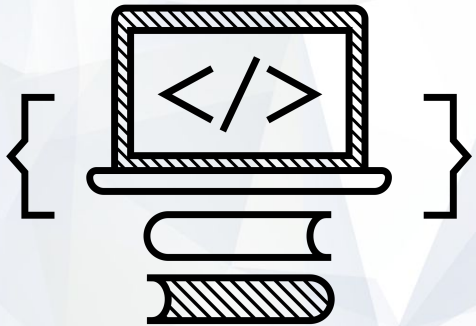
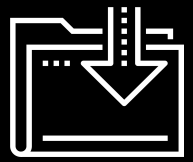


CSS Layout and the Box Model

Skills Bootcamp in Front-End Web Development

Lesson 1.3




WELCOME

Housekeeping

- Breakout rooms -> we'll use countdown timers
  - no more surprise teleportations 😅
- Asking verbal questions in Zoom - please use raise hand feature ✋
  - helps minimize people talking over each other
- Communication channels -> TA Chelsea will chat about this

More housekeeping

- We’re aware that Gitlab and Github confusion is still there, and we’re going to address that today.

- Also, where should your code live on your computer? We’ll cover that too.

Today’s Objectives

By the end of class, students will be able to:

Set an element as an inline or block element using the CSS **`display`** property.

Determine how an element is positioned in an HTML document using the CSS `position` property.

Identify the box model and its role in web design and layout.

*** We’ll also be covering git and github**

Instructor Demonstration

Get Started with Git

Visit the new class repo (on github)

https://github.com/skills-bootcamp/frontend-dev

We'll be using this for all class activities.

Once cloned- you can put all your
in-class activities in this directory.

Code along

1. Open terminal

2. git clone git@github.com:skills-bootcamp/frontend-dev.git

Quick walk-thru of repo.

What if we want to ‘pull’ the latest changes?

Make sure you are on the ‘main’ branch, type:

```
git checkout main
```

Then:

```
git pull
```

How about sharing our changes with others?

If you are working with other people-
NEVER ‘push’ on the ‘main’ branch. Let’s create your own branch.

```
git checkout -b <your-branch-name>
```

for class let’s use our names, so if your name was derek zoolander:

```
git checkout -b derek-zoolander
```

continued...

the `-b` in `git checkout -b` is only necessary when you first create the branch.

After creation, you can simply use `git checkout <branch-name>`

*remember: whenever you see `<words inside carrot brackets>` that text is placeholder text, don’t copy it, be creative, come up with your own text.

Now make a change on your branch

1. Let’s create a new folder: firstname-lastname

2. Create a README.md

3. Add some text to it.

continued...

Stage the changes:

```
git add .
git commit -m “Add readme”
```

now you can ‘push’ your changes

```
git push -u origin derek-zoolander
```

note: `-u` only needs to be set the first time

Activity: Git Add, Commit, Push

Using GitHub and the command line, do the following:

- Create a new public GitHub repository and name it whatever you like. Be sure to check the box for “Initialize this repository with a README.”
- Next, clone the repo to your local directory.
- Then create an HTML file inside the local directory.
- Add, commit, and push the code to GitHub.

Suggested Time:

15 minutes

Activity: Git Add, Commit, Push

Bonus:

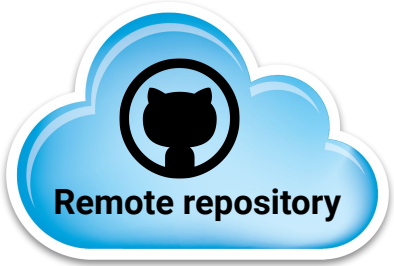
- Create a new public GitHub repository and name it `zen-garden`. Be sure to check the box for “Initialize this repository with a README.”
- Clone the repo to your local directory.
- Go to CSS Zen Garden. Navigate to a few of the examples and choose a page that you like.
- Download the HTML and CSS. Each page has a link to download the code, normally near the top of the page.
- Move the HTML and CSS into your newly cloned repo and open the HTML in Chrome.
- Use inspect element to identify a page element you would like to change in some way.
- Change the CSS in any way you'd like.
- Add, commit, and push the code to GitHub.

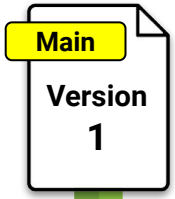
Basic Git Commands

`git clone`	Copies an entire repo (to begin)
`git add`	Adds a file for inclusion in Git
`git commit`	Notes a change to the local repo
`git push`	Sends changes to hosting service
`git pull`	Downloads freshest version of repo

Git Workflow

Git Clone





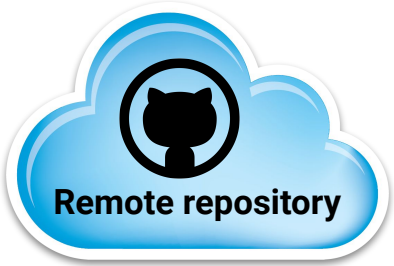
Staging area is where you edit the files that will be part of the next commit.

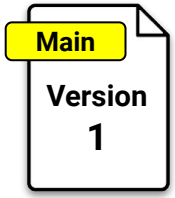
Staging area

`clone`

Takes an existing GitHub repository, downloads it to the local computer, and links it to GitHub.

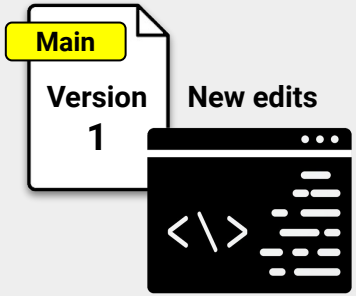
Git Add





Staging area is where you edit the files that will be part of the next commit.

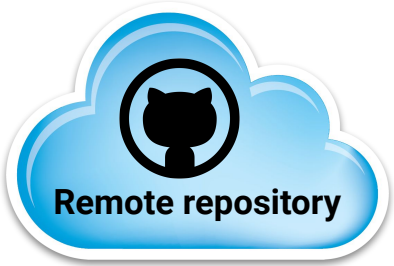
Staging area

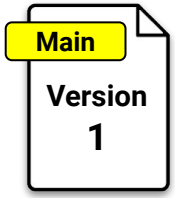


`git add`

Adds your edits to be committed later.

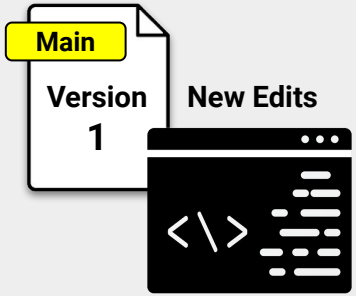
Git Commit





Staging area is the where you edit the files that will be part of the next commit.

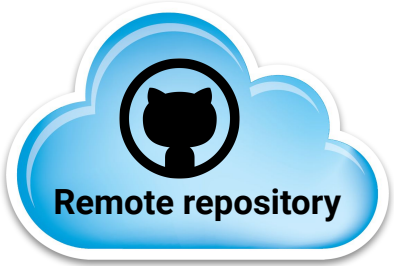
Staging area

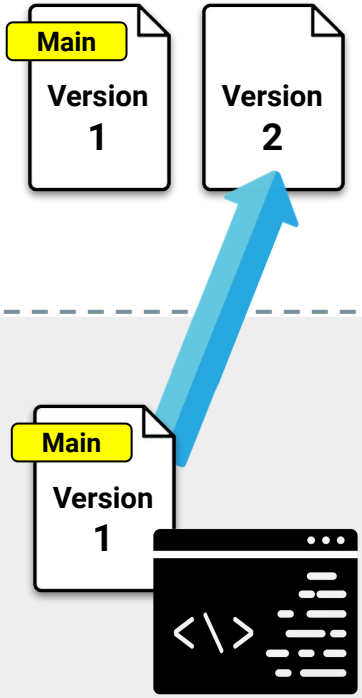


`git commit`

Your staged changes are saved once you commit.

Git Push





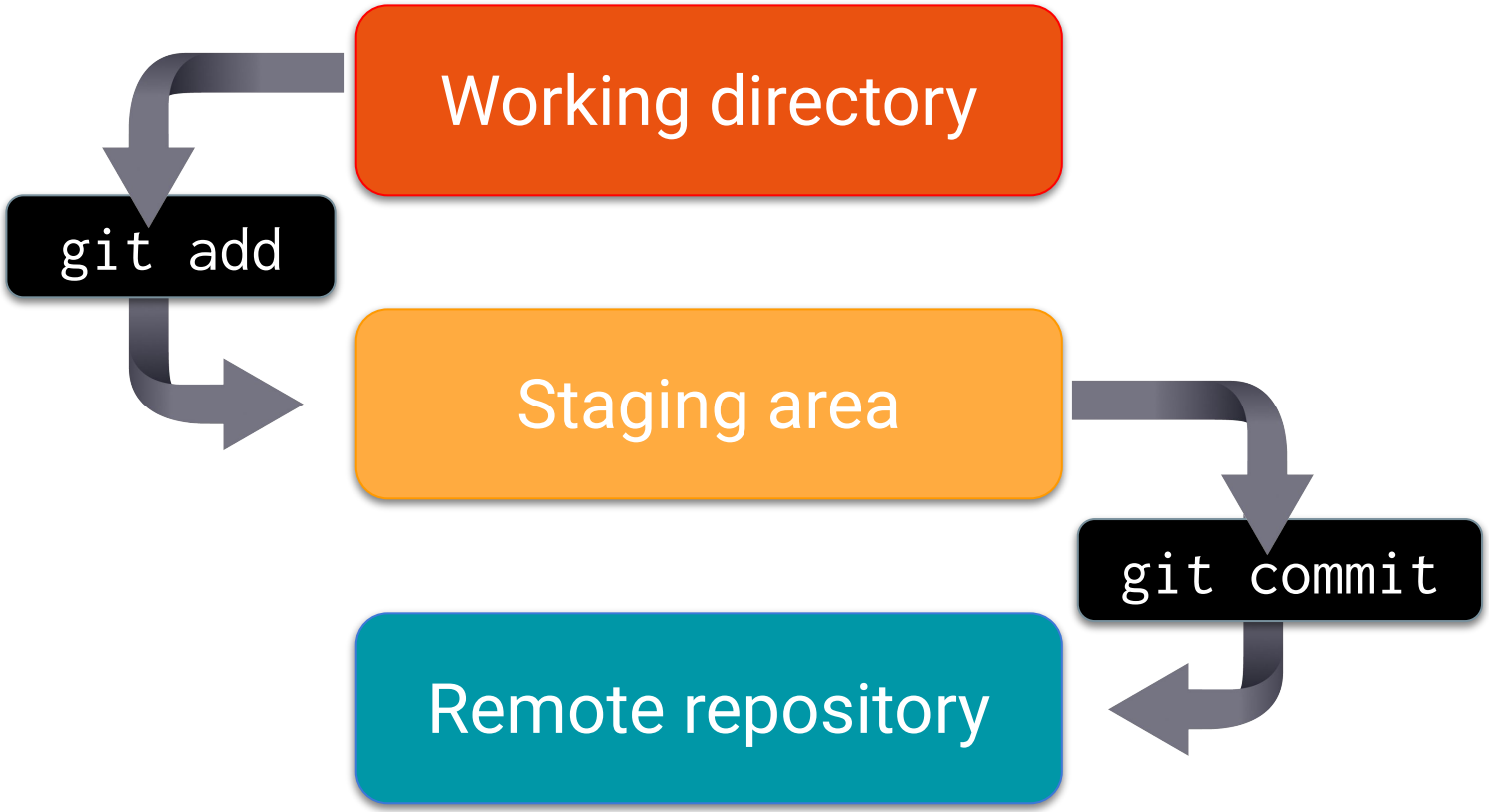
Staging area is where you edit the files that will be part of the next commit.

Staging area

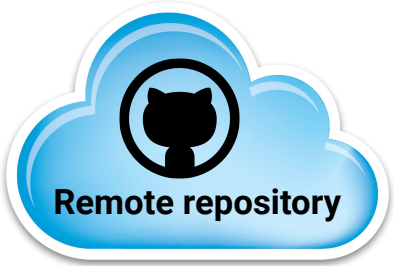
`git push`

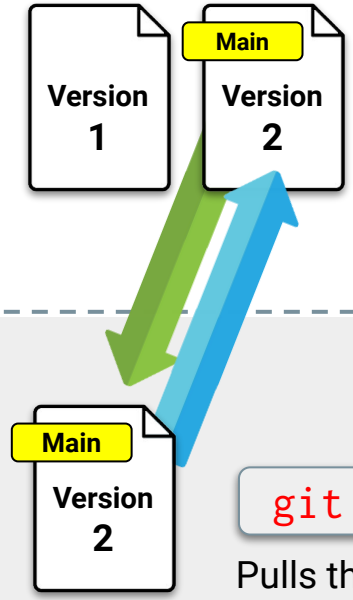
Pushes your changes to the remote repository.

Basic Git Commands



Git Pull





Staging area is where you edit the files that will be part of the next commit.

Staging area

`git pull`

Pulls the latest version of the remote repository and updates the local repository to match.

Working directory (VS code)

Staging area

Local repository

Remote repository

git add

git commit

git push

git pull

git checkout

Basic Git Commands

`git clone`	Copies an entire repo (to begin)
`git add`	Adds a file for inclusion in Git
`git commit`	Notes a change to the local repo
`git push`	Sends changes to hosting service
`git pull`	Downloads freshest version of repo

GitHub Guide

Get Started with GitHub

What is Git?

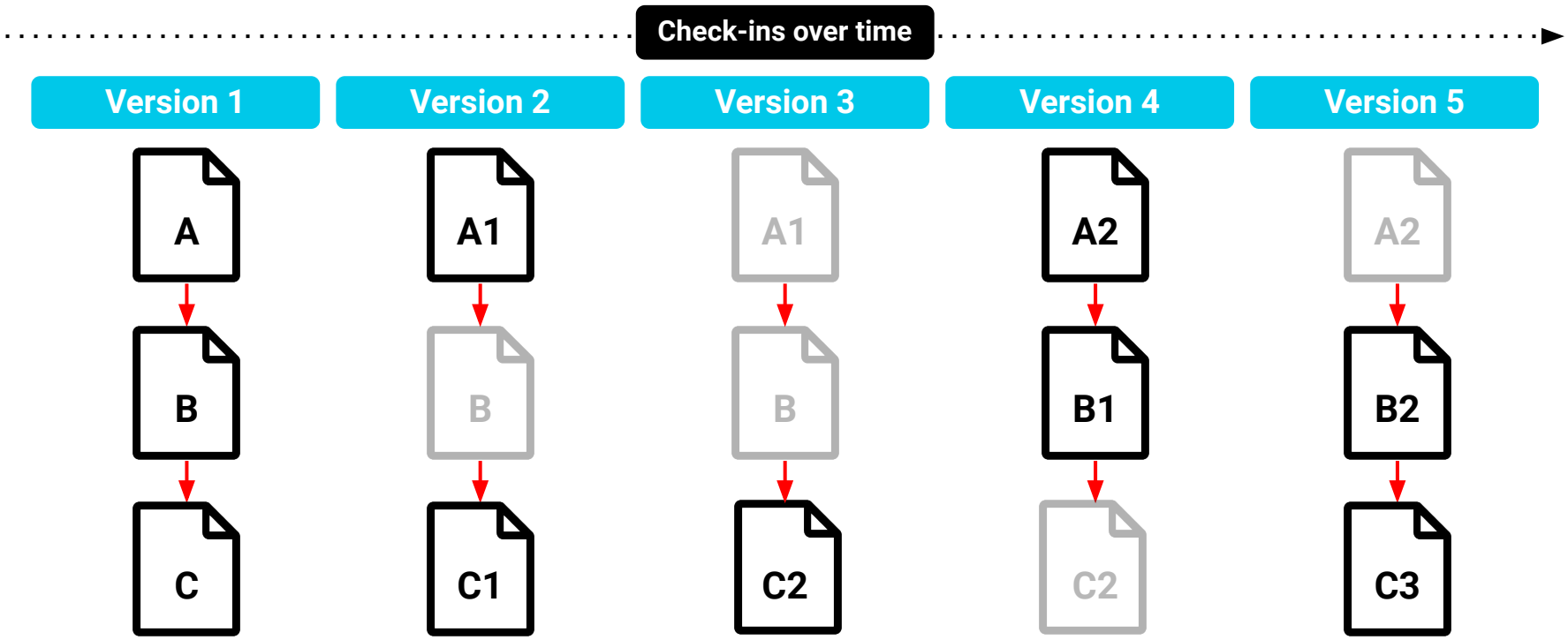
Git is a free and open source distributed version control system for managing source code.

It’s designed to handle everything, from small to very large projects, with speed and efficiency.

What is version control?

Introduction to Git

Version control is a system that allows you to manage changes over time.



Why is version control important?

Introduction to Git

Version control is important because:

01 Modern web development is highly collaborative.

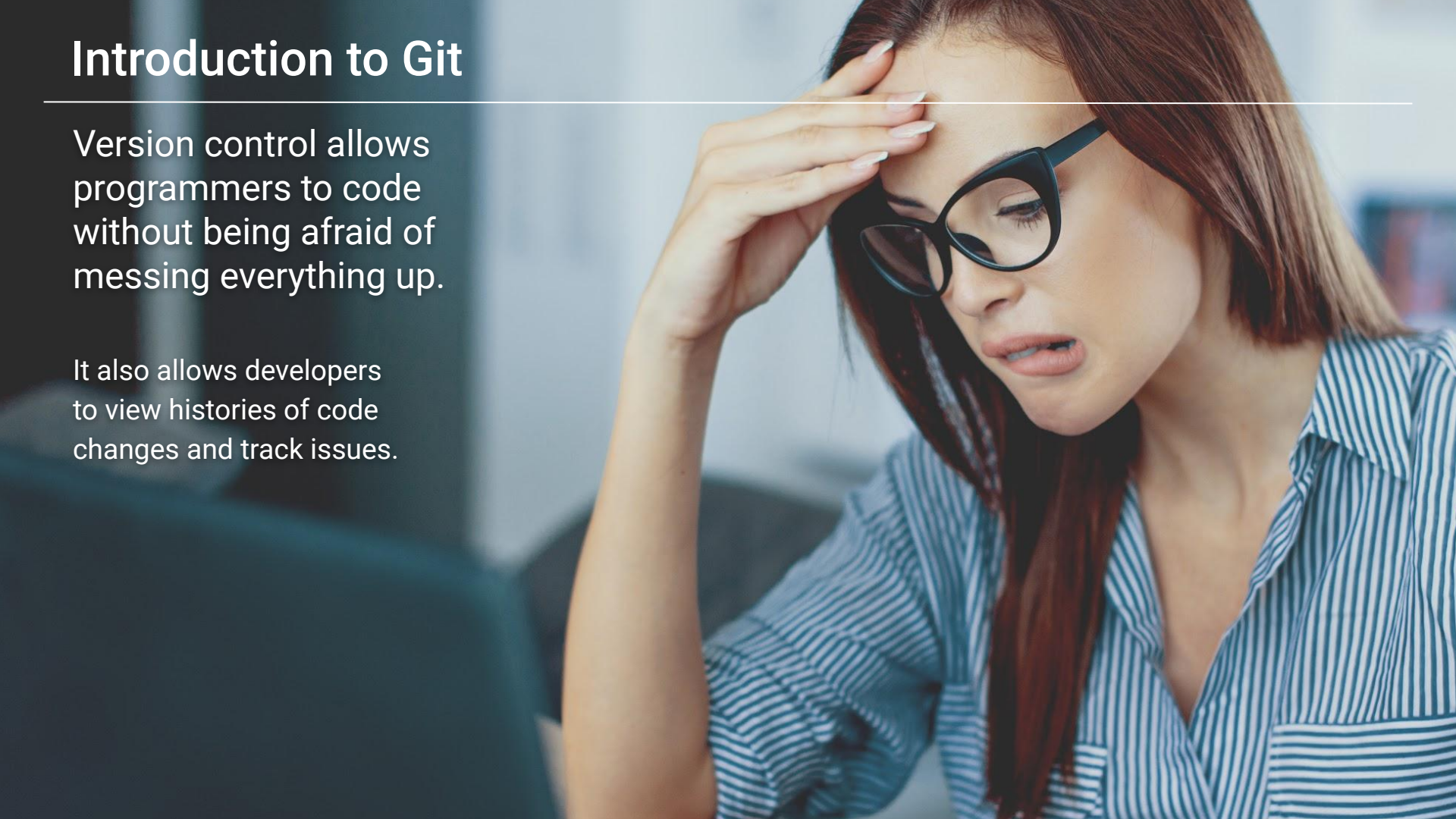
02 Teams are often extremely large and spread out across the country or world.

03 Apps are sometimes made up of hundreds or even thousands of files.

Introduction to Git

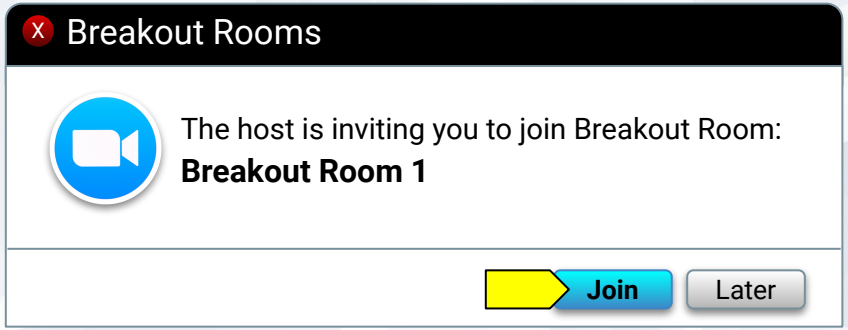
Version control allows programmers to code without being afraid of messing everything up.

It also allows developers to view histories of code changes and track issues.



Quick Activity: Explain Git

Take a few minutes to explain to one another what Git version control is for.



Jira GitHub IntelliJ Idea Spring Java Heroku Apache Cassandra Jenkins JUnit Swagger

Suggested Time:

3 minutes

Time's Up! Let's Review.

Why use Git?

Why Use Git?

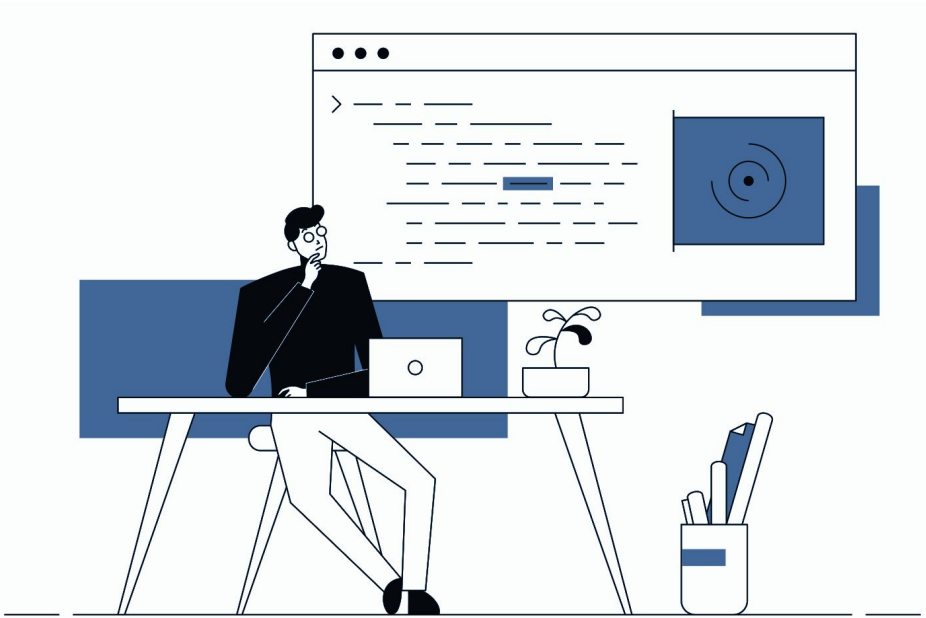
A version control system like Git makes it easy to:

01 Keep track of code history.

02 Collaborate on code as a team.

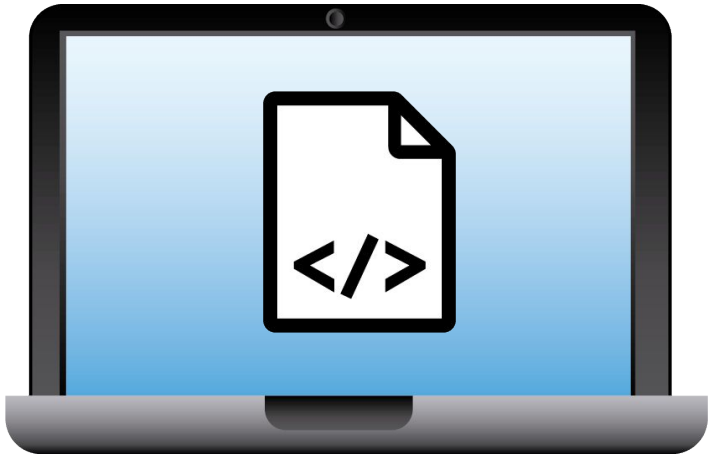
03 See who made which changes.

04 Deploy to staging or production.



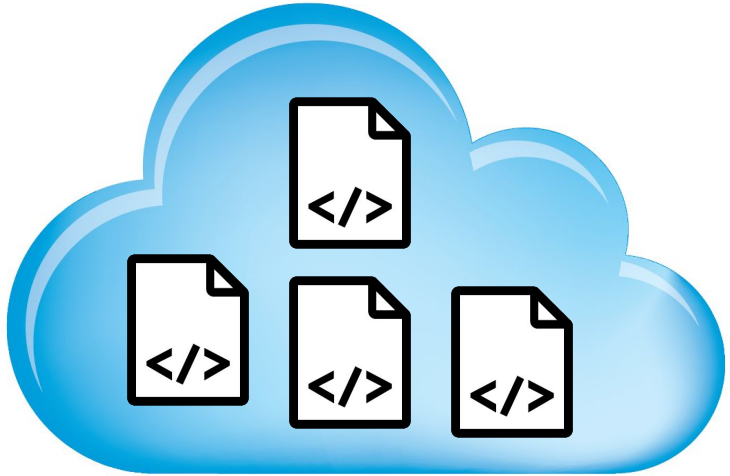
What is the difference between Git and GitHub?

Git vs. GitHub



git

is a **version control system**
that lets you manage and keep track
of your source code history.



GitHub

is a **cloud-based hosting service**
that lets you manage Git repositories.

What is a repository?

A **repository** is a directory or storage space where your projects can live.

Sometimes GitHub users shorten the word to “repo.”

GitHub

Instructor Demonstration

GitHub Examples

Questions?

Instructor Demonstration

Get Started with Git

Activity: Git Add, Commit, Push

Using GitHub and the command line, do the following:

- Create a new public GitHub repository and name it whatever you like. Be sure to check the box for “Initialize this repository with a README.”
- Next, clone the repo to your local directory.
- Then create an HTML file inside the local directory.
- Add, commit, and push the code to GitHub.

Suggested Time:

15 minutes

Basic Git Commands

`git clone`	Copies an entire repo (to begin)
`git add`	Adds a file for inclusion in Git
`git commit`	Notes a change to the local repo
`git push`	Sends changes to hosting service
`git pull`	Downloads freshest version of repo

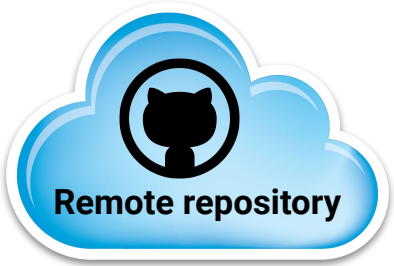
Activity: Git Add, Commit, Push

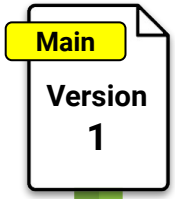
Bonus:

- Create a new public GitHub repository and name it `zen-garden`. Be sure to check the box for “Initialize this repository with a README.”
- Clone the repo to your local directory.
- Go to CSS Zen Garden. Navigate to a few of the examples and choose a page that you like.
- Download the HTML and CSS. Each page has a link to download the code, normally near the top of the page.
- Move the HTML and CSS into your newly cloned repo and open the HTML in Chrome.
- Use inspect element to identify a page element you would like to change in some way.
- Change the CSS in any way you'd like.
- Add, commit, and push the code to GitHub.

Git Workflow

Git Clone





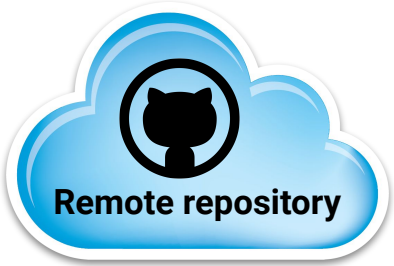
Staging area is where you edit the files that will be part of the next commit.

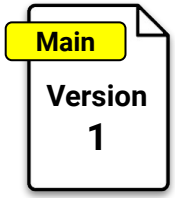
Staging area

`clone`

Takes an existing GitHub repository, downloads it to the local computer, and links it to GitHub.

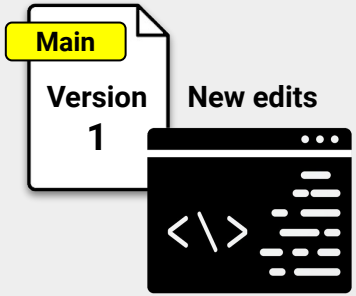
Git Add





Staging area is where you edit the files that will be part of the next commit.

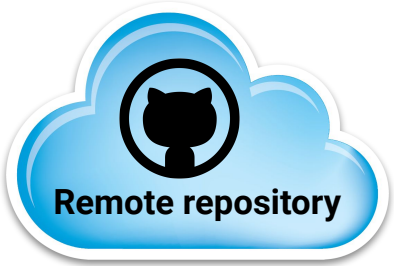
Staging area

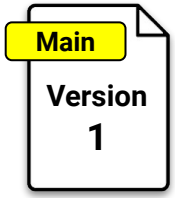


`git add`

Adds your edits to be committed later.

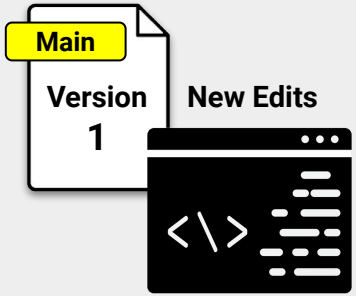
Git Commit





Staging area is the where you edit the files that will be part of the next commit.

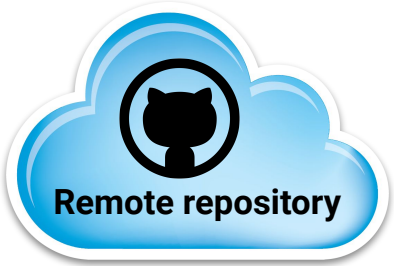
Staging area

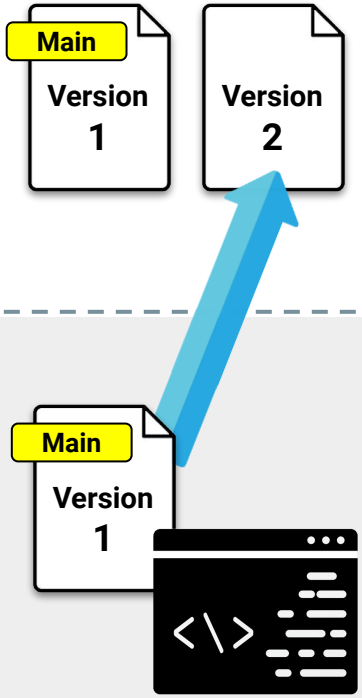


`git commit`

Your staged changes are saved once you commit.

Git Push





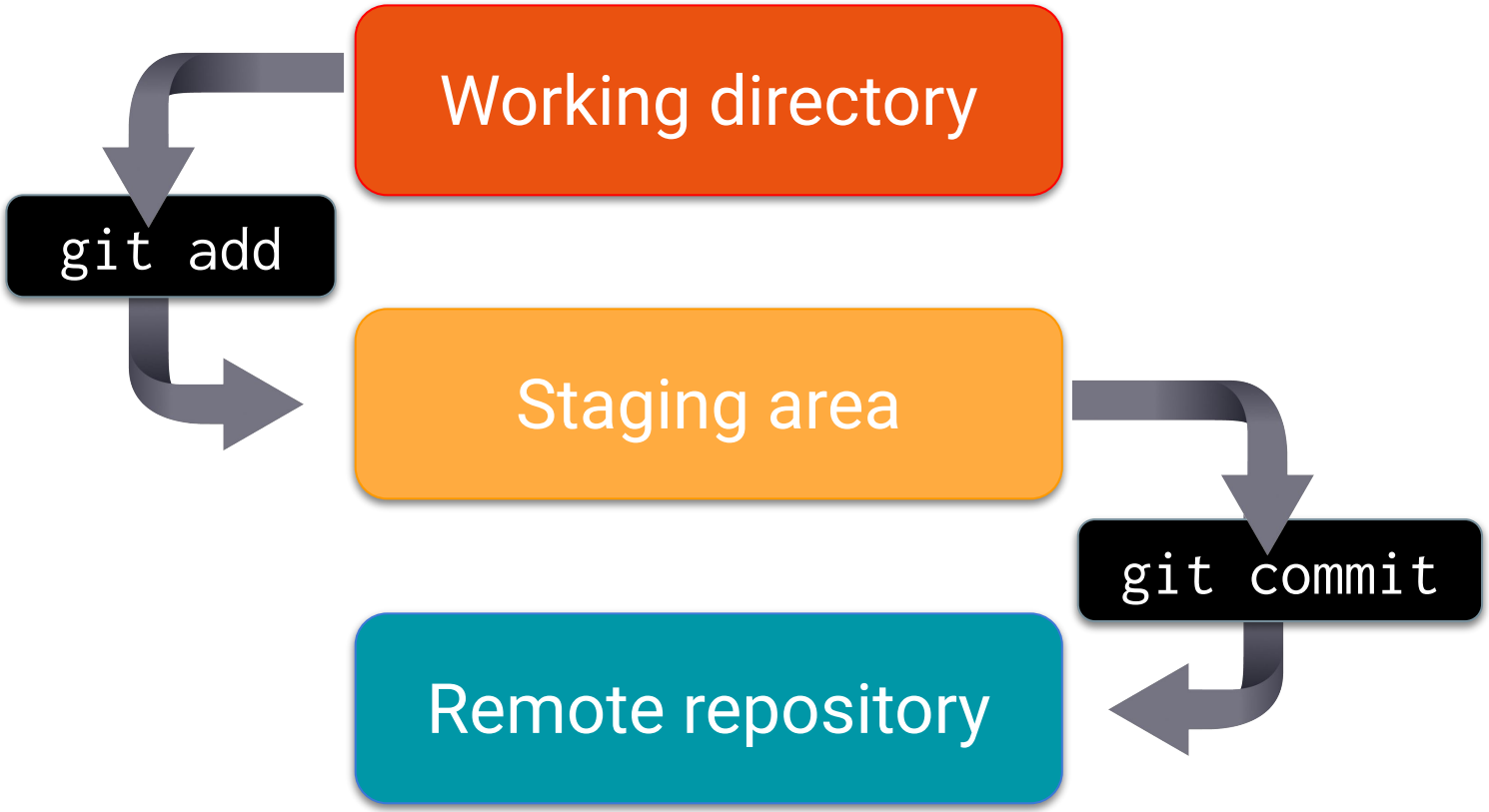
Staging area is where you edit the files that will be part of the next commit.

Staging area

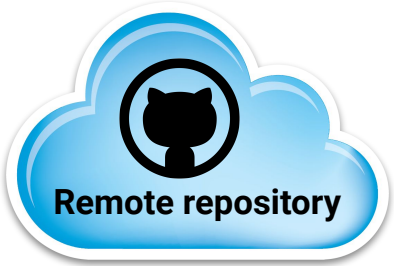
`git push`

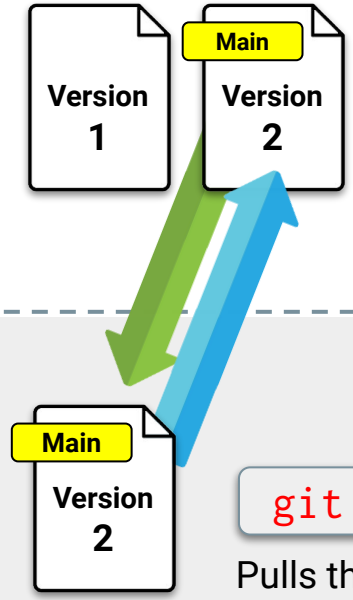
Pushes your changes to the remote repository.

Basic Git Commands



Git Pull





Staging area is where you edit the files that will be part of the next commit.

Staging area

`git pull`

Pulls the latest version of the remote repository and updates the local repository to match.

Working directory (VS code)

Staging area

Local repository

Remote repository

git add

git commit

git push

git pull

git checkout

Basic Git Commands

`git clone`	Copies an entire repo (to begin)
`git add`	Adds a file for inclusion in Git
`git commit`	Notes a change to the local repo
`git push`	Sends changes to hosting service
`git pull`	Downloads freshest version of repo

GitHub Guide

Get Started with GitHub

Questions?

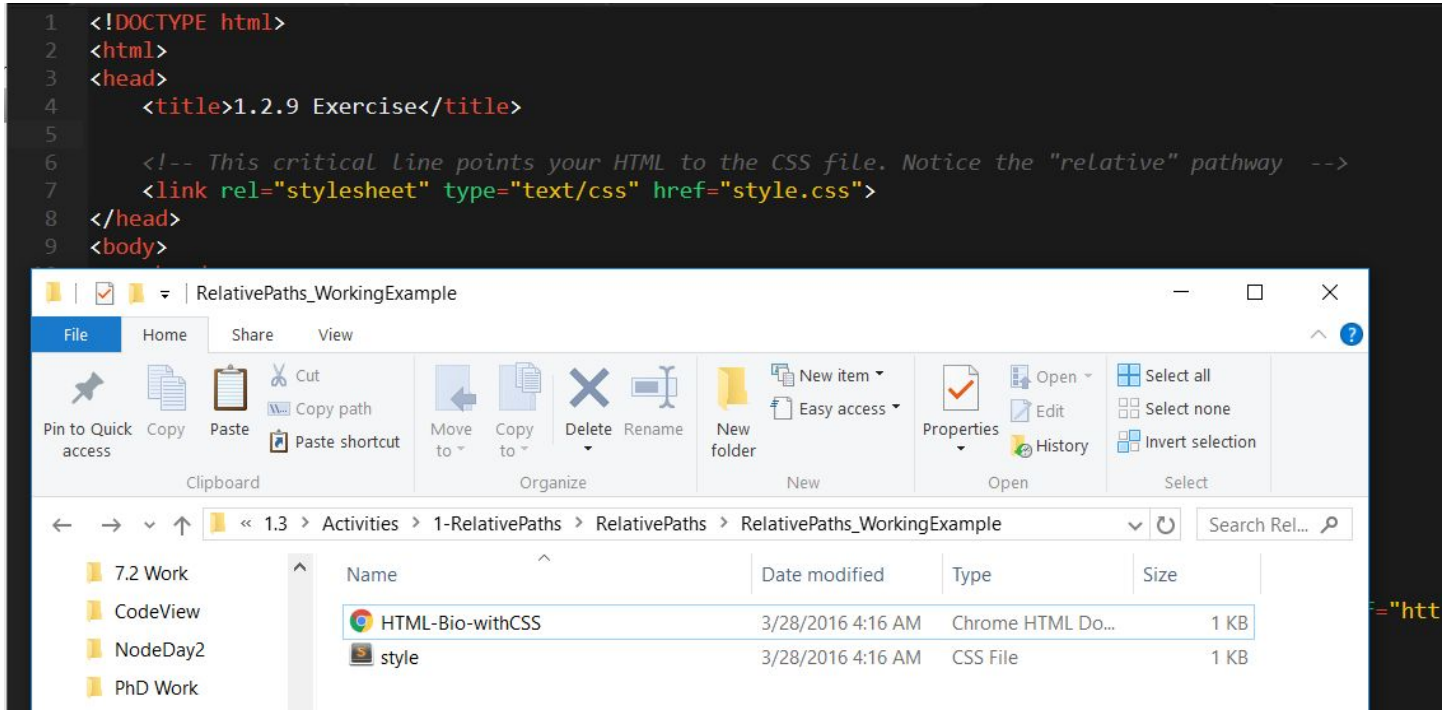
Instructor Demonstration

Chrome Inspector

Relative File Paths

Relative File Paths

Relative file paths connect us with other files in our working directory.
In this case, style.css is in the same folder as our HTML document.



Instructor Demonstration

Relative File Paths

Absolutely No Absolute Paths

Always use relative file paths!

- If you deploy websites without relative file paths, **all of your links will fail**.
- The same will happen if you move your project from one folder to another.
- Remember, there is no such thing as a C: drive on the internet.

VERY, VERY BAD

```
<!-- BAD!!!! -->
<link rel="stylesheet" href="D:/trilogy/FullStack-Lesson-Plans/02-lesson-plans/01-
html-css-three-days/1-Class-Content/1.3/Activities/1-RelativePaths/RelativePaths/
RelativePaths_WorkingExample/style.css">
```

Activity: Relative File Paths Activity

Instructions:

- Unzip the folder provided to you (or copy and paste the contents outside).
- Then modify each of the four `html-bio.html` pages such that they can access the CSS inside their folder. **Don't move the CSS file, and don't move the HTML file.**
- Use relative linking to make it work!
- **Hint:** If you need some reading material on relative linking, you can use https://css-tricks.com/quick-reminder-about-file-paths/.

Suggested Time:

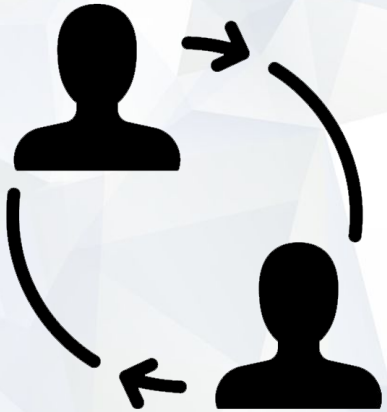
15 minutes

Time's Up! Let's Review.

Display

Instructor Demonstration

Display



Partner Activity: Display Activity

In this activity, you'll work with a partner to resolve issues within the given code

Suggested Time:

10 minutes

Time's Up! Let's Review.


Break

Box Model

CSS box model: Every HTML element you create is represented as a rectangular box, with the box's content, padding, border, and margin built up around one another like the layers of an onion!

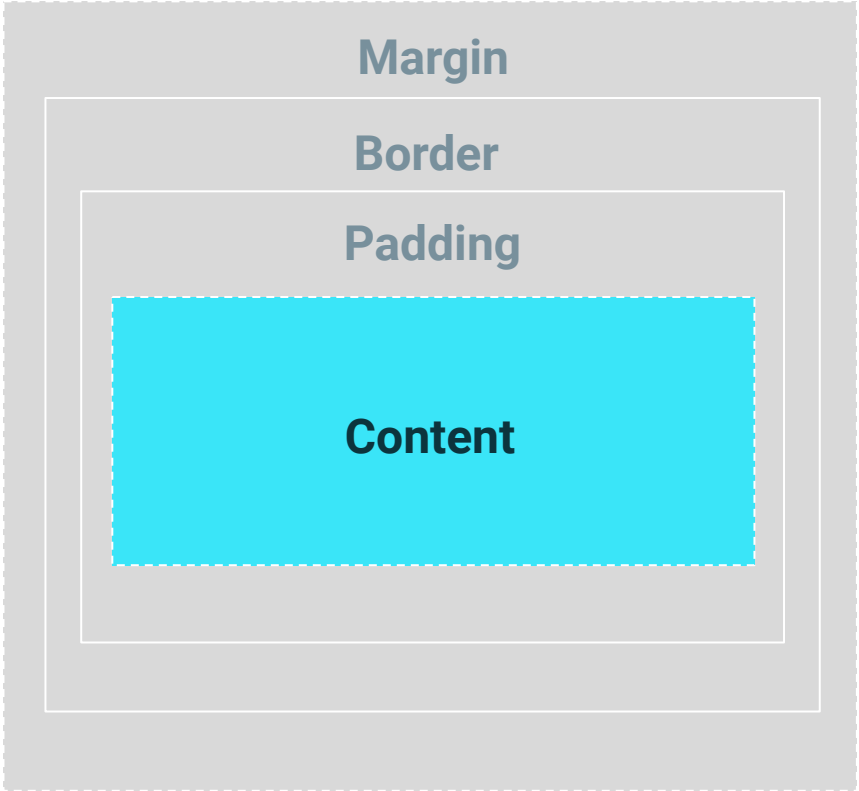
The CSS Box Model

Content: This is what's inside a container, such as text and images.

Padding: Padding is used to expand the space *inside* a box, between our content and our border.

Border: A border surrounds your content and distinguishes it from other elements around it.

Margin: Margin is used to create additional space *outside* of your border. Margin spacing increases the distance between your box and other elements on the page.



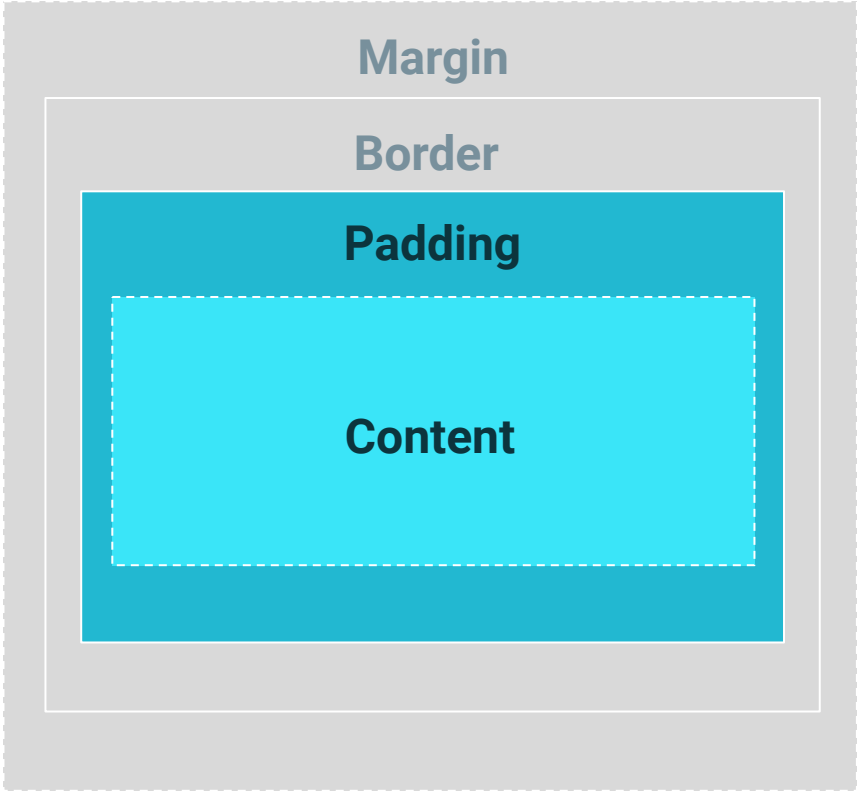
The CSS Box Model

Content: This is what's inside a container, such as text and images.

Padding: Padding is used to expand the space *inside* a box, between our content and our border.

Border: A border surrounds your content and distinguishes it from other elements around it.

Margin: Margin is used to create additional space *outside* of your border. Margin spacing increases the distance between your box and other elements on the page.



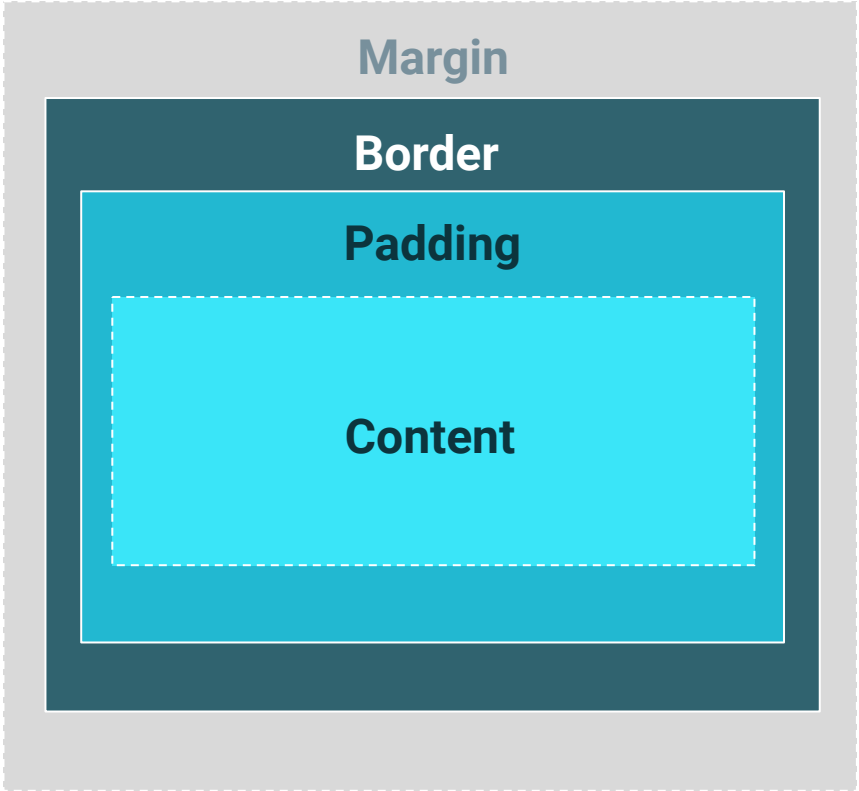
The CSS Box Model

Content: This is what's inside a container, such as text and images.

Padding: Padding is used to expand the space *inside* a box, between our content and our border.

Border: A border surrounds your content and distinguishes it from other elements around it.

Margin: Margin is used to create additional space *outside* of your border. Margin spacing increases the distance between your box and other elements on the page.



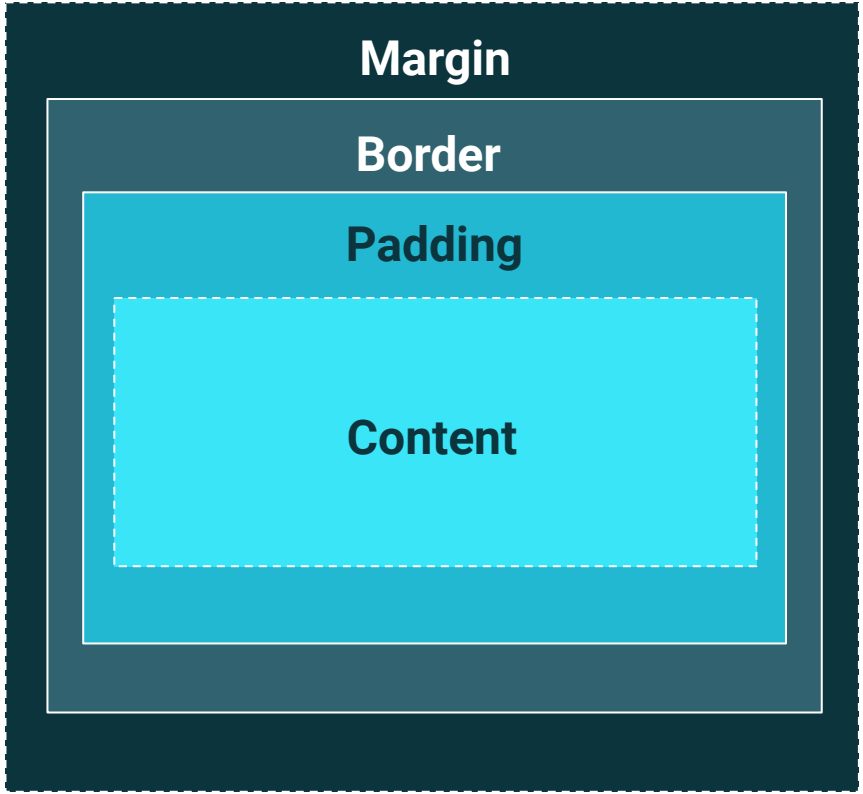
The CSS Box Model

Content: This is what's inside a container, such as text and images.

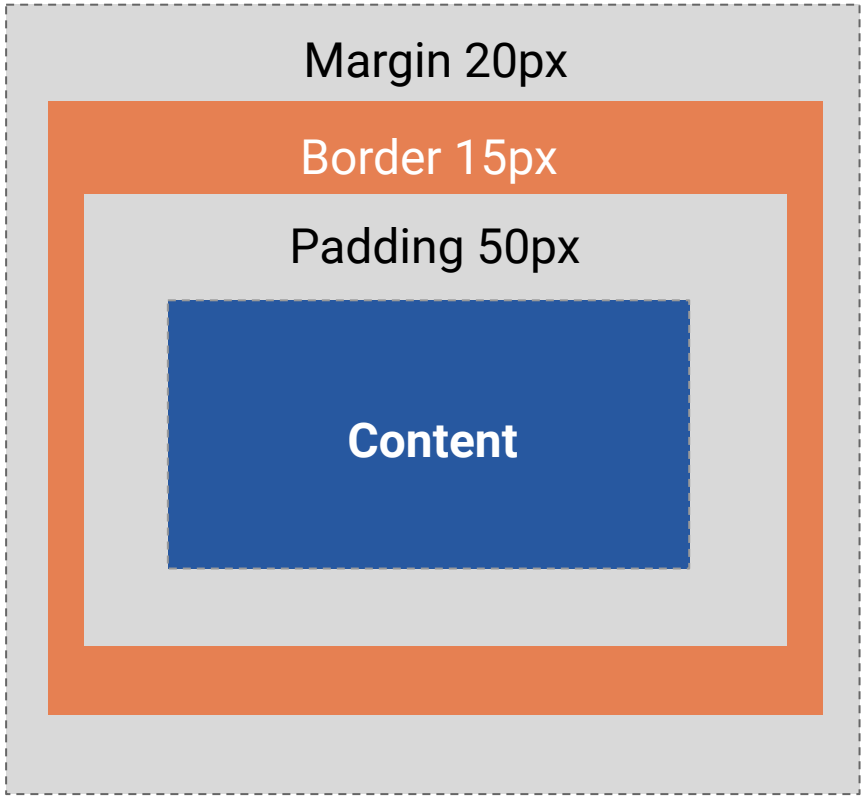
Padding: Padding is used to expand the space *inside* a box, between our content and our border.

Border: A border surrounds your content and distinguishes it from other elements around it.

Margin: Margin is used to create additional space *outside* of your border. Margin spacing increases the distance between your box and other elements on the page.

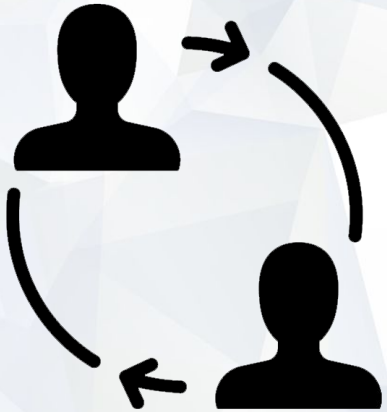


The CSS Box Model




```
div {
    background-color: navy blue;
    padding: 50px;
    border: 15px orange;
    margin: 20px;
}
```

The box model applies to all elements, whether they are text, images, div sections, etc.



Partner Activity: Box Model Activity

Work with a partner to implement the following user story:

As a developer, I want to use the CSS box model properties to position four boxes inside a frame

Suggested Time:

10 minutes

Time's Up! Let's Review.

Positioning

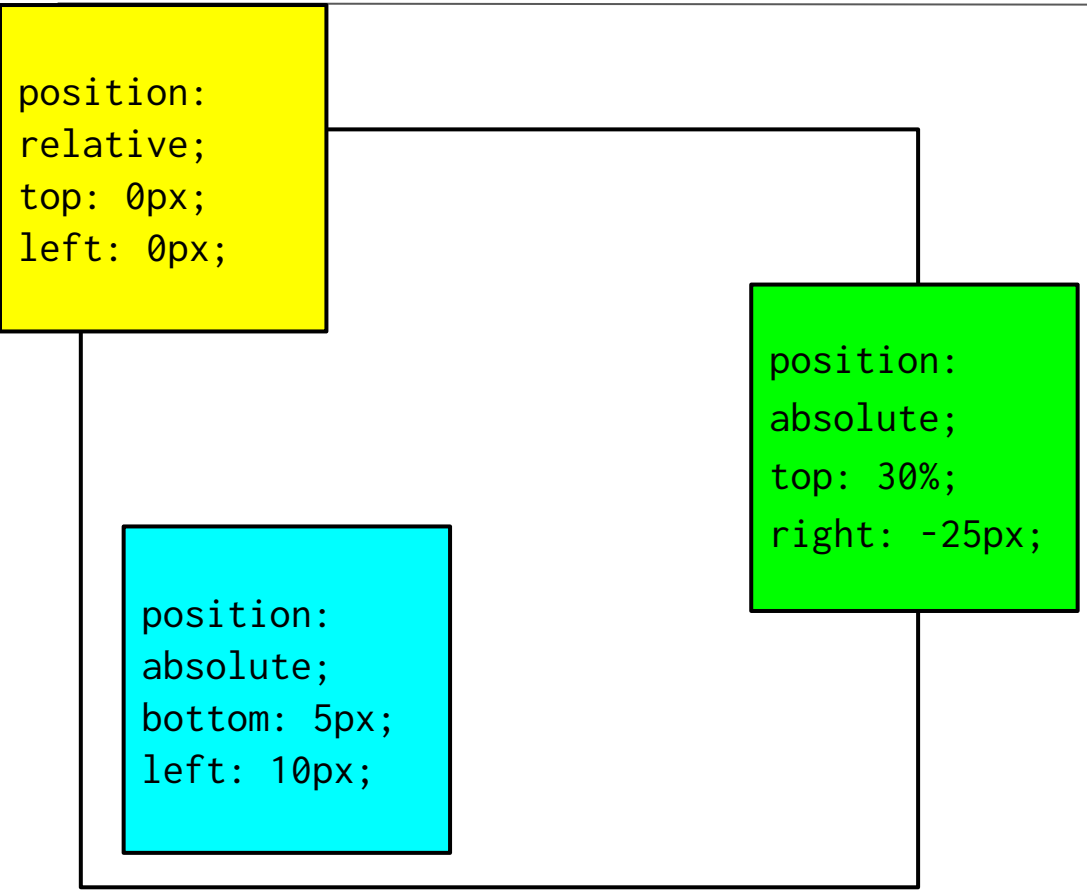
Position: Static

Positions an element according to the normal flow of the document.

```
.myClass {
position: static;
}
```

Static is the default position property applied.

position: static;



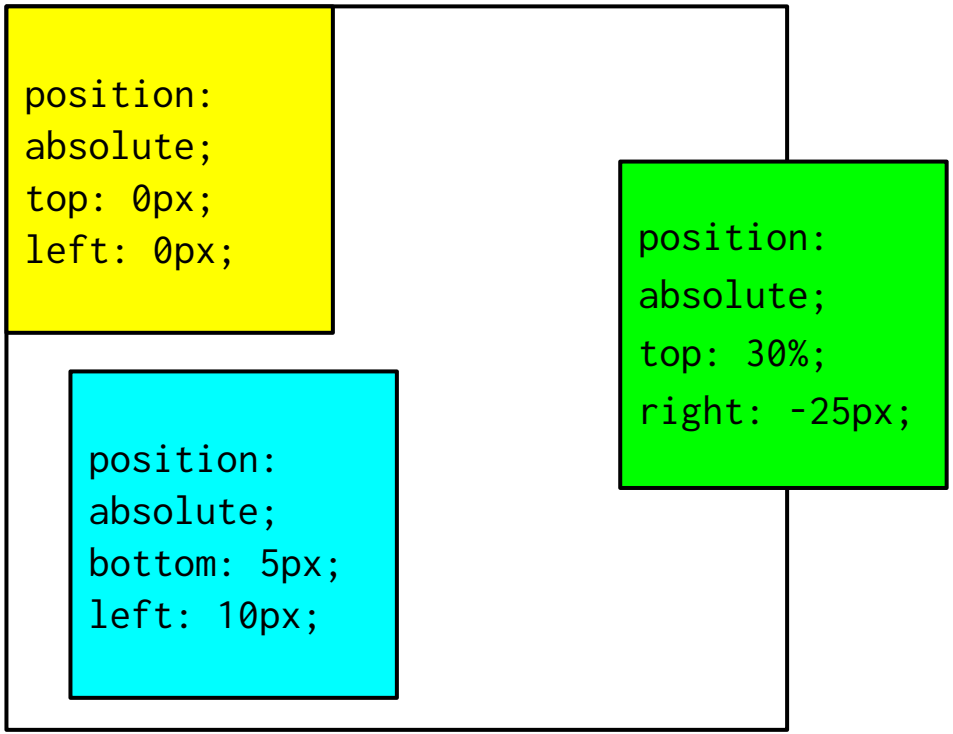

```
position: relative;
top: 0px; left: 0px;

position: absolute;
bottom: 5px; left: 10px;

position: absolute;
top: 30%; right: -25px;
```

Position: Relative

Positions the element relative to the normal position it would otherwise have (e.g., if it were left as a static element). Used to offset an element from its default, based on any values assigned (top, right, bottom, left).

```
.myClass {
position: relative;
}
```

position: relative;

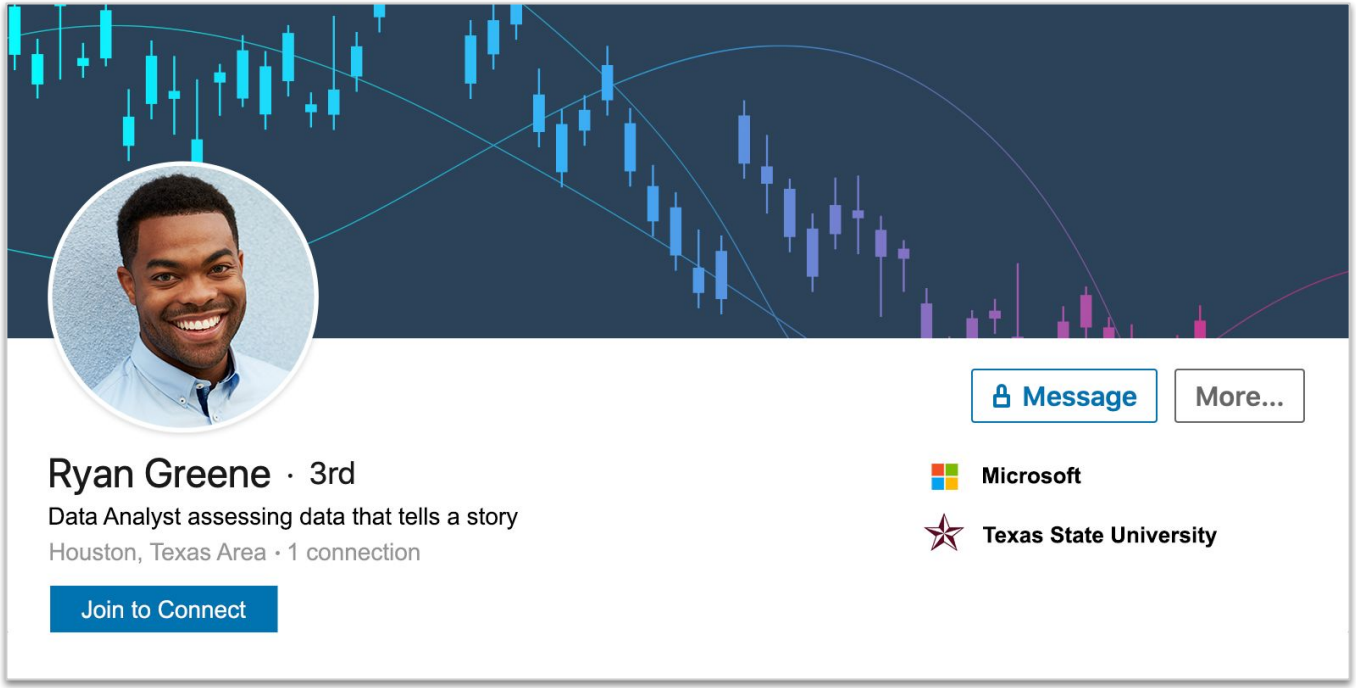



```
position: absolute;
top: 0px; left: 0px;

position: absolute;
bottom: 5px; left: 10px;

position: absolute;
top: 30%; right: -25px;
```

Switching the boxes to relative will nudge the boxes in relation to their “original” location.

Position: Relative

An example of a relative element is the container for any profile picture that overlaps other HTML elements. Think of the container for the profile picture for LinkedIn.



Position: Absolute

Absolute elements are removed from the document flow and are placed in absolute position based either on their parent container or the overall document body.

```
.myClass {
position: absolute;
}
```

Absolute Positioning vs. Relative Positioning

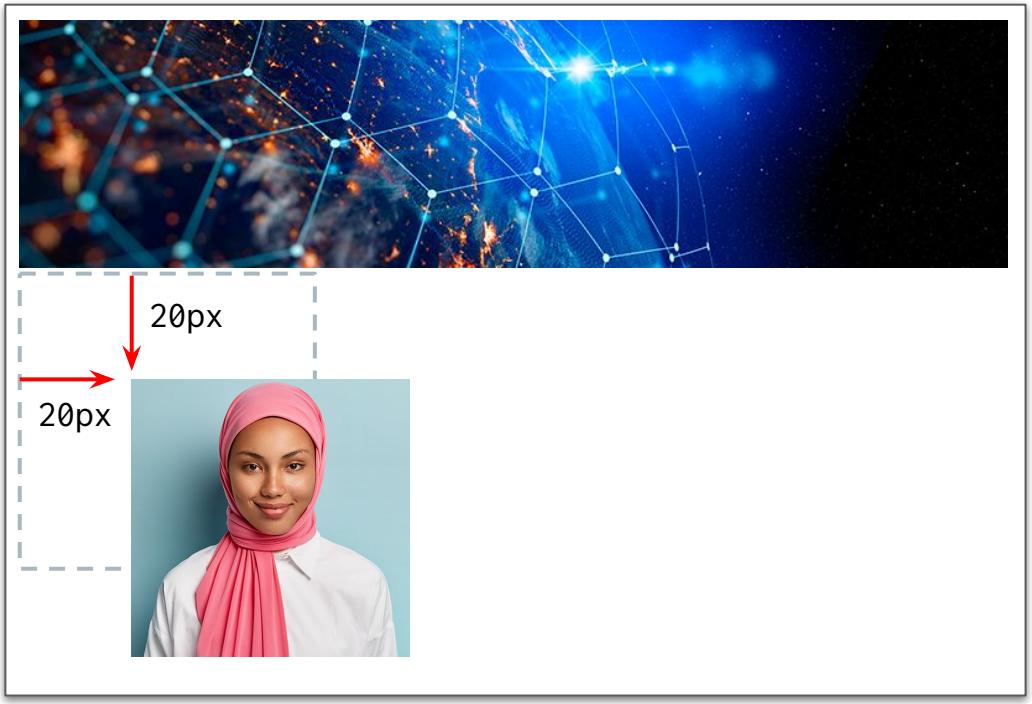
Relative

Relative positioning is just like stating no positioning at all! The left, right, top, and bottom attributes “nudge” elements out of their normal layout.

Absolute

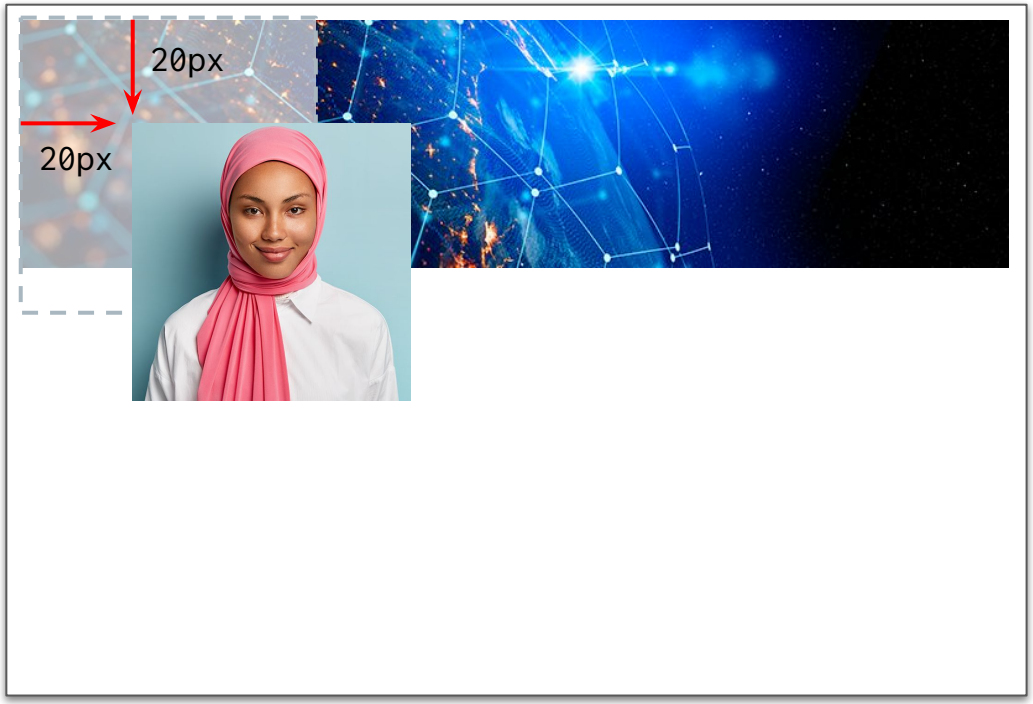
Absolute positioning allows you to place your element *precisely* where you want it—and it won’t budge.

position: relative;




```
position: relative;
top: 20px;
left: 20px;
```

position: absolute;




```
position: absolute;
top: 20px;
left: 20px;
```

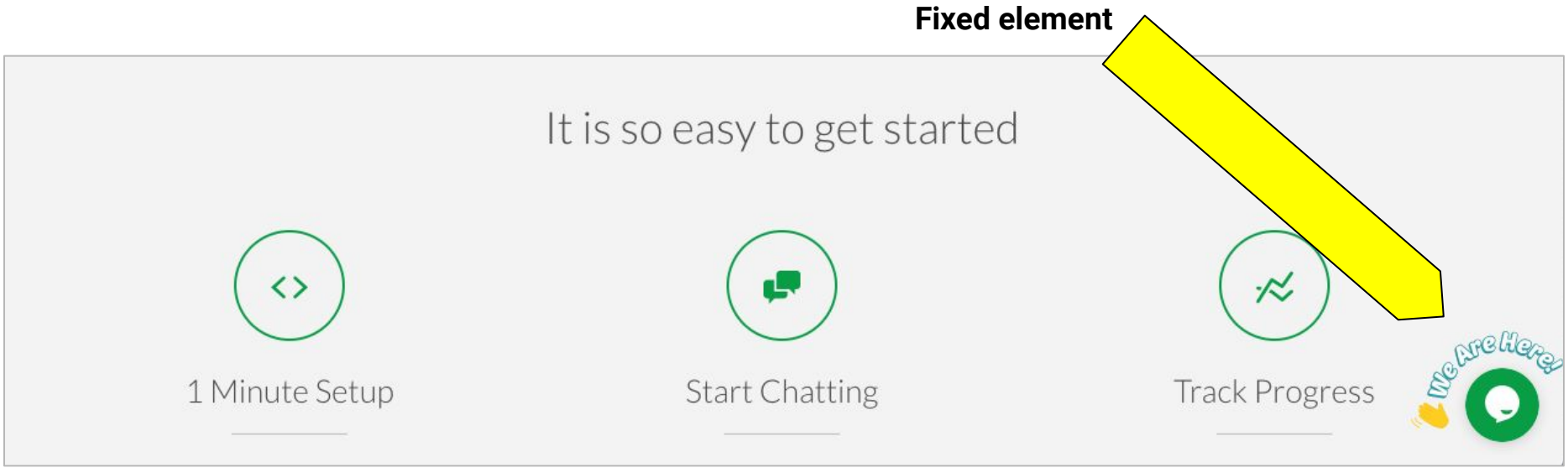
Position: Fixed

The element is removed from the normal document flow. It will always stay where you put it, even if a user scrolls down the page.

```
.myClass {
position: fixed;
}
```

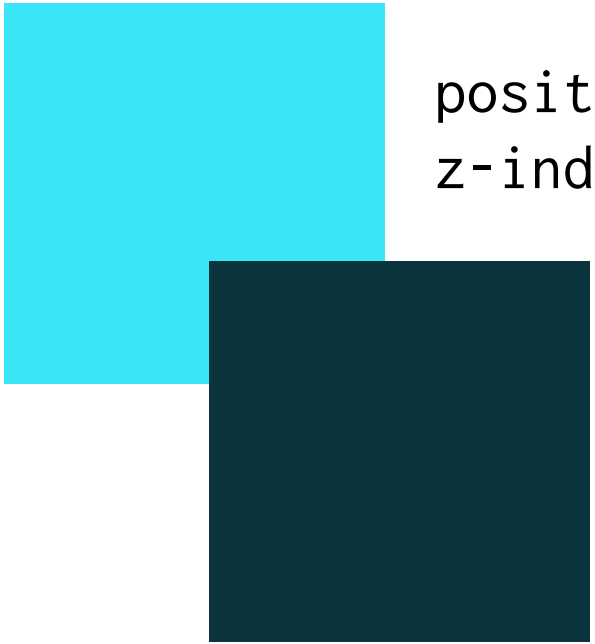
Position: Fixed

An example of a fixed element might be a “Contact us” button that follows you as you scroll.



Layering with z-Index

The z-index property allows you to layer elements on top of each other.



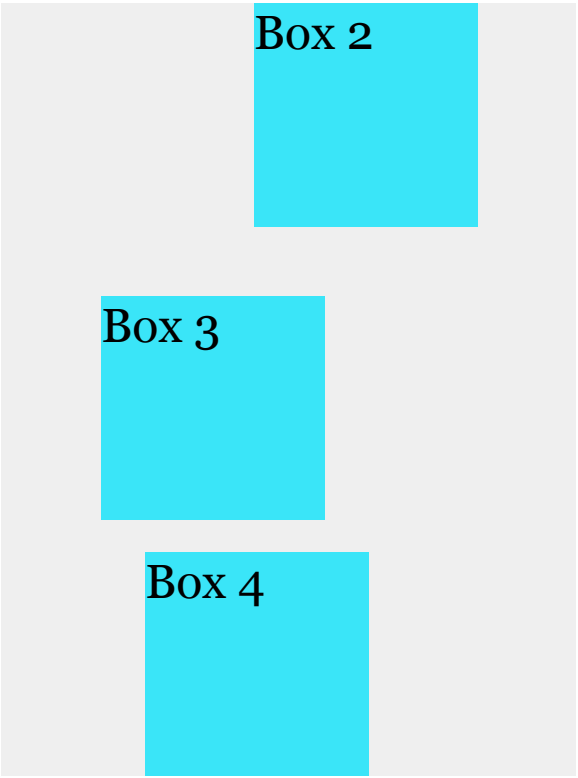
```
position: absolute;
z-index:1;
```

```
position: absolute;
z-index:2;
```

Hiding Things

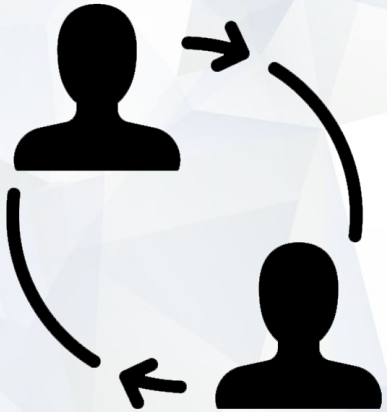
Display: none allows you to hide elements from view.

This will become useful in later sections, when we'll hide and reveal specific HTML elements of our choosing.



Instructor Demonstration

Positioning



Partner Activity: Positioning Activity

Work with a partner to implement the following user story:

As a developer, I want to use the CSS position property to change the layout of my page.

Suggested Time:

10 minutes

Time's Up! Let's Review.

Mini-Project: Landing Page

Instructor Demonstration

Mini-Project: Landing Page

Mini-Project: Landing Page

Instructions:

- Unzip the folder provided to you (or copy and paste the contents outside).
- Follow the provided instructions to complete this project

Suggested Time:

30 minutes

Time's Up! Let's Review.

Questions?


RECAP


The End