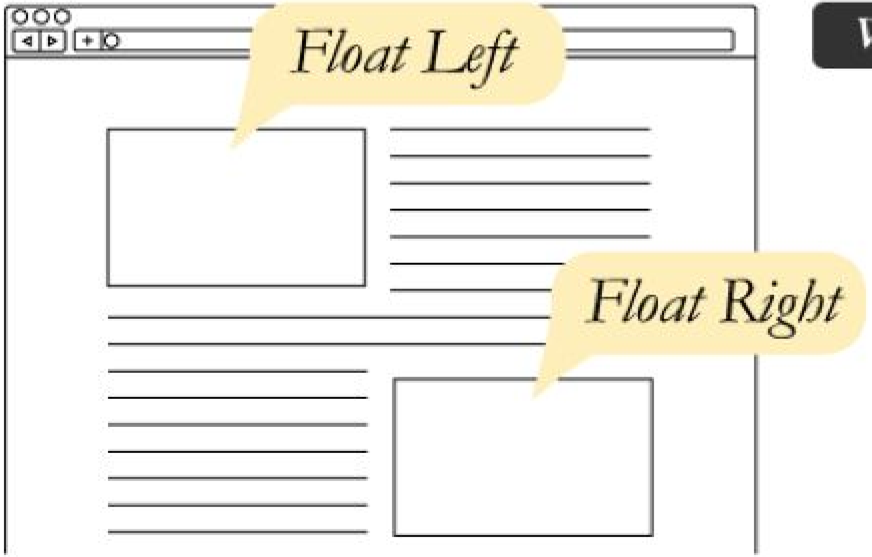
See 11-CSS_Positioned_Activity in the class repo for instructions

Netflix && Chill

https://youtu.be/sHfJn0jqBro

Flow

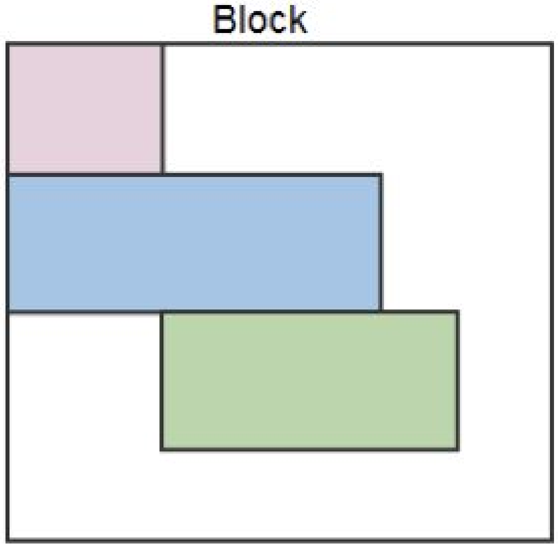
- By default, every HTML element displayed in the browser is governed by a concept called flow.
- This means that HTML elements force their adjacent elements to flow around them.
- This concept of “flow” is very similar to the wrap-text options you may be familiar with in Microsoft Word.
- Just as in MS Word, you can have images in-line with text, on-top of text, etc.



Web Layout

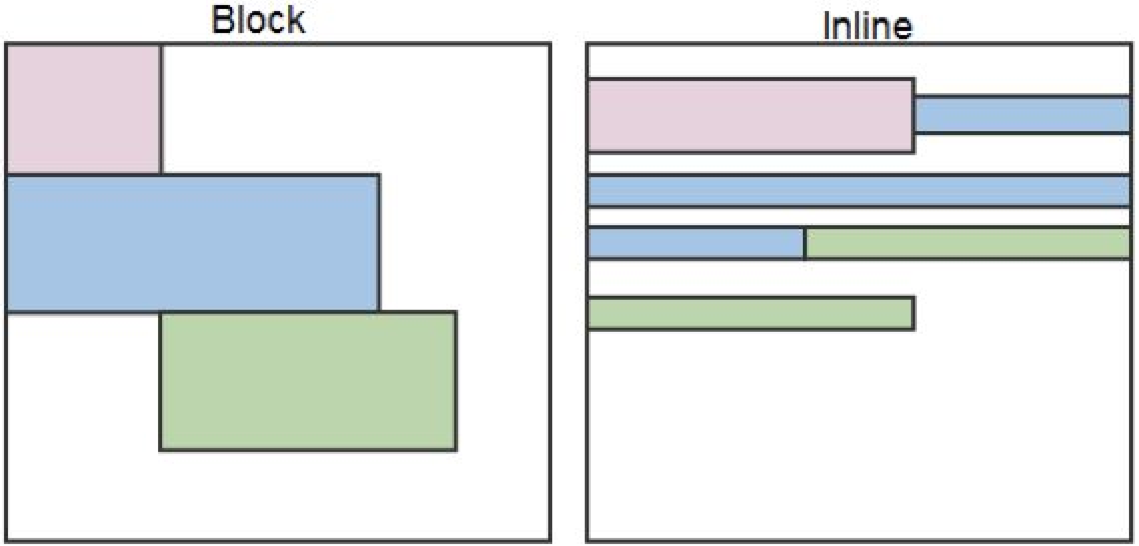
Block Elements

- By default, web clients render many HTML elements as block elements. Paragraphs, headers, divs and more receive this treatment.
- A block element will take up an entire line of space—unless you intervene with CSS properties.



Block Elements vs Inline Elements

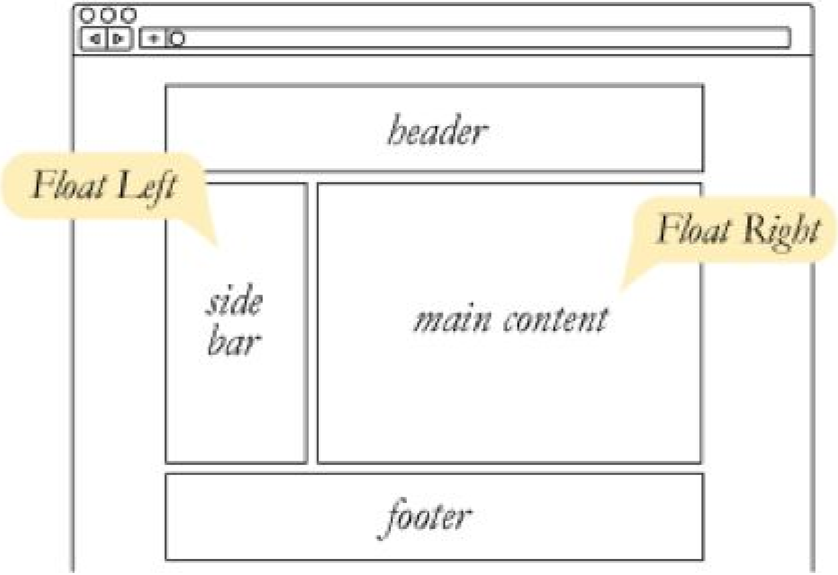
- Now contrast the block elements with inline elements.
- By using float CSS properties, we can command our website to display multiple HTML elements adjacently.



Float

- To transform these block elements into inline elements, we use a CSS property called float.

- Floats are necessary for building web layouts.



Demo: Floats

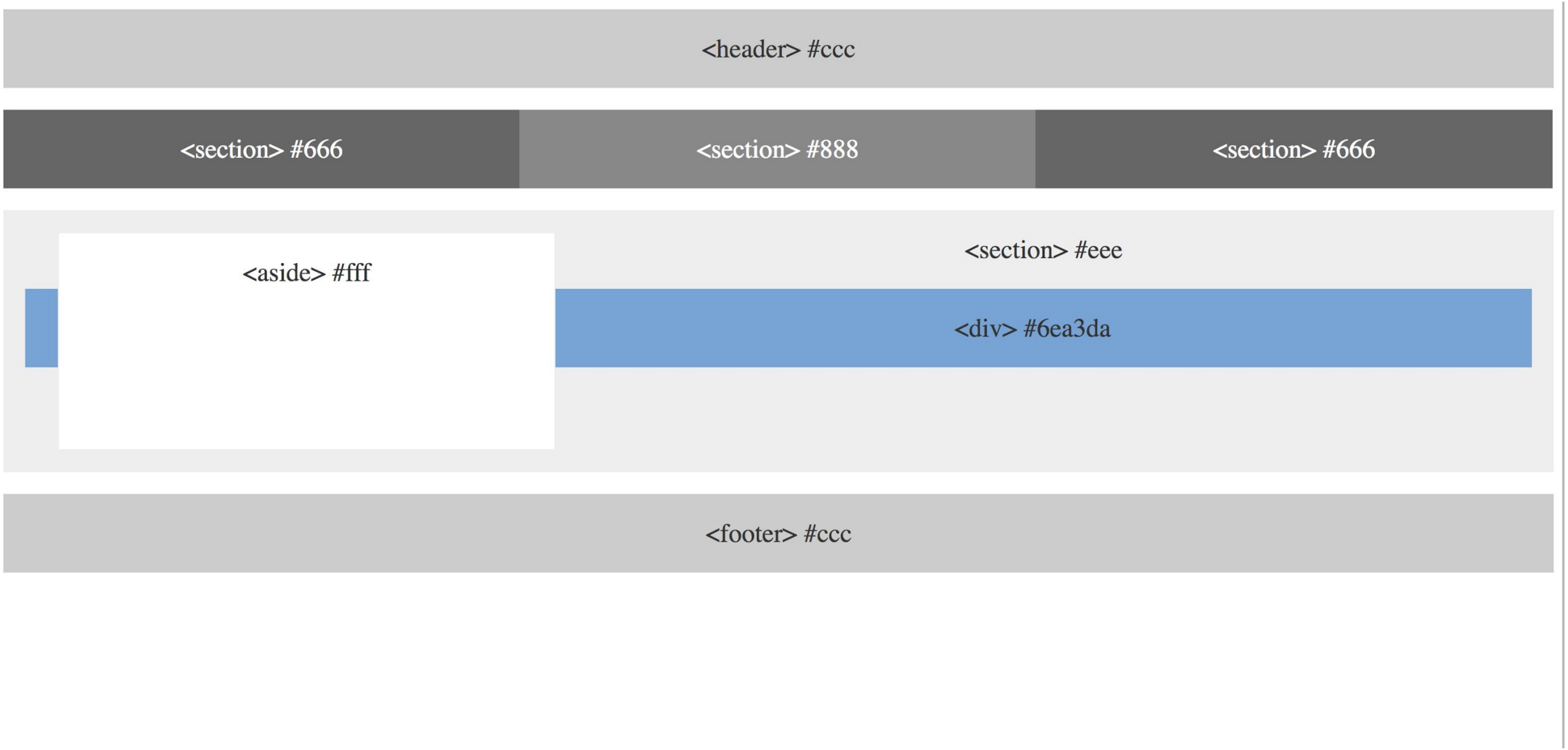
08-FloatExamples

See also https://css-tricks.com/all-about-floats/

Activity! Floats

See 09-FloatLayout-Activity

See also https://css-tricks.com/all-about-floats/



<header> #ccc

<section> #666

<section> #888

<section> #666

<aside> #fff

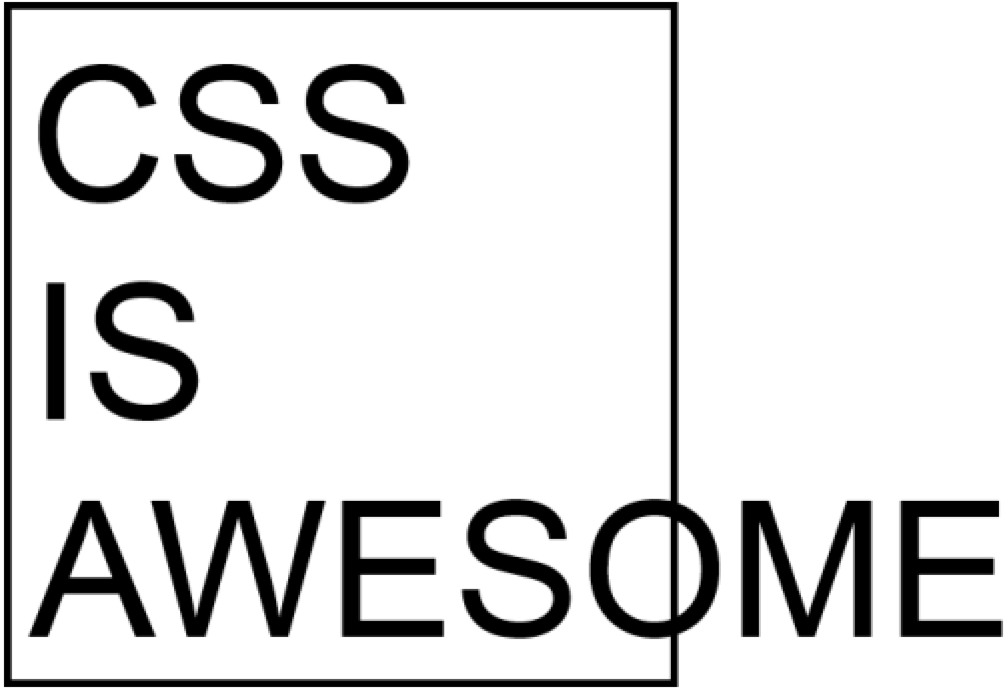
<section> #eee

<div> #6ea3da

<footer> #ccc

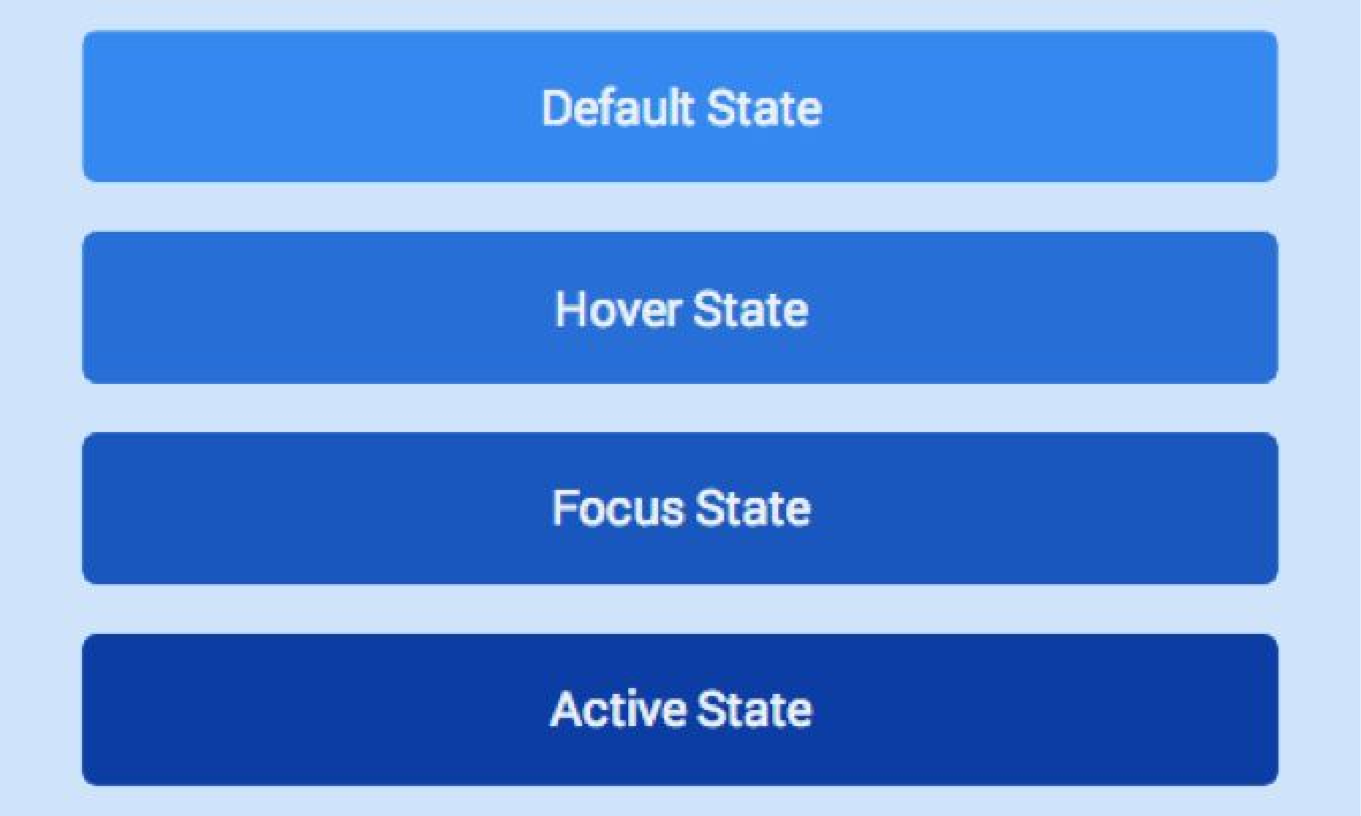
Netflix && Chill

https://youtu.be/0lpxKw6E90Y



CSS
IS
AWESOME

Pseudo-classes


Default State
Hover State
Focus State
Active State

Demo: Pseudo-classes

09-Pseudoclass

- CSS has keywords that can be added to selectors. These highlight the special states of the selected element.
- Great example: CSS can hook onto the specific moment when a link is hovered over using the :hover pseudo-class.
- Complete list of pseudo-classes found here: https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-classes

Activity! Pseudo Styles

See 09-Pseudoclass for instructions

Review: Pseudo Styles

RTFM https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-classes

Multiple Style Sheets

Demo: Multiple CSS

03-MultipleCSS

Resets

Cross-browser compatibility is critical in web development

But.

Each browser has a built-in, default style.

We can incorporate a `reset.css` file that ensures your site will look the very same in different browsers.

Demo: reset.css

http://meyerweb.com/eric/tools/css/reset/

reset.css is important because of:

- Cross Browser Compatibility
- Ability to use pre-made CSS from someone else (an idea we will exploit repeatedly)
- It's a common front-end question

Activity! CSS Resets

See 04-ResetCSS in the class repo for instructions

Developer Tools

Demo: Chrome Inspector

- Visit google.com
- Right click (MS) || CTRL click (Apple)
- Select Inspect
- Click Element Selector

Activity! Hack Websites (15 min)

- For the next 15 minutes, take a website that you commonly use (Amazon, Google, Huff Po, etc.) and heavily modify it using the Chrome Developer Tools.
- Be sure to at least modify:
  - Content (Change words)
  - Colors
  - Spacing
  - Etc.
- Slack out a screenshot to #class-instruction when you’re done.

Deploying to GitHub Pages

In order for our websites to be accessible by the public, they need to be deployed on a server.

What’s a host?

A web host is the activity or business of providing storage space and access for websites. You cannot put a website online without it being hosted on a server somewhere.

RTFM https://pages.github.com/

Demo: Deploy

- Create a new repository on your GitHub account. You can name this repository whatever you would like.
- Once inside of the repository, create a new file and name it index.html
- Add some very basic HTML into this file, save it, and then navigate into your repository's Settings tab.
- Scroll down to the GitHub Pages section and then, in the section labeled "Source", select that you would like to use the master branch as your source.
- Navigate to _username_.github.io/_repositoryname_ and you will find that your new web page has gone live!

Activity! Deploy

Create a basic portfolio and deploy it to GitHub pages

See Also

http://learn.shayhowe.com/html-css/positioning-content/