

FWDP 1000 – Day 3

Course: Web Development 1

Instructor: Gabbie Bade

Morning Review

- Download the files.
- Open the HTML file and CSS file in your code editor.
- Read the comments in the files and complete the tasks.

We will go over this in 15 minutes.

Agenda

- Connecting to GitHub
- CSS Box Model
- Developer Tools
- Mobile First CSS
- Normalize.css
- Pseudo Classes
- Assignment #3

Connecting to GitHub

From Local to Online

NOTE: It is typically easier to initialise an empty repository online first and then ***clone*** it locally. For practice purposes, what we are doing now is ***pushing*** up to GitHub.

- Login to your GitHub account and click **New** and name your repo: **web-dev-1**.
- In your code editor, switch to branch master.
- Copy and paste the second set of code from GitHub in your terminal. Take note that this set of code renames your branch **master** → **main**.

Pushing the branches

- You can either do:
 - One branch at a time, switch to branch day-1 and day-2 and type the following in terminal:

```
git push -u origin <branch-name>
```

- All branches at once.

```
git push --all
```

You can also simply click the cloud icon beside the branch name. This will likely open a dialog box in your code editor that asks you to login to your GitHub.

The built-in GitHub extension in VSCode is very easy to use and navigate.
Explore!

Pushing changes

- Switch to **main** and make a new branch called **day-3** and **push it** to your GitHub repo.
- Next, while on branch day-3, update the files. To commit the changes, we need to do one more step:

```
git add .  
git commit -m "descriptive commit message"  
git push origin <branch-name> or just git push
```

Merging into your Main branch

- When your branch is approved with production ready code, merge to the master branch.
- First switch to the master branch: **git checkout main** or **git switch main**
- Then merge the code from the new branch to the master branch:
git merge <branch-you-want-to-merge>
git push origin main

Typically, you can just merge branches **into main** directly. When we get to Web Dev 2, we will discuss how to protect the main branch to avoid accidentally overwriting code when working in a group.

Practice, practice, practice

Over the next few days, we will stick to just:

- Making, switching, and pushing branches
- Adding, commit, and pushing changes

It may be confusing and intimidating now, but the more you do it, the easier it gets!

CSS Box Model

CSS Box Model

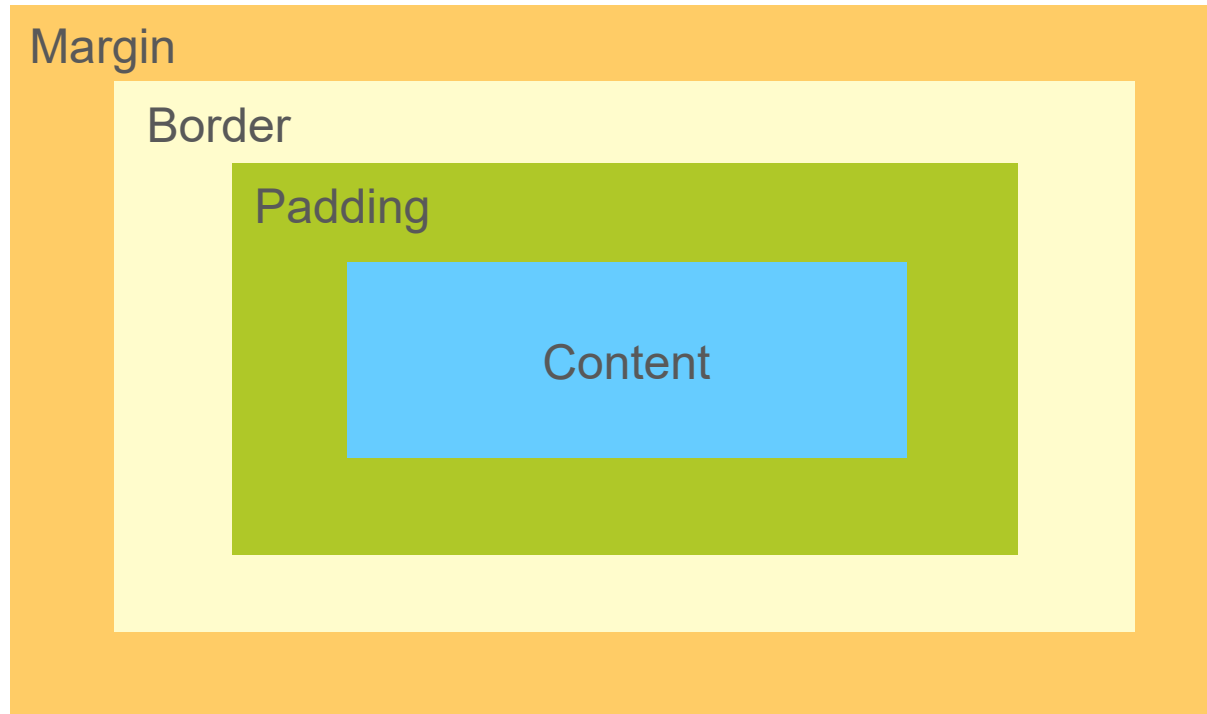
The CSS Box Model is the set of rules that CSS uses to determine the size of block-level elements.

The width and height of an element is determined by the addition of the following areas:

- Content + Padding + Border + Margin

https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Box_Model/Introduction_to_the_CSS_box_model

CSS Box Model



Block-level Elements

Block-level elements...

- Have a default width of 100% of their parent's width. This width can be changed with CSS.
- Cause a line break.
- Height is determined by the content. This height can be changed with CSS.

Inline Elements

Inline elements...

- Do not cause a line break.
- Width is determined by the content inside their tags.
- No width or height can be set on them in CSS.

Changing Display Values

Block-level elements can be made inline and inline elements can be made block using CSS.

```
a.block {  
    display: block;  
}  
p.inline {  
    display: inline-block;  
}
```

← Any <a> tag with the class “block” will display like a block-level element.

← Any <p> tag with the class “inline” will display like an inline element.

<https://developer.mozilla.org/en-US/docs/Web/CSS/display>

Box Sizing Reset

The default box model in CSS is called “content-box” and is very confusing.

Example: You set an element’s width to 25%. Then add padding and a border to that element. It is now 25% **plus** the padding and border, potentially breaking your layout.

Because of this, we use a reset at the top of our CSS to switch to the “border-box” model.

Example: You set an element’s width to 25%. Then add padding and a border to that element. It is still 25% because the padding and border is added **within** that 25% width.

Box Sizing Reset Code

Add the following code at the top of your CSS files going forward...

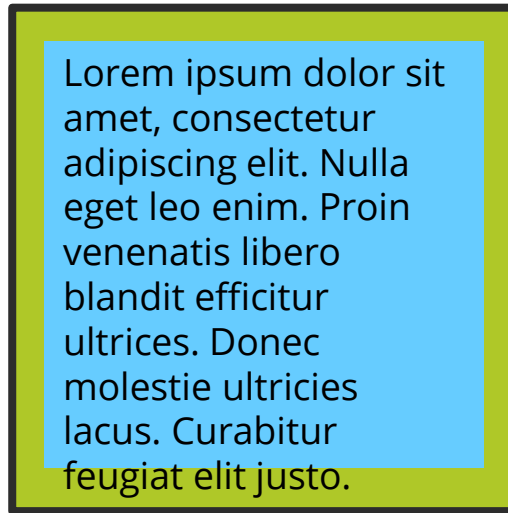
```
html {  
    box-sizing: border-box;  
}  
*, *:before, *:after {  
    box-sizing: inherit;  
}
```

<https://css-tricks.com/inheriting-box-sizing-probably-slightly-better-best-practice/>

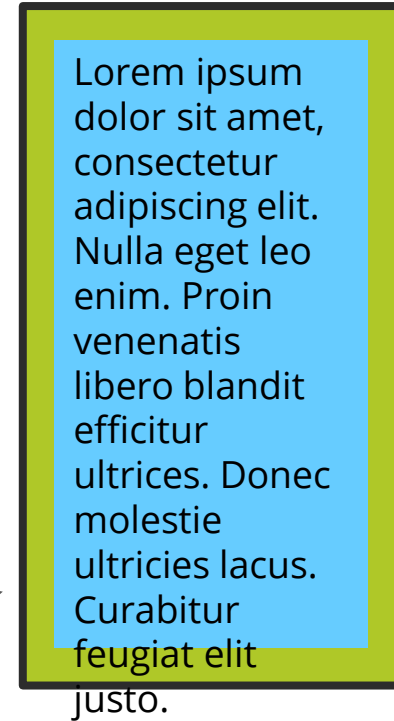
Border-box vs Content-box

If we set the width in CSS to 200px using the default “content-box” and then add padding and border, the width is actually larger than 200px because...

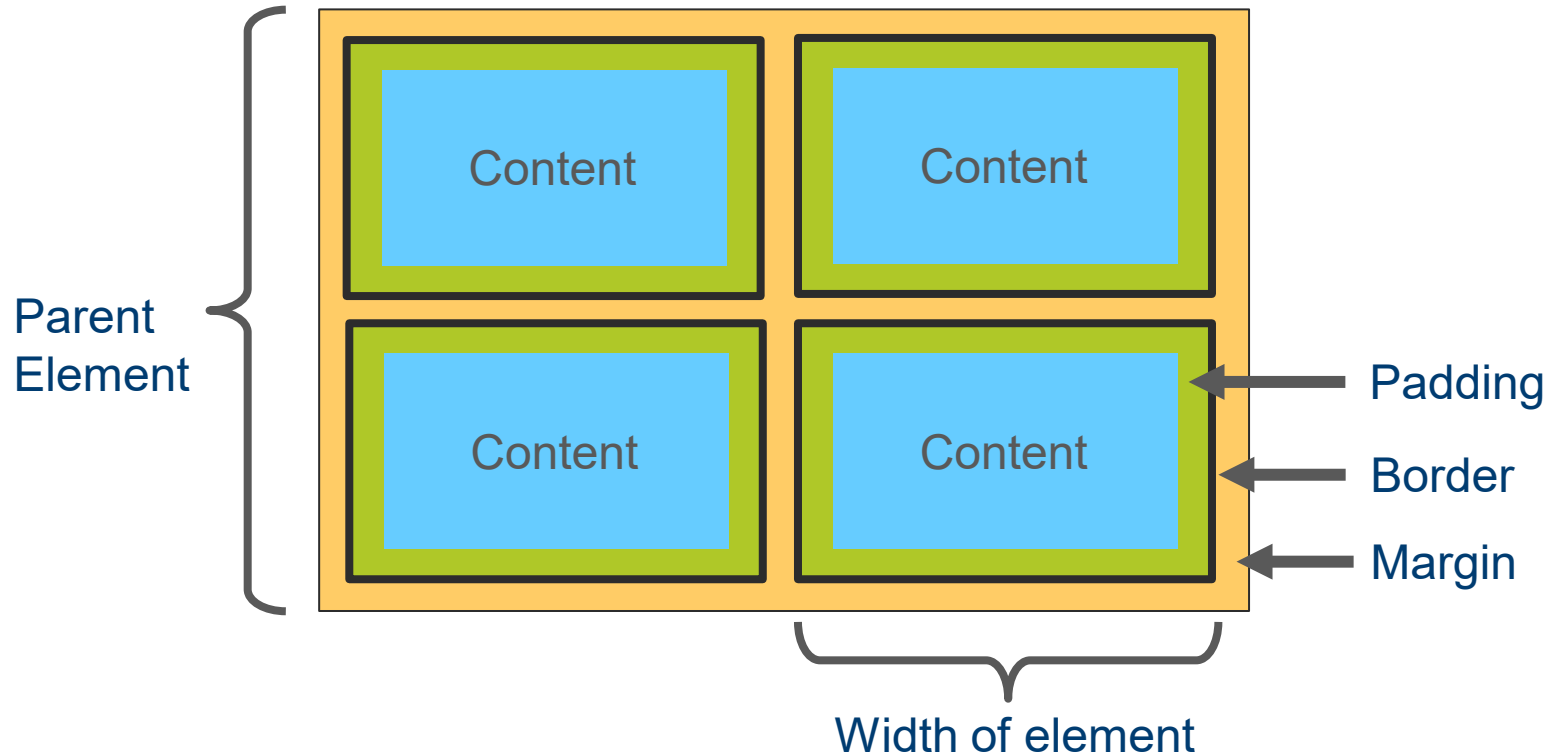
width + padding + border



Using the same CSS but switching to use “border-box”, the width stays at 200px.



CSS Margin, Padding, Border



CSS Margin, Padding, Border

Border is the line around the HTML element.

Padding is the space between the content and the border.

Margin is the space **around** the HTML element. It is not included in the width of the element.

CSS Margin & Padding

When setting margin or padding on block-level elements, you can write your CSS one of two ways.

```
.item {  
    margin-top: 1rem;  
    margin-right: 1.5rem;  
    margin-bottom: 1.25rem;  
    margin-left: .75rem;  
}
```

```
.item {  
    margin: 1rem 1.5rem 1.25rem .75rem;  
}
```

CSS Shorthand

The CSS shorthand for margin and padding can handle the following scenarios:

- All sides have the same value
- Left and right have the same value, top and bottom have the same value
- Left and right have the same value, top and bottom have different values
- Top, right, bottom, and left have different values

<https://developer.mozilla.org/en-US/docs/Web/CSS/Margin>

<https://developer.mozilla.org/en-US/docs/Web/CSS/Padding>

CSS Shorthand – Example 1

Example 1: All sides have the same value.

```
.item {  
    margin: 1.25rem;  
}
```

The top, right, bottom, and left will all have a margin of 1.25rem, or 20px for most browsers.

CSS Shorthand – Example 2

Example 2: Top and bottom have the same value, left and right have the same value.

```
.item {  
  margin: 1rem .5rem;  
}
```

The first value represents the top and bottom margins.

The second value represents the left and right margins.

CSS Shorthand – Example 3

Example 3: Top and bottom have different values, left and right have the same value.

```
.item {  
  margin: 1rem .5rem 2rem;  
}
```

The first value
represents the
top margin.

The second value
represents the left
and right margins.

The third value
represents the
bottom margin.

CSS Shorthand – Example 4

Example 4: Top, right, bottom, left have different values.
Notice it moves clockwise in this scenario (see next slide).

```
.item {  
  margin: 1rem .5rem 2rem .75rem;  
}
```

The first value
represents the
top margin.

The second value
represents the
right margin.

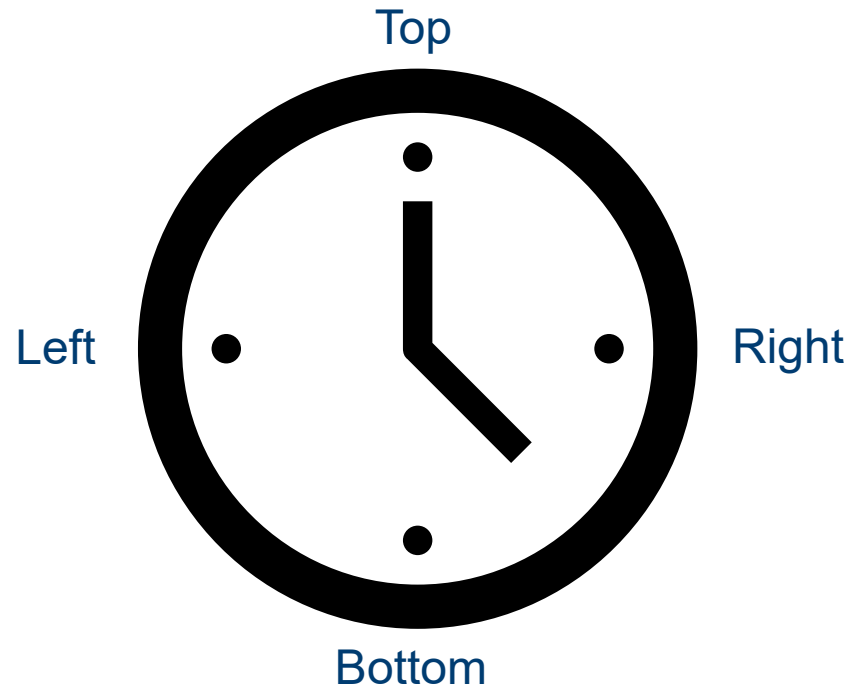
The third value
represents the
bottom margin.

The fourth value
represents the
left margin.

CSS Shorthand

```
.item {  
  margin: 1rem .5rem 2rem .75rem;  
}
```

When all sides have different values, you can think of the CSS shorthand as a clock moving from 12 to 3 to 6 to 9.



Recap

True or False: In the commonly used "border-box" model of CSS, padding and border are included within the element's width and not outside of it.

If all FOUR values are set on the CSS "margin" or "padding" property, what does the THIRD value represent?

Top Bottom Left Right

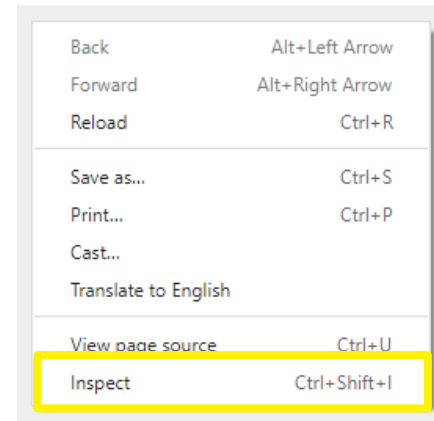
Developer Tools

Dev Tools in Browsers

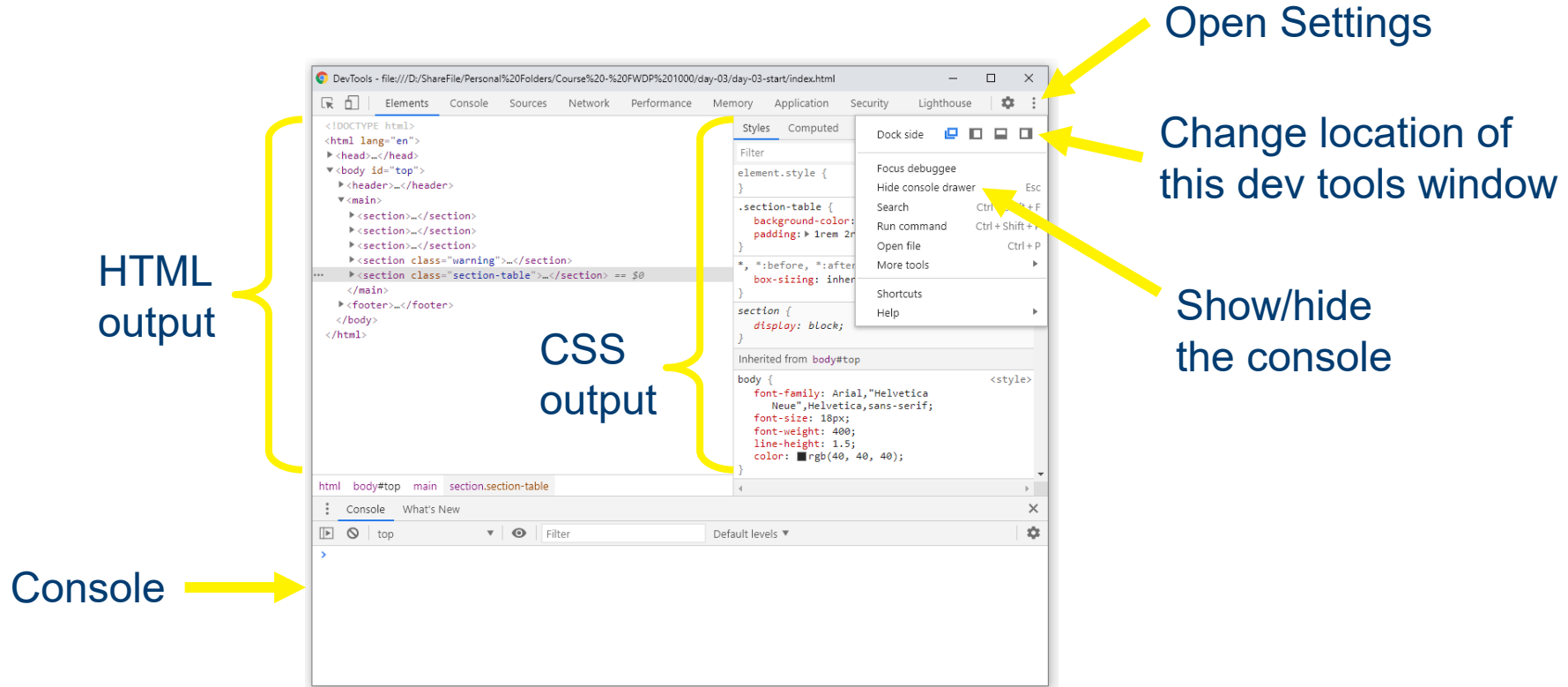
Every browser has a version of Developer Tools to help you troubleshoot and build websites.

It can be opened by pressing F12 or right-clicking the page and choosing Inspect.

They are similar across browsers but Chrome and Firefox have the best developer tools.



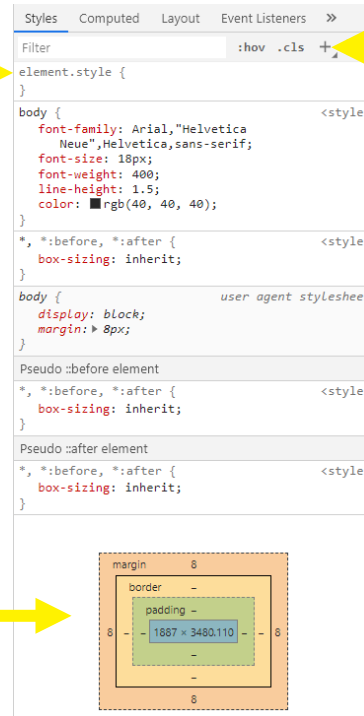
Chrome Dev Tools



Testing CSS in Chrome Dev Tools

Write inline CSS on whatever element you have targeted in the HTML

See the margin, border, padding and width/height of the selected element

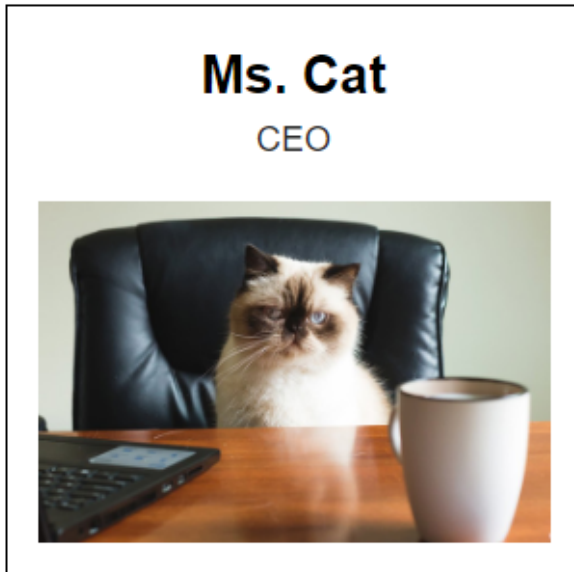


Write new CSS properties with the + symbol

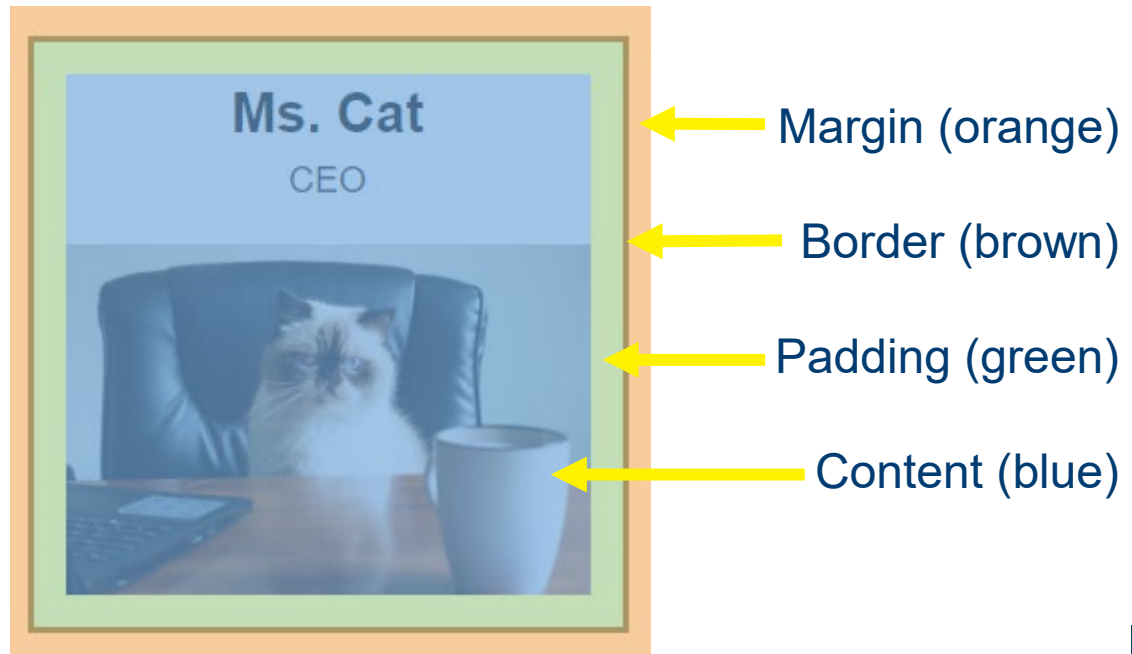
See all of the CSS applying to the selected element and the hierarchy

Inspecting Elements

What it looks like
in the browser...

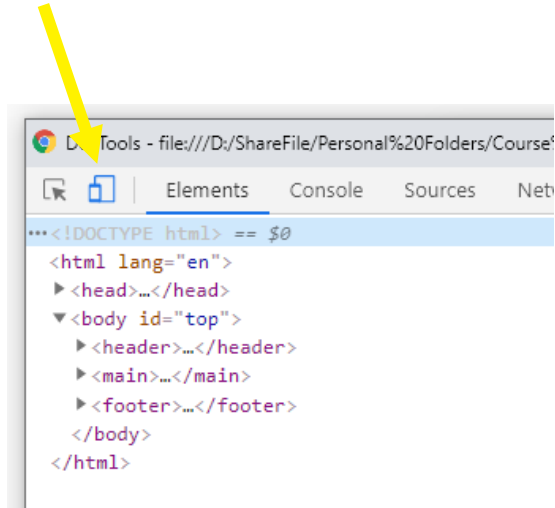


What it looks like when
inspecting in dev tools...



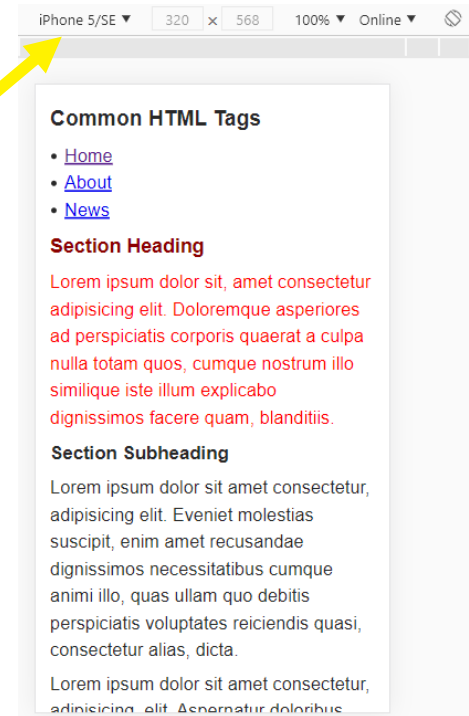
Responsive Layouts Tool

Click here to open the device toolbar



You can select devices to see what the webpage will look like on them.

Or choose “Responsive” to adjust to any size by dragging the side or bottom.



Developer Tools

Nothing you do in the developer tools will save anywhere.

As soon as you refresh your page, everything will be lost.

It is an **extremely** useful tool for testing, troubleshooting and understanding.

Mobile First CSS

Mobile First CSS

Over 55% of all page views on the web come from mobile devices.

Another ~2% come from tablets.

<https://gs.statcounter.com/platform-market-share/desktop-mobile-tablet/>

Mobile First CSS

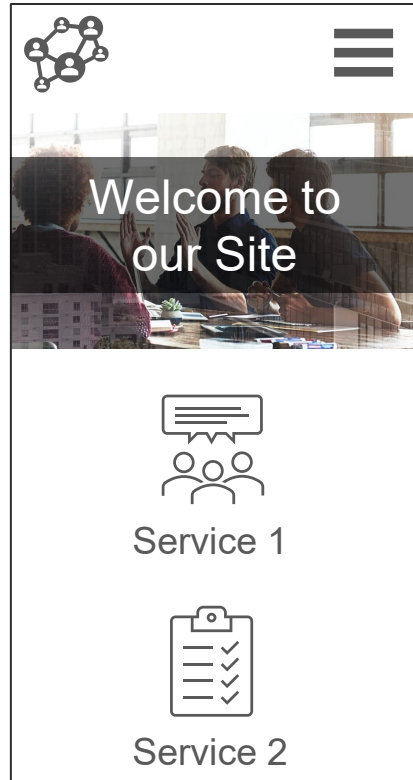
When we write our CSS, we write it “mobile first”.

This means we write our styles according to how the webpage will look on smaller devices, **then** we write styles for how it will look on laptops and desktops.

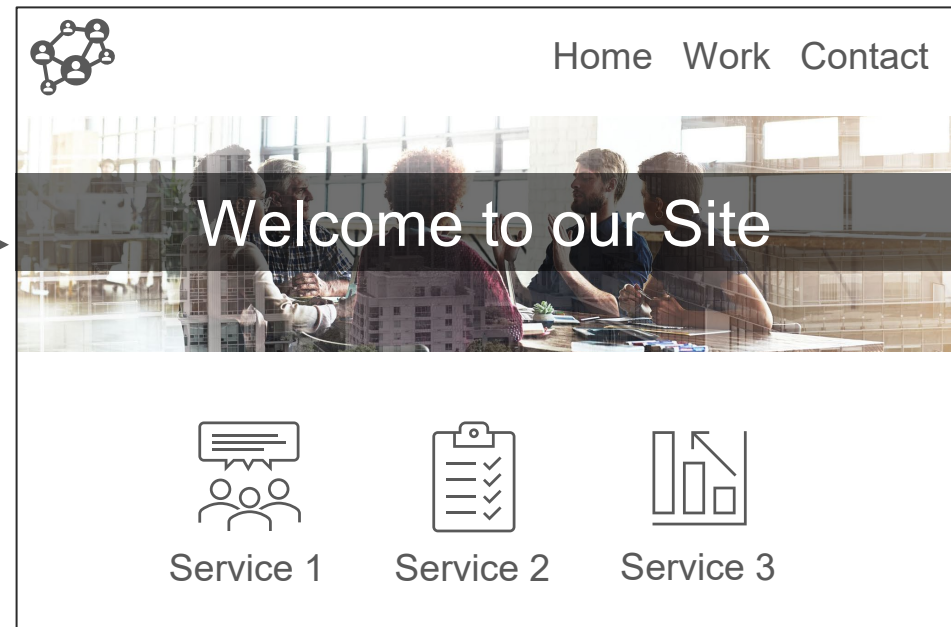
Larger screens involve more advanced layouts, we will cover how to do that with CSS later.

Desktop vs. Mobile Layout

Mobile layouts are usually a single column because of the narrow screen width.



Desktop layouts are usually multiple columns because of the wider screen width.



Force a Mobile Size Only

To make testing easier, let's set a **max-width** on the **body** element for our index.html file. *(We will delete this later.)*

We can also center the body element after setting the width by using the `margin-left: auto` and `margin-right: auto` trick.


```
body {  
  max-width: 31.25rem;  
  margin: 0 auto;  
}
```


Normalize CSS

Cross Browser CSS

All browsers have default styles that they apply to some HTML elements. These styles are not always the same from one browser to the next though.

Example of the default styles in Chrome for an `<h1>` element



```
h1 {  
  display: block;  
  font-size: 2em;  
  margin-block-start: 0.67em;  
  margin-block-end: 0.67em;  
  margin-inline-start: 0px;  
  margin-inline-end: 0px;  
  font-weight: bold;  
}
```

user agent stylesheet

To address this, developers tend to use either Eric Meyer's CSS Reset or Normalize.css.

Normalize vs Reset

The CSS Reset removes the browser defaults entirely.

<https://meyerweb.com/eric/tools/css/reset/>

Normalize.css instead tries to make the defaults consistent so we can still take advantage of the defaults.

<http://necolas.github.io/normalize.css/>

Normalize.css gives us less work and provides nice fallbacks for any element we didn't style, so we will use that.

Normalize.css

To use **normalize.css** all you need to do is download it as a **.css** file and attach the CSS file to your HTML file.

Normalize.css should be attached **before** any other stylesheets so your styles can override these defaults.

```
<head>
  <meta charset="utf-8">
  <title>Page Title</title>
  <link rel="stylesheet" href="styles/normalize.css">
  <link rel="stylesheet" href="styles/styles.css">
</head>
```

Responsive Media

To make sure your images and a few other elements don't expand outside of their parent elements, add the following code to your normalize.css file...

```
embed,  
iframe,  
object {  
    max-width: 100%;  
}  
  
img,  
video {  
    max-width: 100%;  
    height: auto;  
}
```

Rename **normalize.css**

Since we have modified the original **normalize.css** file, we should rename it so we know it is different than the original.

I will rename mine to **normalize-fwd.css**.

Remember to update your HTML file(s) with the new file name too!

Going forward...

Consider moving the box sizing reset code from earlier to the bottom of your normalize.css file.

Now, you can use this normalize.css file on all of your webpages for any project.

Pseudo-classes

CSS Pseudo-classes

A pseudo-class is a keyword added to a CSS selector that specifies a special state.

For instance, when someone hovers over an <a> element, we can change the color...

```
a:hover {  
    color: red;  
}
```

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

Common Pseudo-classes

- `:visited` – style an `<a>` element if the user has already visited the URL.
- `:hover` – style any HTML element when the user hovers the mouse over the element. Commonly used for links and buttons.
- `:active` – style any HTML element when the user is holding down the mouse button. Commonly used for links and buttons.
- `:focus` – style any focusable HTML element once it has been clicked into or selected with the keyboard Tab key. Commonly used for links, buttons and form fields but any HTML element with the “`tabindex`” attribute can accept focus.

Similar Pseudo-classes

There are two more “focus” pseudo-classes for specific situations:

- `:focus-visible` – lets the browser only apply focus styles if keyboard navigating, instead of using a mouse.
- `:focus-within` – style a parent element if any of it's children have focus.

There are [many other pseudo-classes](#). You will cover some of them in Web Dev 2.

Anchor Link Order

If using these pseudo-classes on your <a> tags, you should put them in the following order:

```
a {}  
a:visited {}  
a:hover {}  
a:focus {}  
a:active {}
```

<https://developer.mozilla.org/en-US/docs/Web/CSS/:active>

Google & AI Tools

Finding Code Online

Every developer uses Google to troubleshoot, find code snippets, and learn new techniques.

When you're new to all of this, you won't really know if what you find online is good advice or good code.

You are learning the fundamentals in the FWD program for a reason... so you know what is good and what is bad.

ChatGPT, GitHub Copilot, etc.

The AI tools are no different than using a random code snippet you find online...

...some of the code is good and some is bad.

Any code you add to your assignments and projects, you should be able to explain!

These are all tools

Google, Stack Overflow, ChatGPT, etc. are all tools to help you with learning and productivity.

Be skeptical of the answers you find. Test and verify the code and techniques you use are the proper and modern way.

Assignment #3

Organizing CSS Files

Remember, the order you write your CSS matters so keep it organized.

Put your most general styles first and your more specific styles later.

Assignment #3 – CSS

Look at the screenshot/mockup and figure out which styles can be applied throughout the page...

... put these styles near the top of your CSS.

Then find the styles that are for a specific section or component...

... put these styles later in your CSS.

CSS Organizing Suggestion

When styling a component on the webpage like the header, the footer, or a specific section...

... put all of the styles for the component together and use CSS comments to organize your CSS file.

```
/* Section - Services */  
.services { background: red; }  
.services h2 { font-size: 2rem; }
```

Assignment #3

- Please refer to Assignment #3 in the Learning Hub.
- To submit the assignment, you can do **one** of these:
 - Have me check your assignment in class before 4pm.
 - Zip today's **folder** and upload it to the Learning Hub before next class.
- If you have questions or need guidance, just ask!

Working Together

You can work on and submit this assignment alone, in pairs, or in a group of 3.

I encourage you to work with classmates that you haven't worked with yet.

You will have multiple group projects throughout the program, so start to get to know each other now!

Practice Git & GitHub

This is not required but highly recommended.

Practice adding and committing until it becomes a habit.
Git/GitHub is incredibly helpful especially for more complex projects.

Practicing it even in simple projects make it less daunting!

Resources

Rems vs Pixels

<https://www.joshwcomeau.com/css/surprising-truth-about-pixels-and-accessibility/>

Box-sizing Reset

<https://css-tricks.com/inheriting-box-sizing-probably-slightly-better-best-practice/>

Normalize.css

<http://necolas.github.io/normalize.css/>

MDN Web Docs – CSS

<https://developer.mozilla.org/en-US/docs/Web/CSS>

Resources

Git Cheat Sheets

<https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet>

<https://education.github.com/git-cheat-sheet-education.pdf>

<https://dev.to/ruppysupply/git-cheat-sheet-with-40-commands-concepts-1m26>

QUESTIONS & ANSWERS