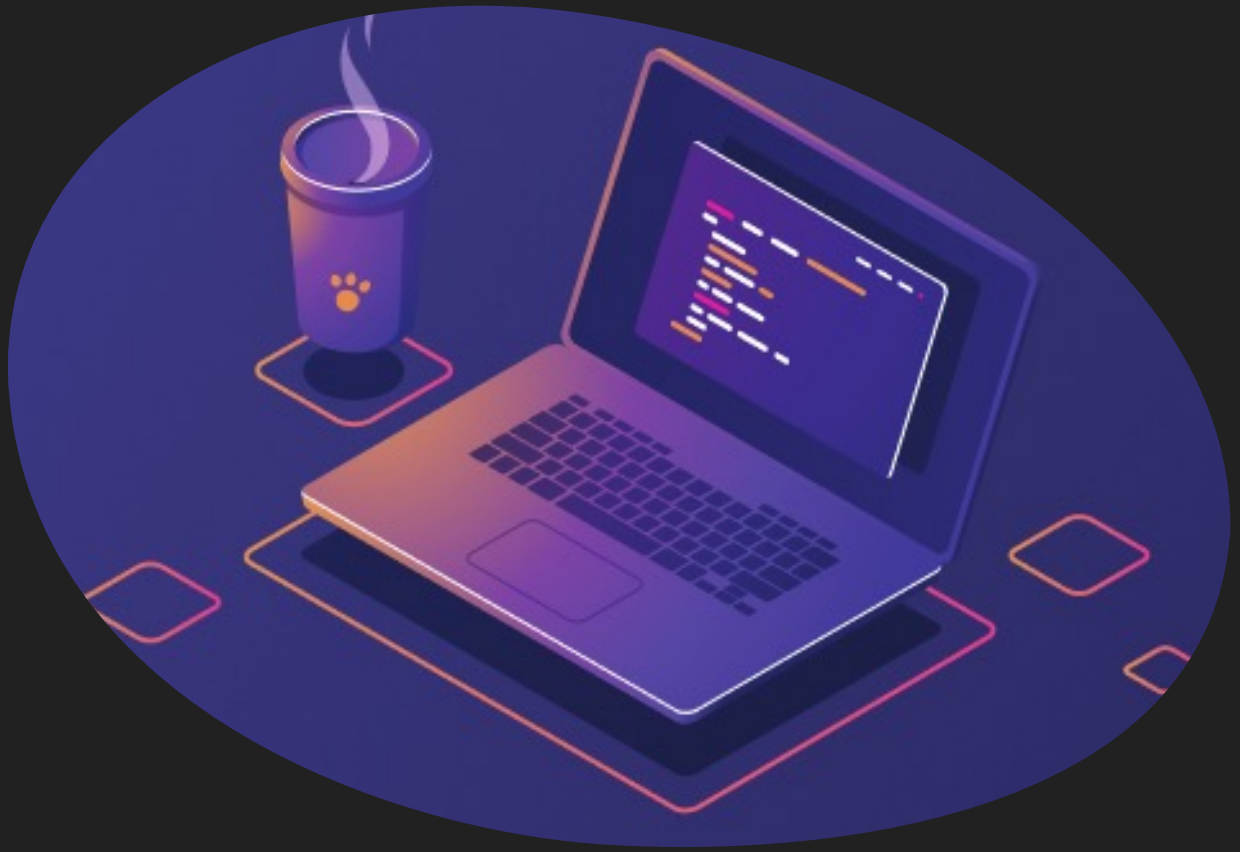


Week 2 : SOFTWARE DEVELOPMENT TOOLS AND ENVIRONMENTS

Week2 Getting Started with Git

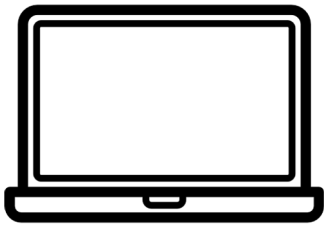
- **Week2 Commands:**
  - Changing code in a Repository
    - **git add**
  - Committing these changes
    - **git commit**
  - Pushing or Pulling Changes
    - **git push** and **git pull**
  - Checking Status, Logs, and Changes
    - ***git status, git log, git diff***
  - *G*etting repository *changes from a remote branch*
    - **git fetch**, **git pull**

Week 2
Basic Git Usage

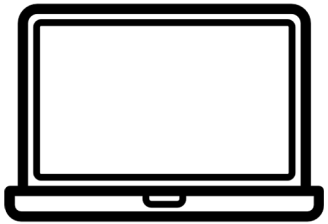
Week 2 Getting Started with Git

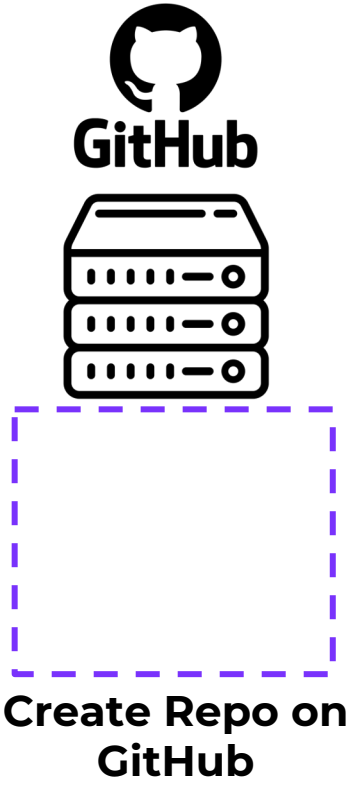
- **Basic Git Usage**
  - Let’s cover the basic cycle of a workflow of using Git and GitHub.
  - This particular basic example will assume just a solo developer and everything working on the same branch.
  - We’ll cover branches and working with others on Week 3.

Week 2 Getting Started with Git

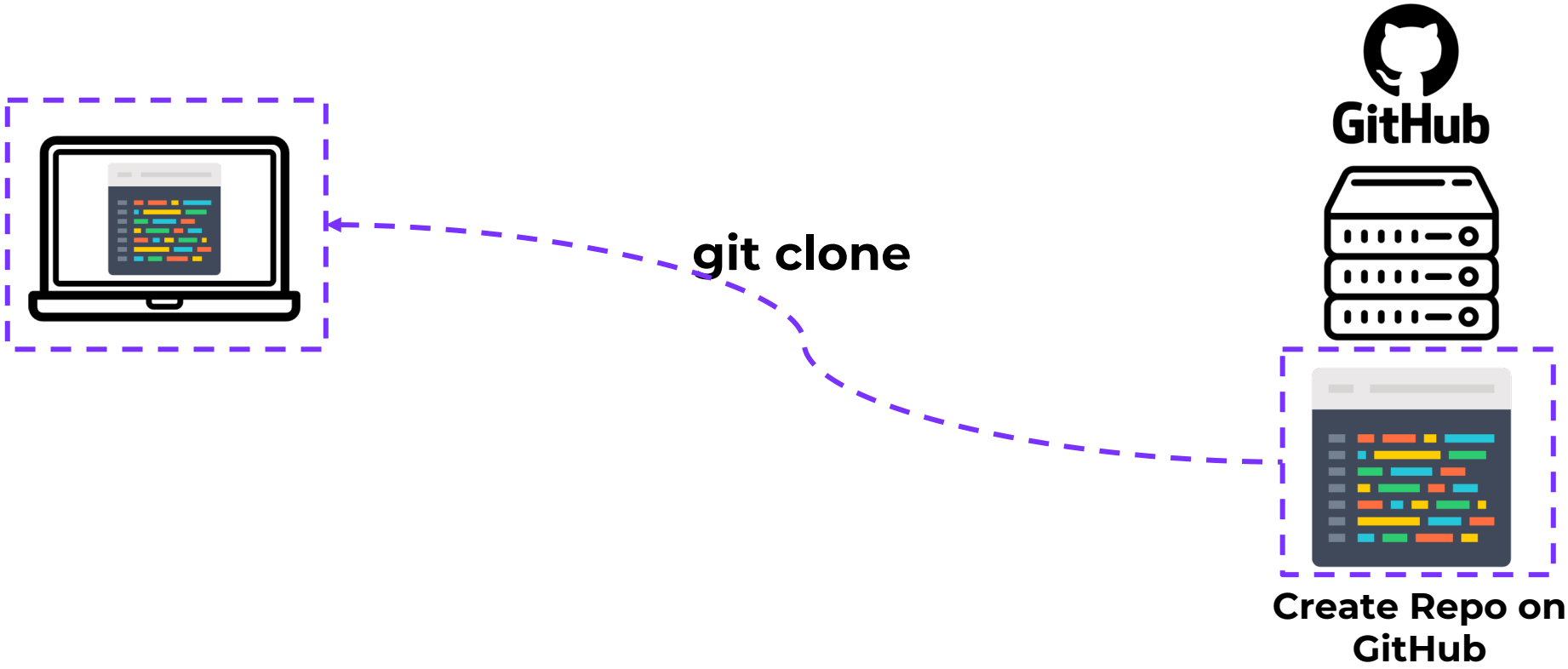


Week 2 Getting Started with Git

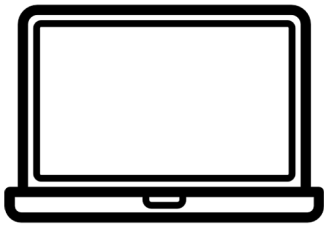




Week 2 Getting Started with Git



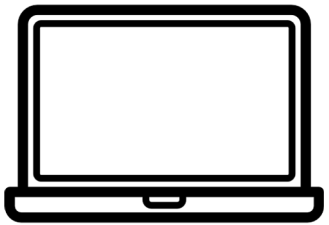
Week 2 Getting Started with Git



Week 2 Getting Started with Git

Week 2 Getting Started with Git

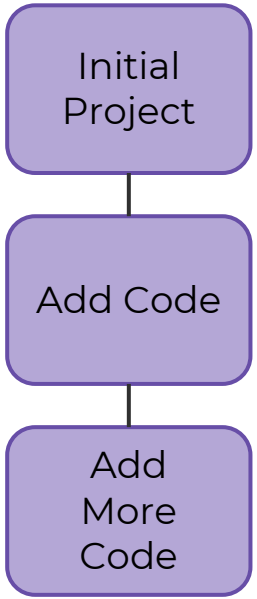
Week 2 Getting Started with Git



What we need to learn toWeek:

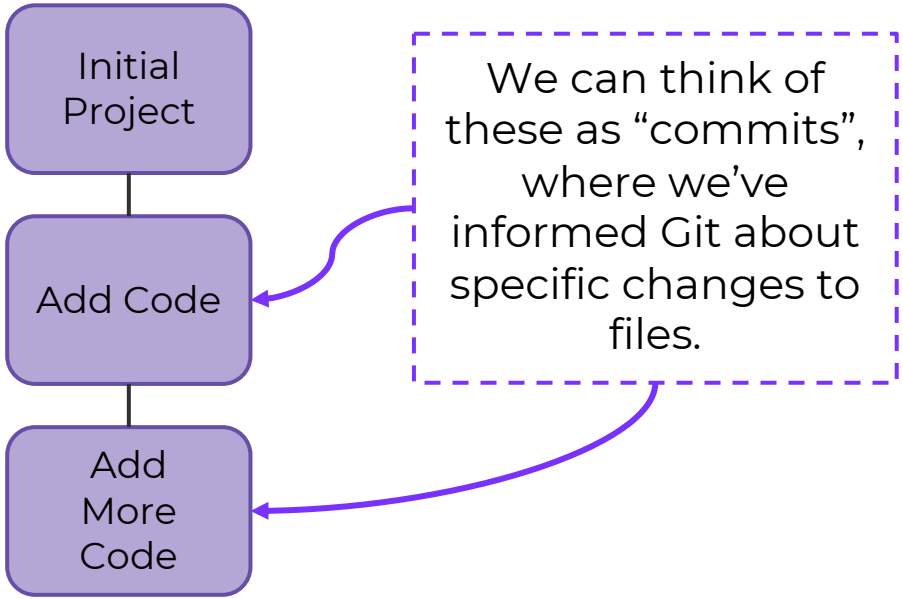
- Git Workflow
- How to tell Git about changes to our code
- How to push changes to GitHub
- How to pull changes from GitHub

Week 2 Getting Started with Git



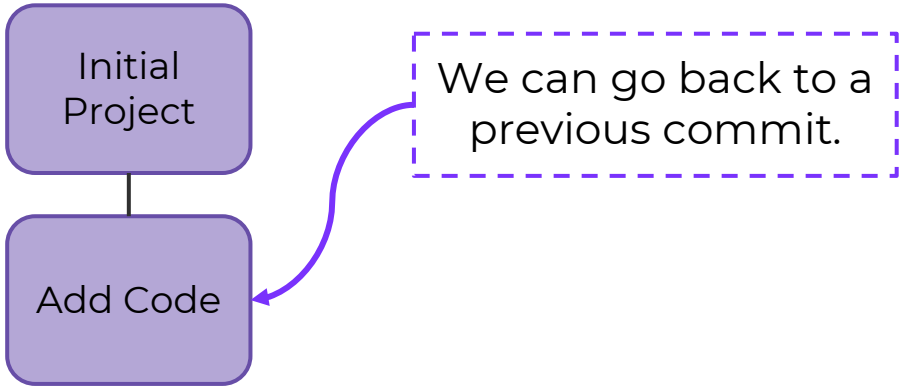

```
Add Code

More Code
```

Week 2 Getting Started with Git



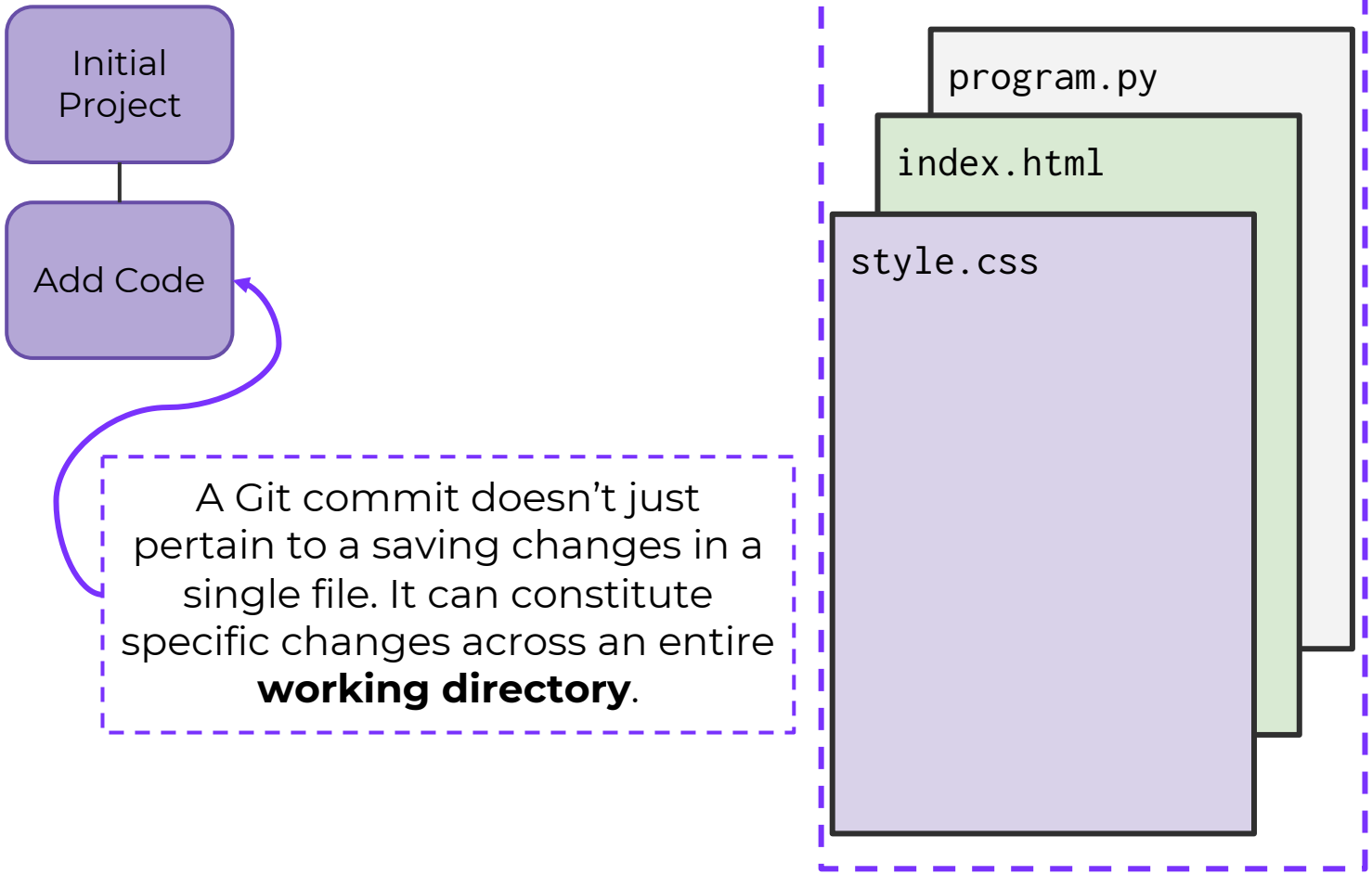

```
Add Code

More Code
```

Week 2 Getting Started with Git



Add Code

Week 2 Getting Started with Git



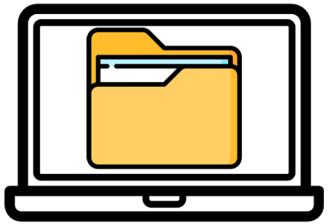
A Git commit doesn't just pertain to a saving changes in a single file. It can constitute specific changes across an entire **working directory**.

Week 2
Add and Commit

Week 2 Getting Started with Git

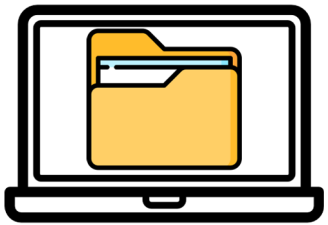
Week 2 Getting Started with Git

Working Directory



Week 2 Getting Started with Git

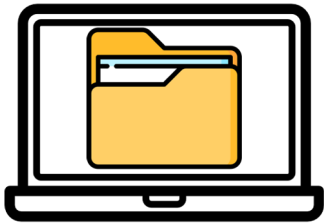
Working Directory



`program.py`

Week 2 Getting Started with Git

Working Directory



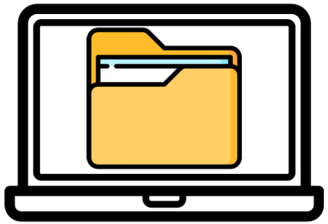
program.py

index.html

style.css

Week 2 Getting Started with Git

Working Directory



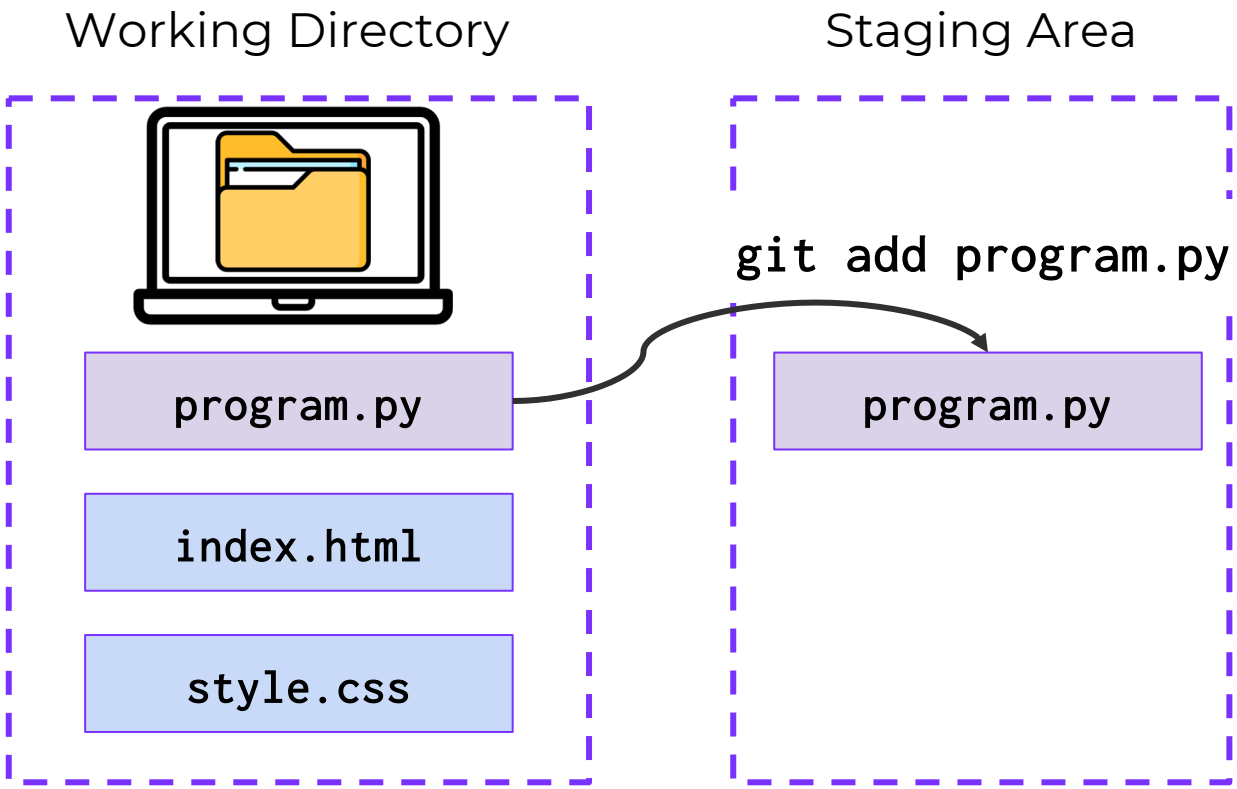
`program.py`

`index.html`

`style.css`

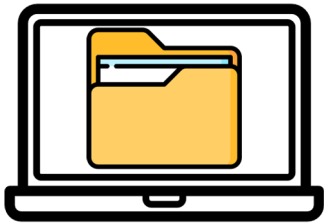
Staging Area

Week 2 Getting Started with Git



Week 2 Getting Started with Git

Working Directory



program.py

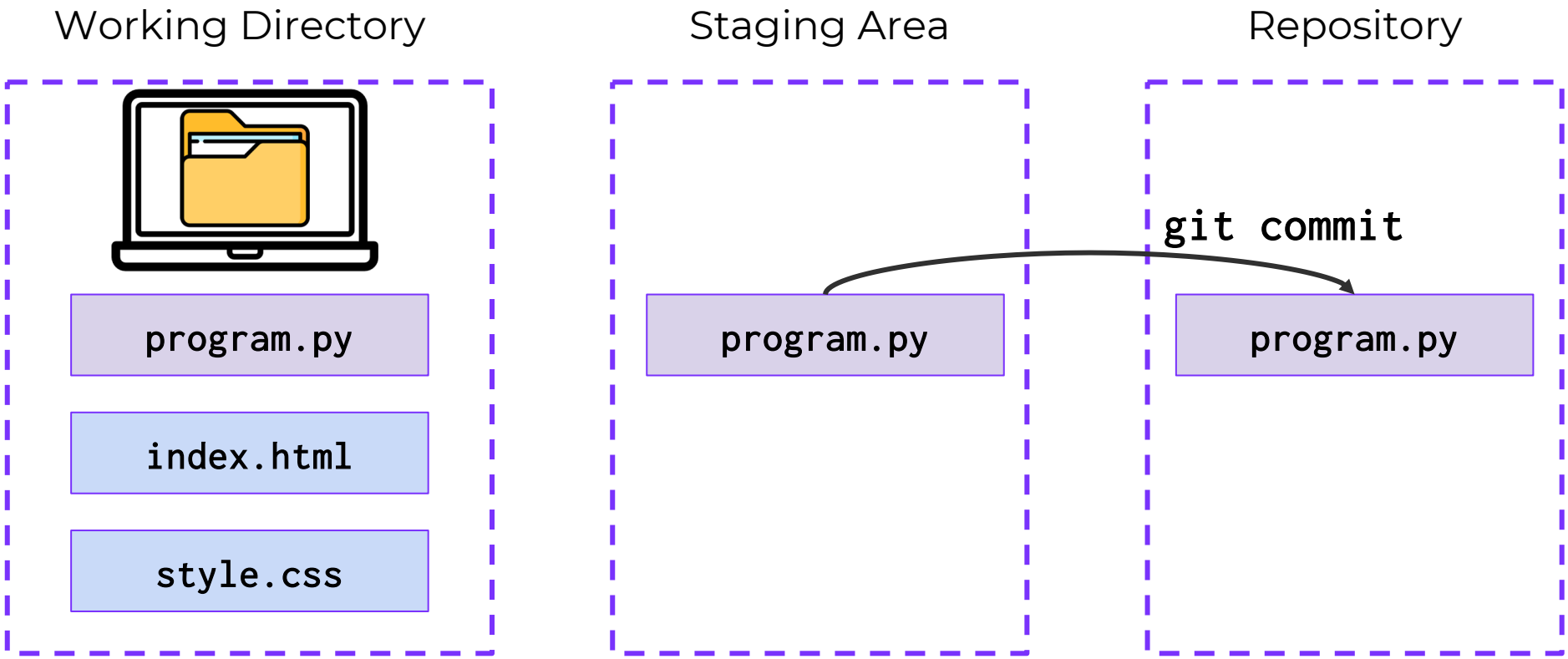
index.html

style.css

Staging Area

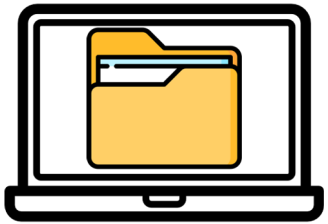
program.py

Week 2 Getting Started with Git



Week 2 Getting Started with Git

Working Directory



`program.py`

`index.html`

`style.css`

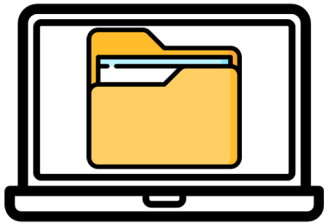
Staging Area

Repository

`program.py`

Week 2 Getting Started with Git

Working Directory



program.py

index.html

style.css

Staging Area

program.py

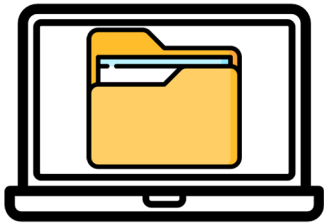
Repository

program.py

git commit -m “python code”

Week 2 Getting Started with Git

Working Directory



program.py

index.html

style.css

Staging Area

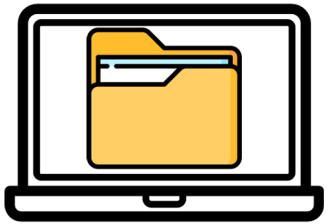
Repository

program.py

“python code”

Week 2 Getting Started with Git

Working Directory



program.py

index.html

style.css

Staging Area

git add .

index.html

style.css

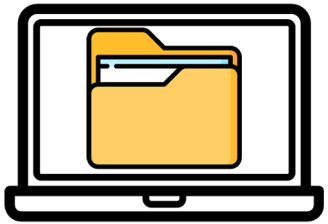
Repository

program.py

“python code”

Week 2 Getting Started with Git

Working Directory



program.py

index.html

style.css

Staging Area

index.html

style.css

Repository

program.py

“python

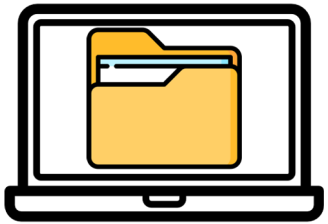
git commit -m “site files”

index.html

style.css

Week 2 Getting Started with Git

Working Directory



program.py

index.html

style.css

Staging Area

Repository

program.py

“python code”

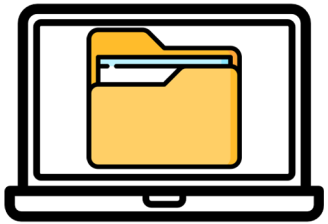
index.html

style.css

“site files”

Week 2 Getting Started with Git

Working Directory



program.py

index.html

style.css

Repository

program.py

“python code”

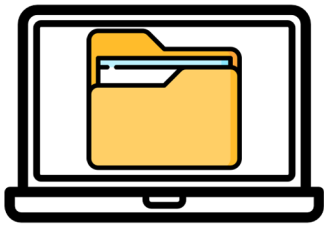
index.html

style.css

“site files”

Week 2 Getting Started with Git

Working Directory



program.py

index.html

style.css

Repository

program.py

“python code”

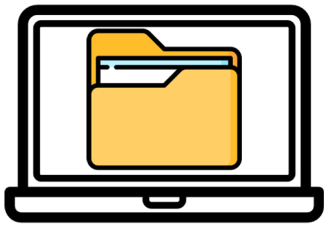
index.html

style.css

“site files”

Week 2 Getting Started with Git

Working Directory



program.py

index.html

style.css

Repository

program.py

“python code”

index.html

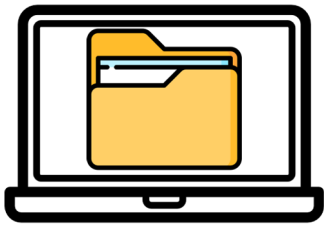
style.css

“site files”

git push

Week 2 Getting Started with Git

Working Directory



program.py

Repository

program.py

“python code”

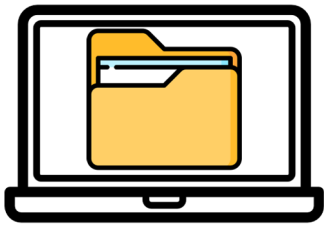
index.html

style.css

“site files”

Week 2 Getting Started with Git

Working Directory



program.py

git pull

Repository

program.py

“python code”

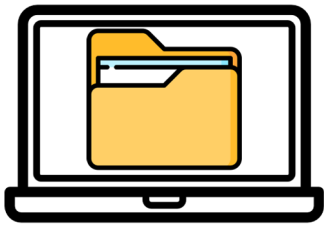
index.html

style.css

“site files”

Week 2 Getting Started with Git

Working Directory



program.py

index.html

style.css

git pull

Repository

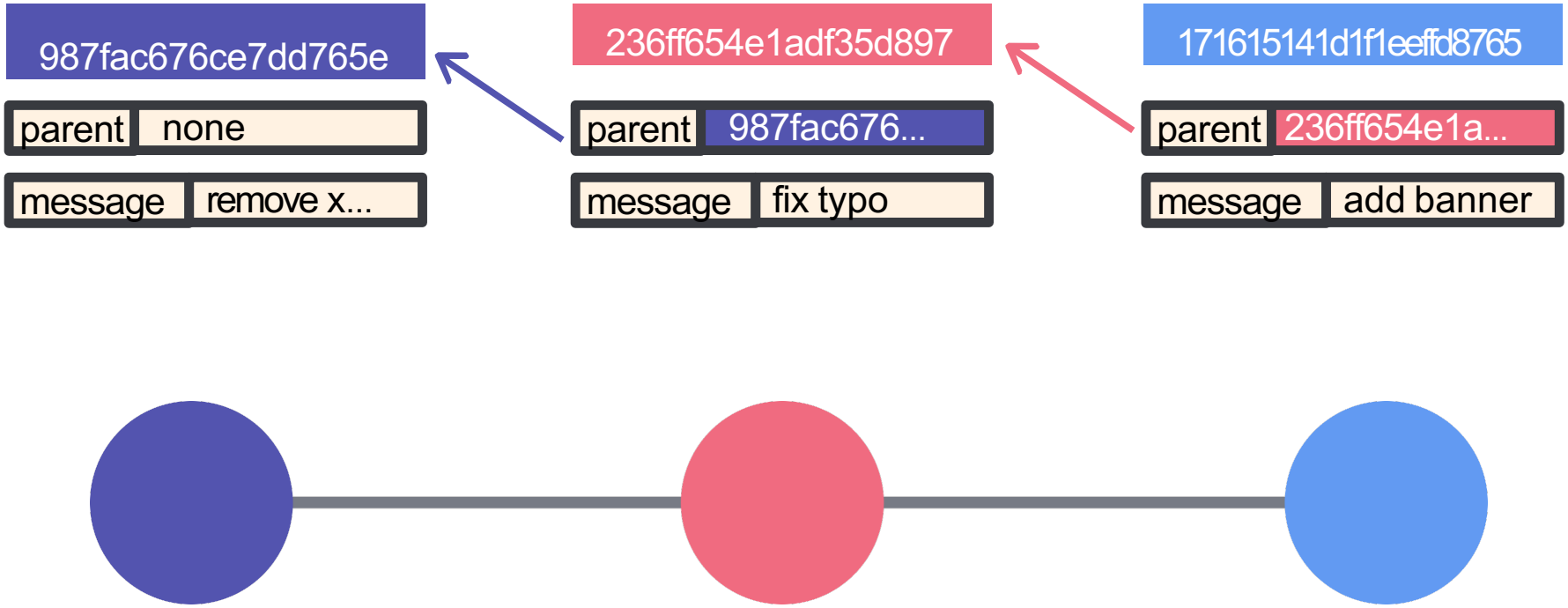
program.py

“python code”

index.html

style.css

“site files”


987fac676ce7dd765e
parent
none
message
remove x...
236ff654e1adf35d897
parent
987fac676...
message
fix typo
171615141d1f1eeffd8765
parent
236ff654e1a...
message
add banner

Week 2
Push and
Remote Branches

Week 2 Getting Started with Git

- We can check for remote branches with the command:
  - **git remote -v**
- If you run this command on a cloned repo, you will **view** the URL of the remote branch, like the GitHub URL.
- If there is no connection to a remote branch, then you won't see a URL.

Week 2 Getting Started with Git

- We tell git we want to add a remote branch using the git remote command syntax:
  - **git remote add name https://url.git**
- By convention, we call this remote branch the **origin** branch.
  - **git remote add origin https://url.git**
- You then replace the .git URL with the .git URL from the repository you created.

Week 2 Getting Started with Git

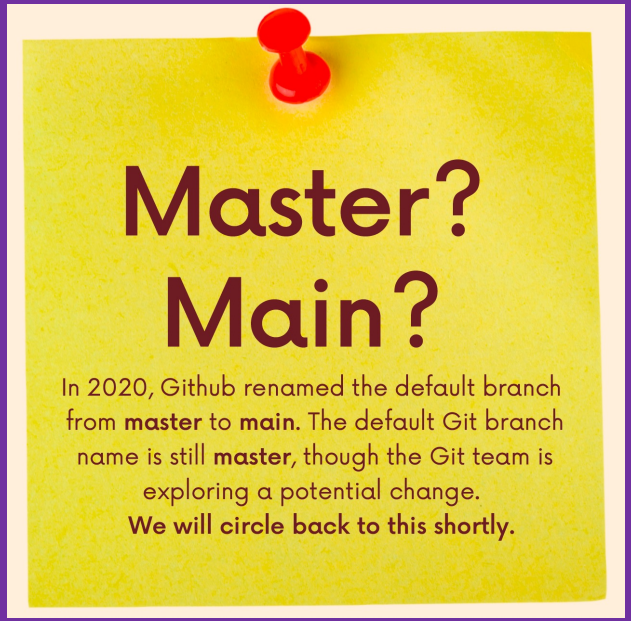
- **Important Note:**
  - We won't use these commands in the video, but just in case you need them in the future:
    - **git remote rename <old> <new>**
    - **git remote remove <name>**

Week 2 Getting Started with Git

- Once we've connected to our remote branch on GitHub, we can **push** our code to the remote branch.
- We tell git to push to the remote main/master branch called origin with the command:
  - **git push -u origin main/master**

Week 2 Getting Started with Git

- **Important Note:**
  - GitHub has officially changed the naming convention of its **master** branch to **main** branch.
  - You'll see this reflected in the instructions that GitHub provides:
    - **git branch -M main**


Master?
Main?
In 2020, Github renamed the default branch from **master** to **main.** The default Git branch name is still **master**, though the Git team is exploring a potential change.
We will circle back to this shortly.

Couple Years Back..

Branch: master

LOCAL DESKTOP

Branch: master

GitHub

Improper Reference

Branch: master

LOCAL
DESKTOP

Branch: main

GitHub

Clone

Branch: main

LOCAL
DESKTOP

git clone

Branch: main

GitHub

Local Folder to GitHub without Clone

Branch: master

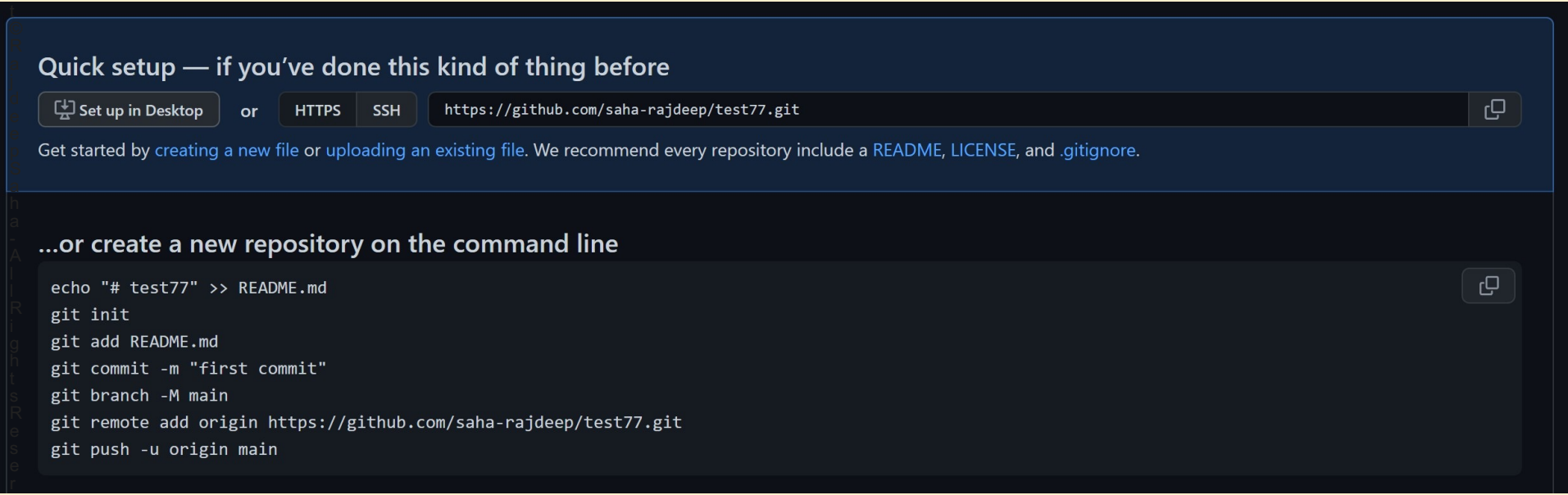
LOCAL
DESKTOP

git push
git pull

Branch: main

GitHub

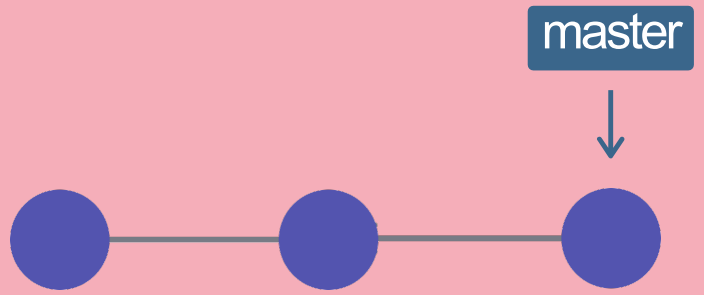
Local Folder to GitHub without Clone



A Closer Look At Cloning

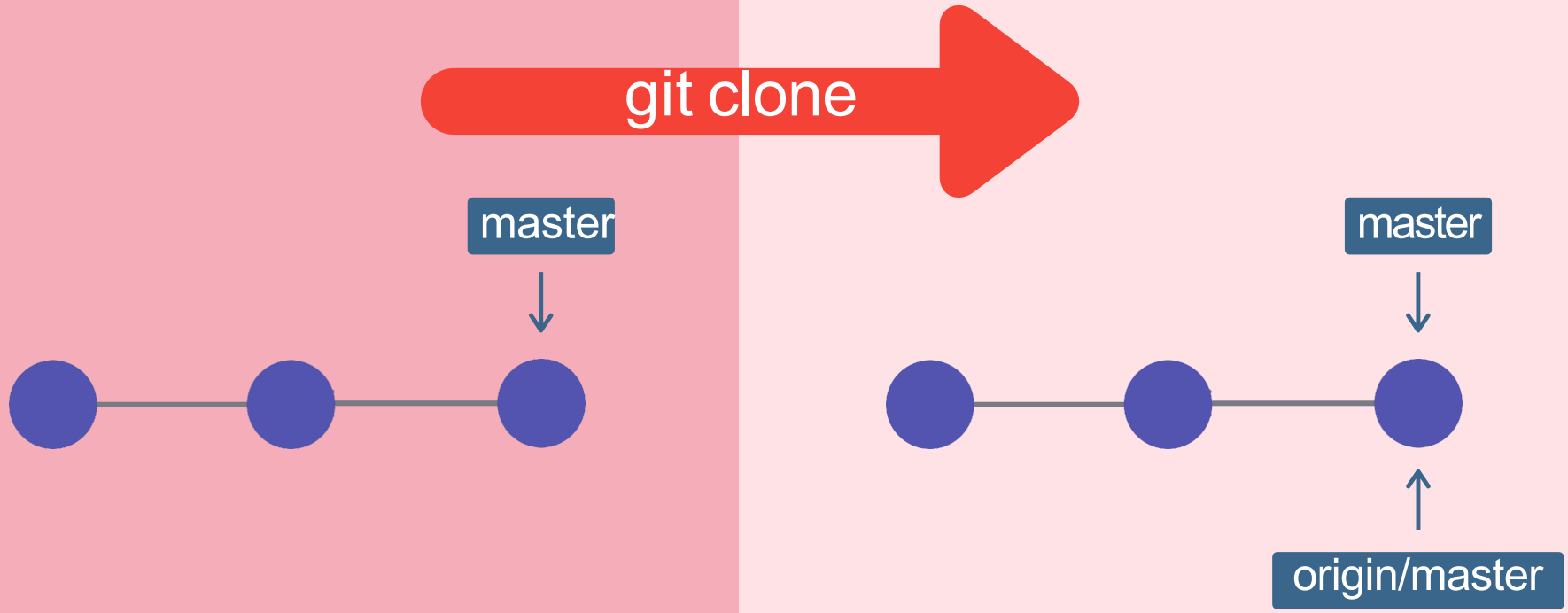
Github Repo

My Computer

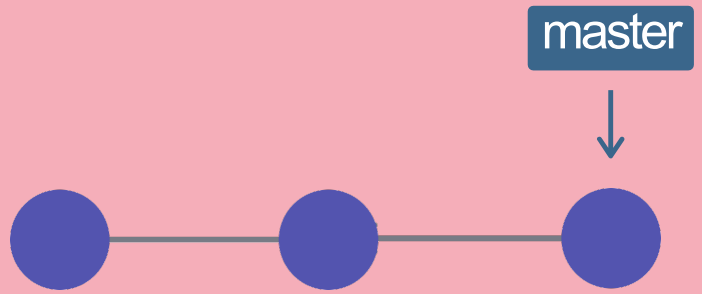


Github Repo

My Computer

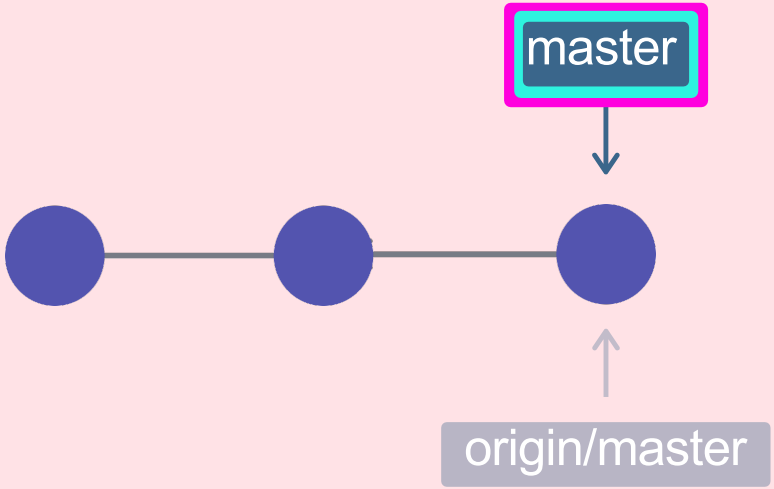


Github Repo



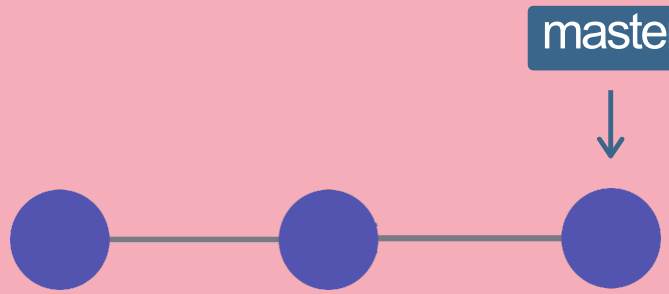
My Computer

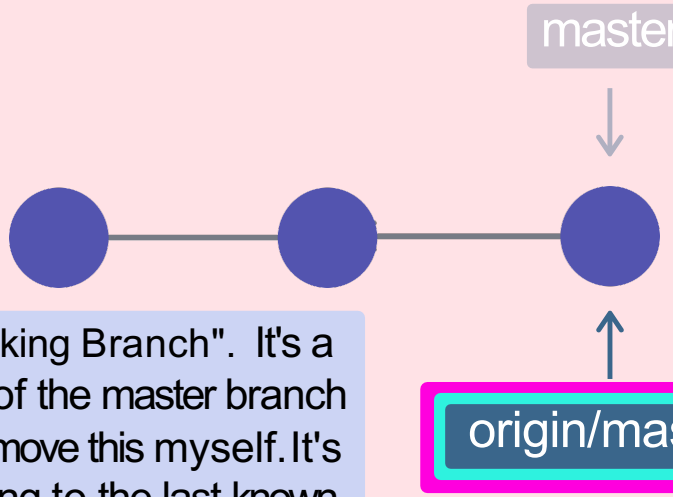
A regular branch reference.
I can move this around myself.



Github Repo

My Computer





This is a "Remote Tracking Branch". It's a reference to the state of the master branch on the remote. I can't move this myself. It's like a bookmark pointing to the last known commit on the master branch on origin

Remote Tracking Branches

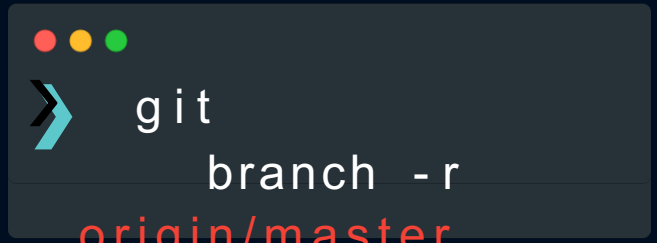
"At the time you last communicated with this remote repository, here is where x branch was pointing"

They follow this pattern <remote>/<branch>.

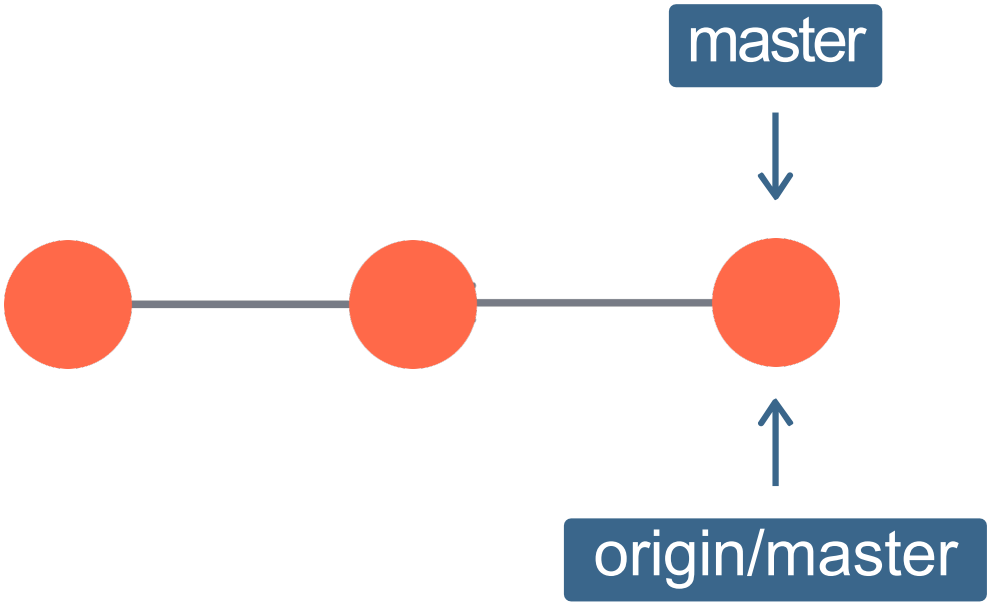
- origin/master references the state of the master branch on the remote repo named origin.
- upstream/logoRedesign references the state of the logoRedesign branch on the remote named upstream (a common remote name)

Remote Branches

Run git branch -r to view the remote branches our local repository knows about.

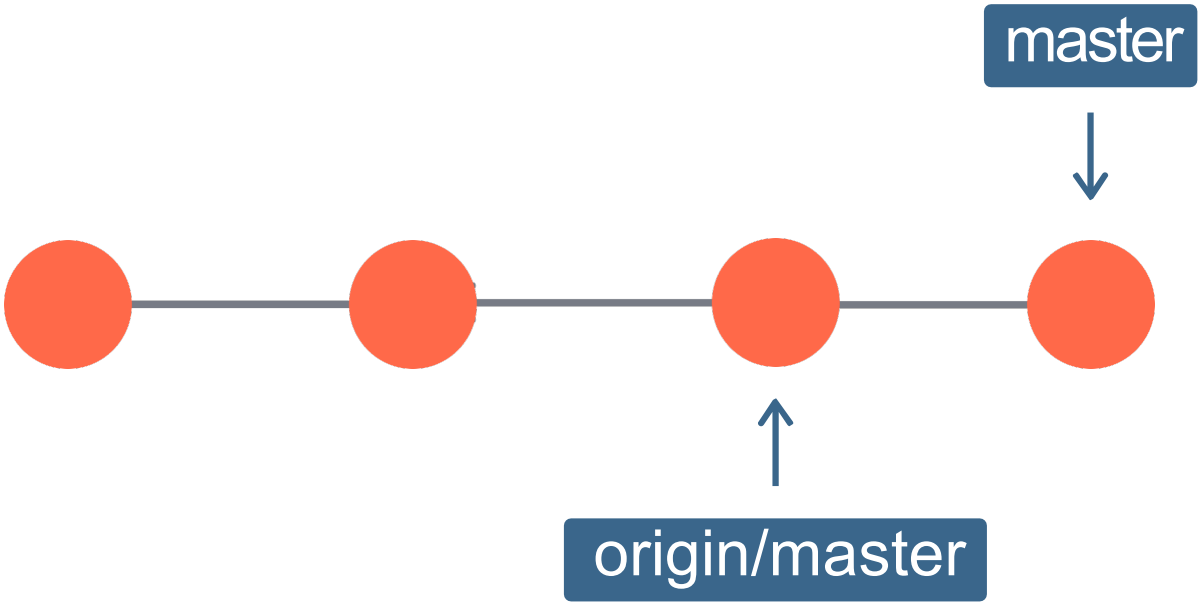


My Computer



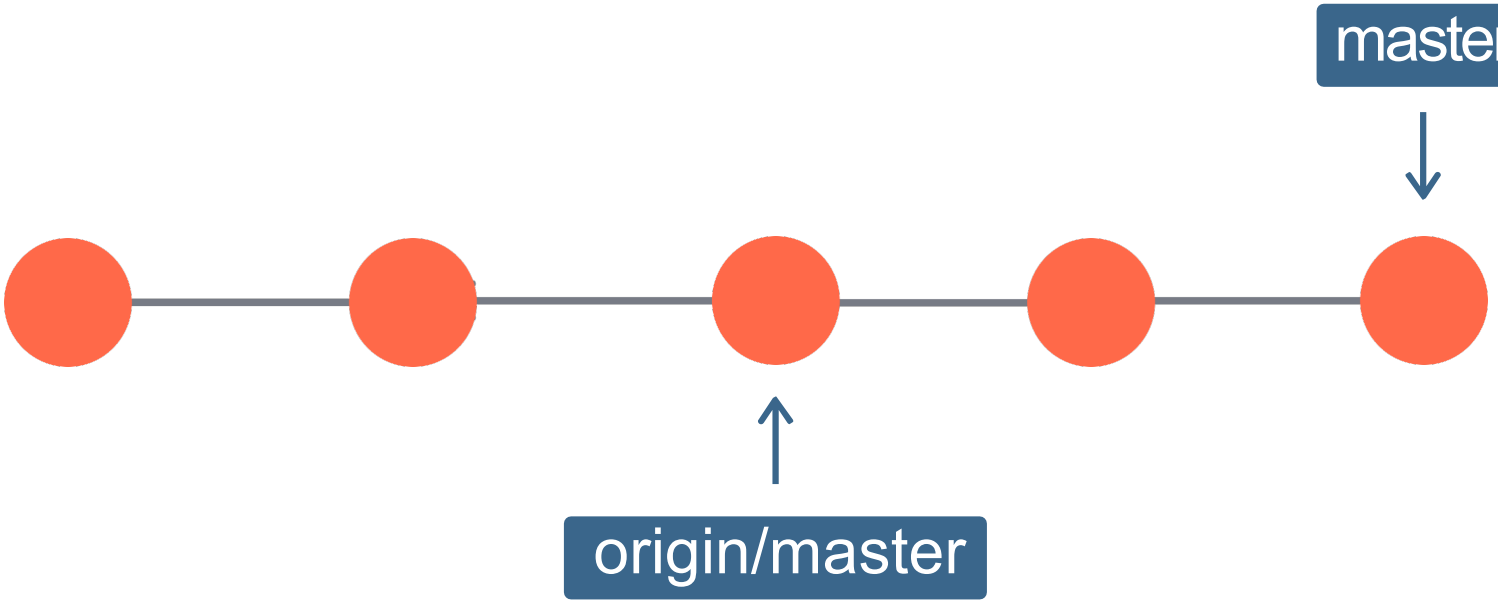
My Computer

I make a new commit locally. My master branch reference updates, like always.



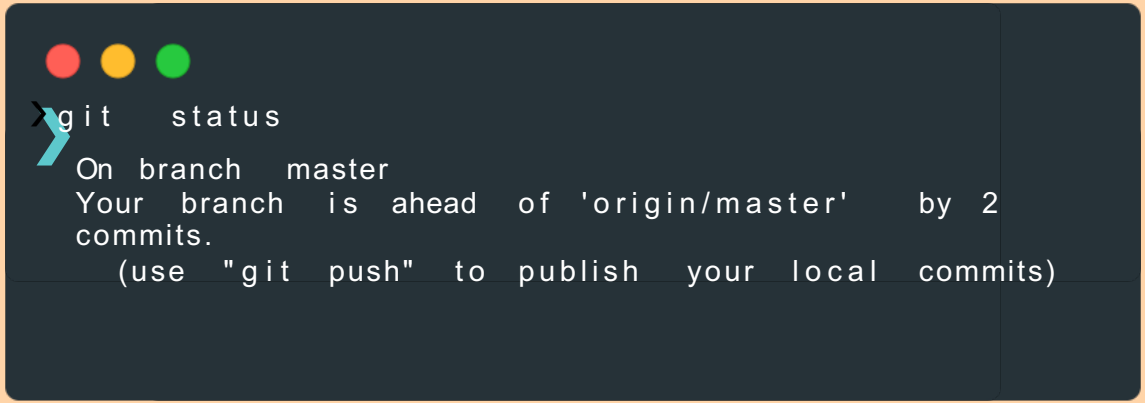
The remote reference stays the same

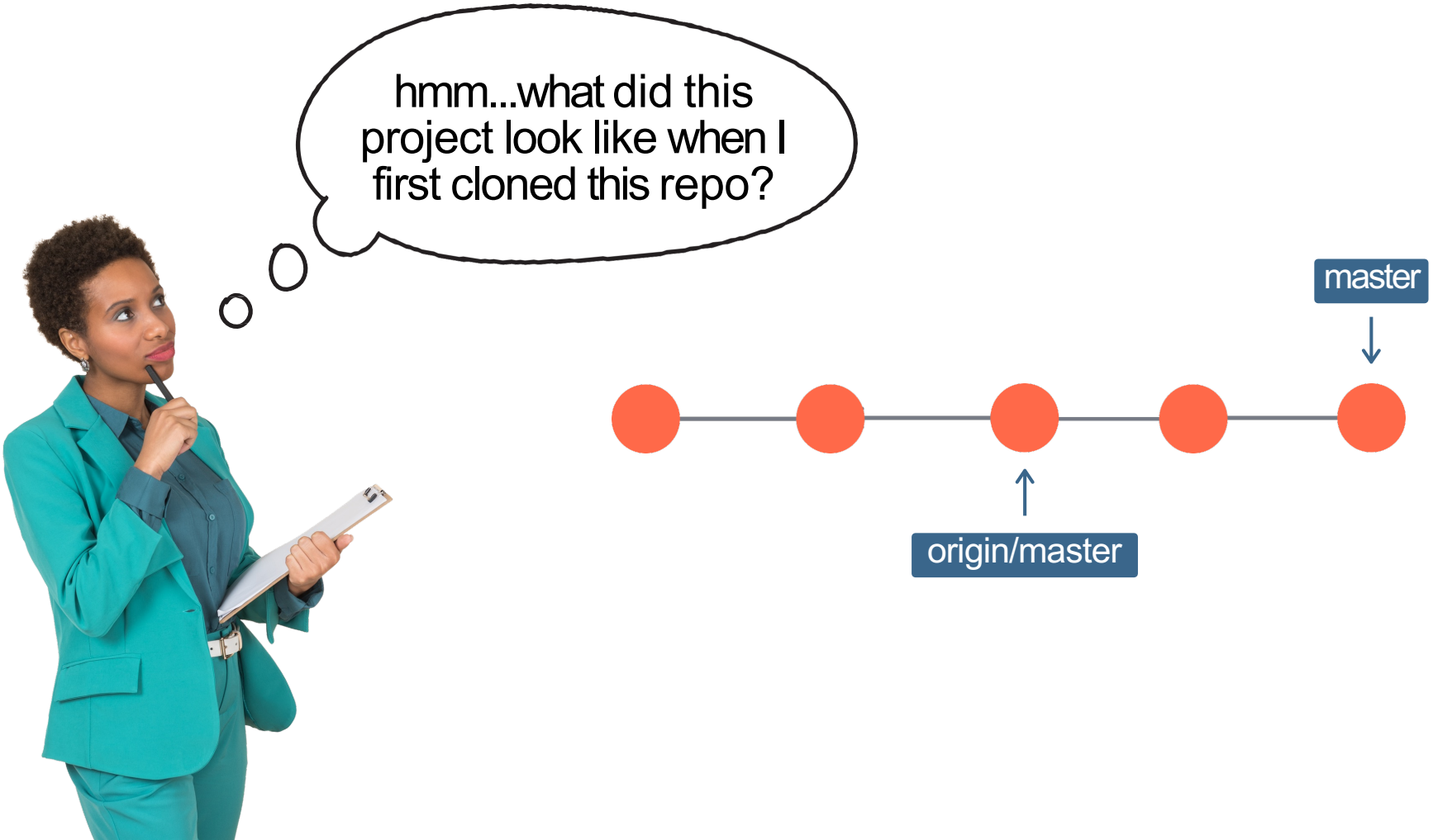
My Computer



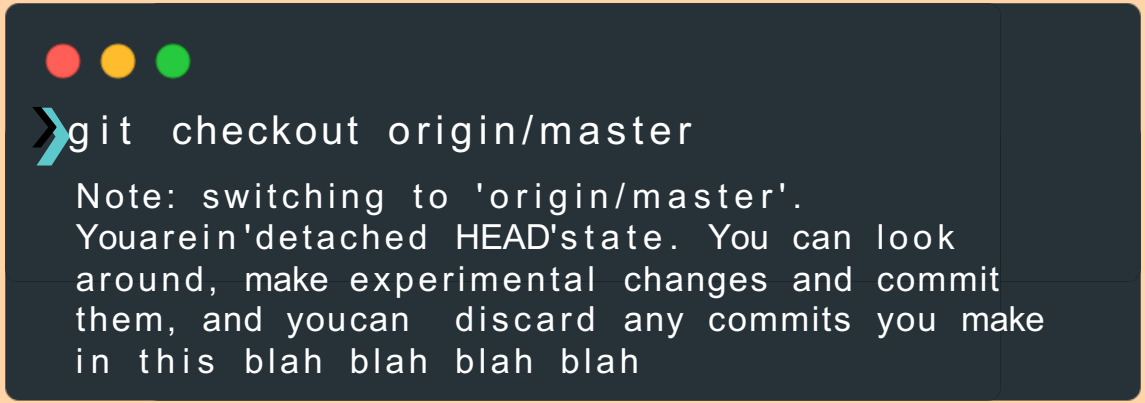
Remote reference doesn't move!

When I run git status




hmm...what did this project look like when I first cloned this repo?
master
origin/master

You can checkout these remote branch pointers



Detached HEAD! Don't panic. It's fine.

Remote Branches

Once you've cloned a repository, we have all the data and Git history for the project at that moment in time. However, that does not mean it's all in my workspace!
The github repo has a branch called puppies, but when I run git branch I don't see it on my machine! All I see is the master branch. What's going on?

Week 2
Git Log

Week 2 Getting Started with Git

- Before we jump into using git fetch and git pull, let's quickly show you how to use **git log**.
- The **git log** command will show a list of all the commits made to a repository, including the hash, message, and metadata.
- Think of it as the history of a repo.

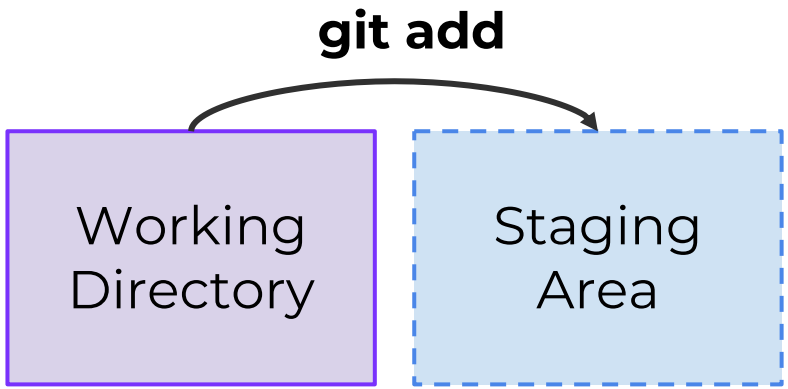
Week 2
Fetch and Pull

Week 2 Getting Started with Git

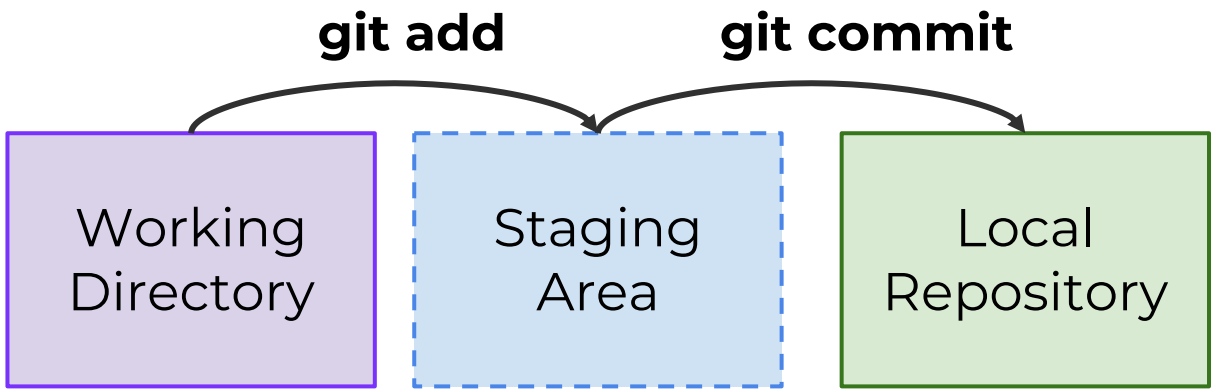
- There are two options of getting repository changes from a remote branch (like the remote branch on GitHub).
  - **git pull**
  - **git fetch**

Week 2 Getting Started with Git

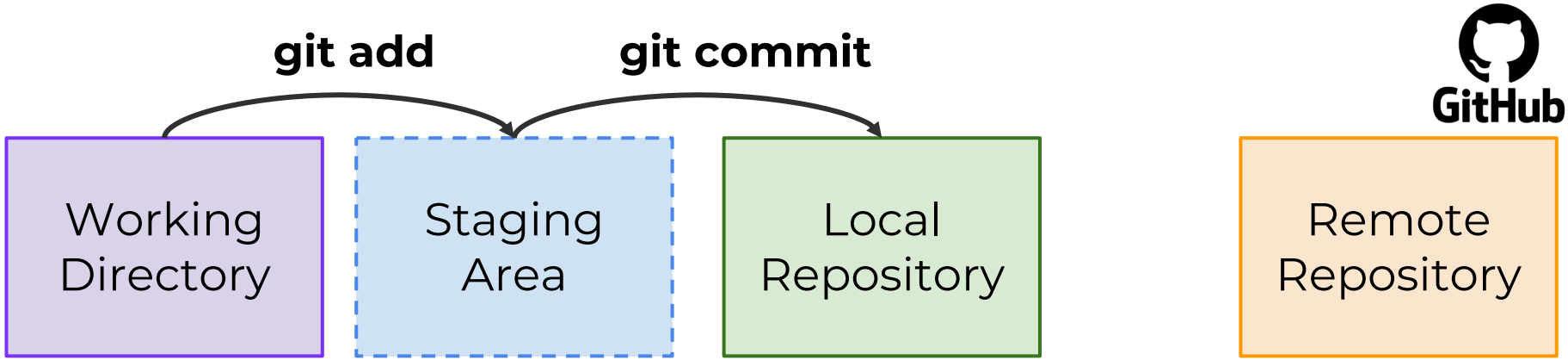
Week 2 Getting Started with Git



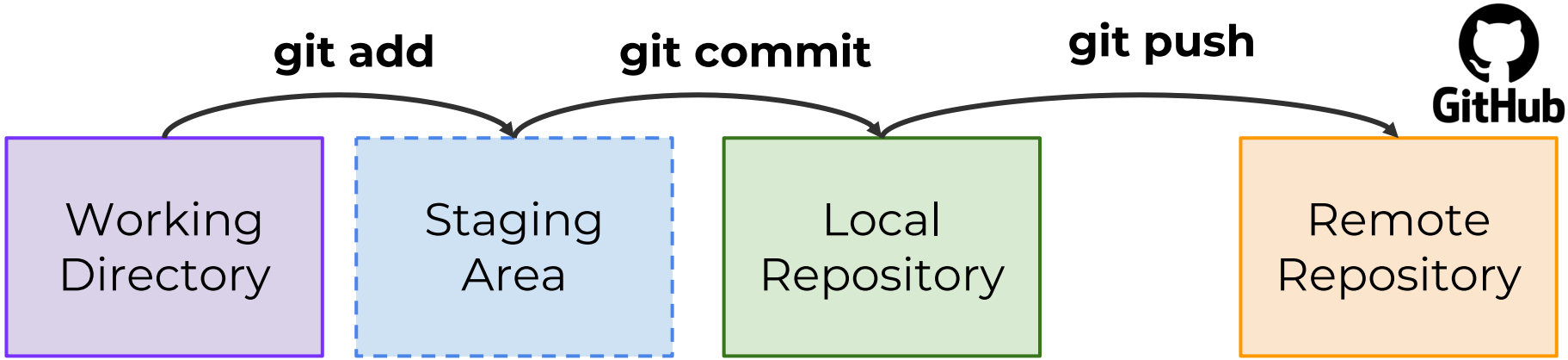
Week 2 Getting Started with Git



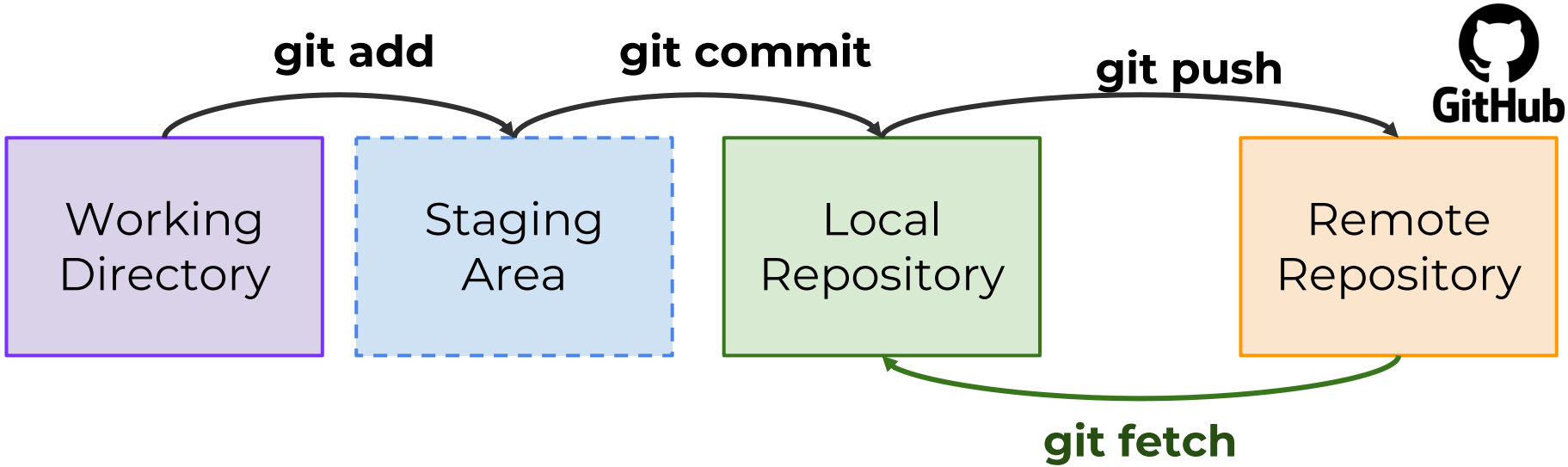
Week 2 Getting Started with Git



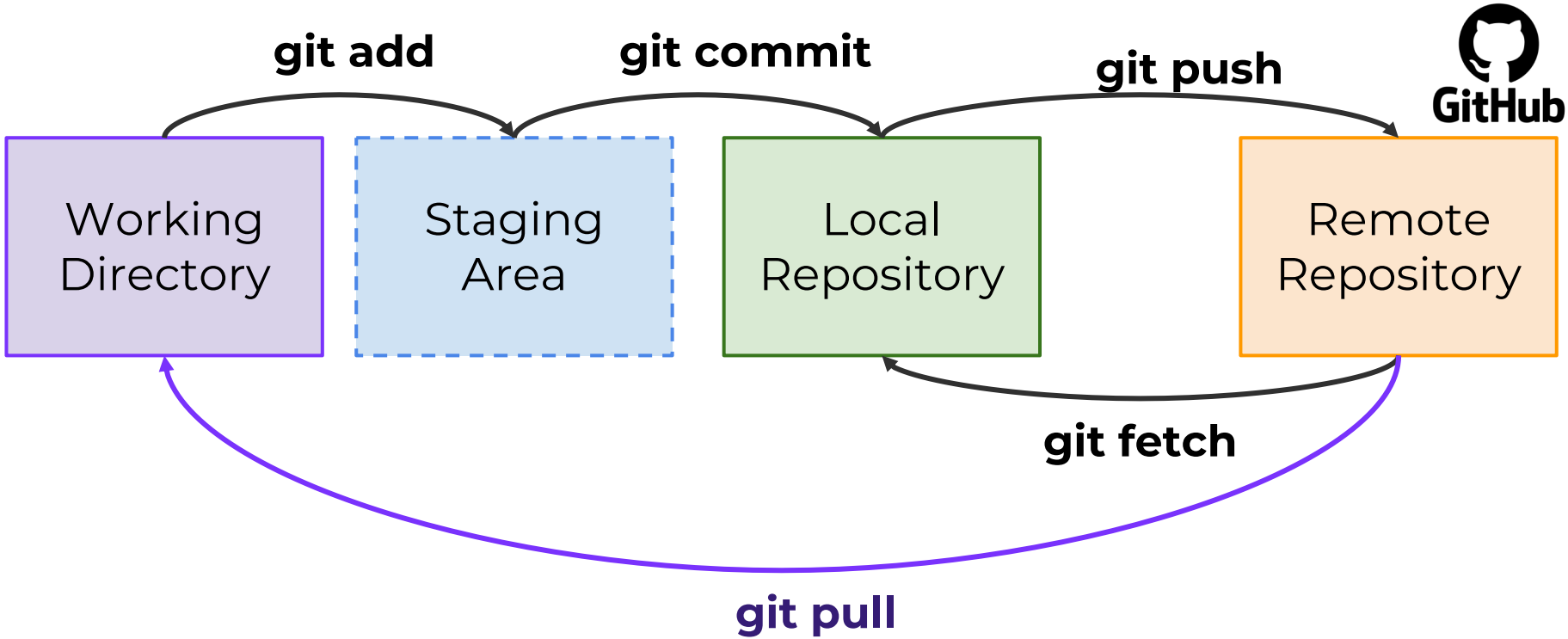
Week 2 Getting Started with Git



Week 2 Getting Started with Git



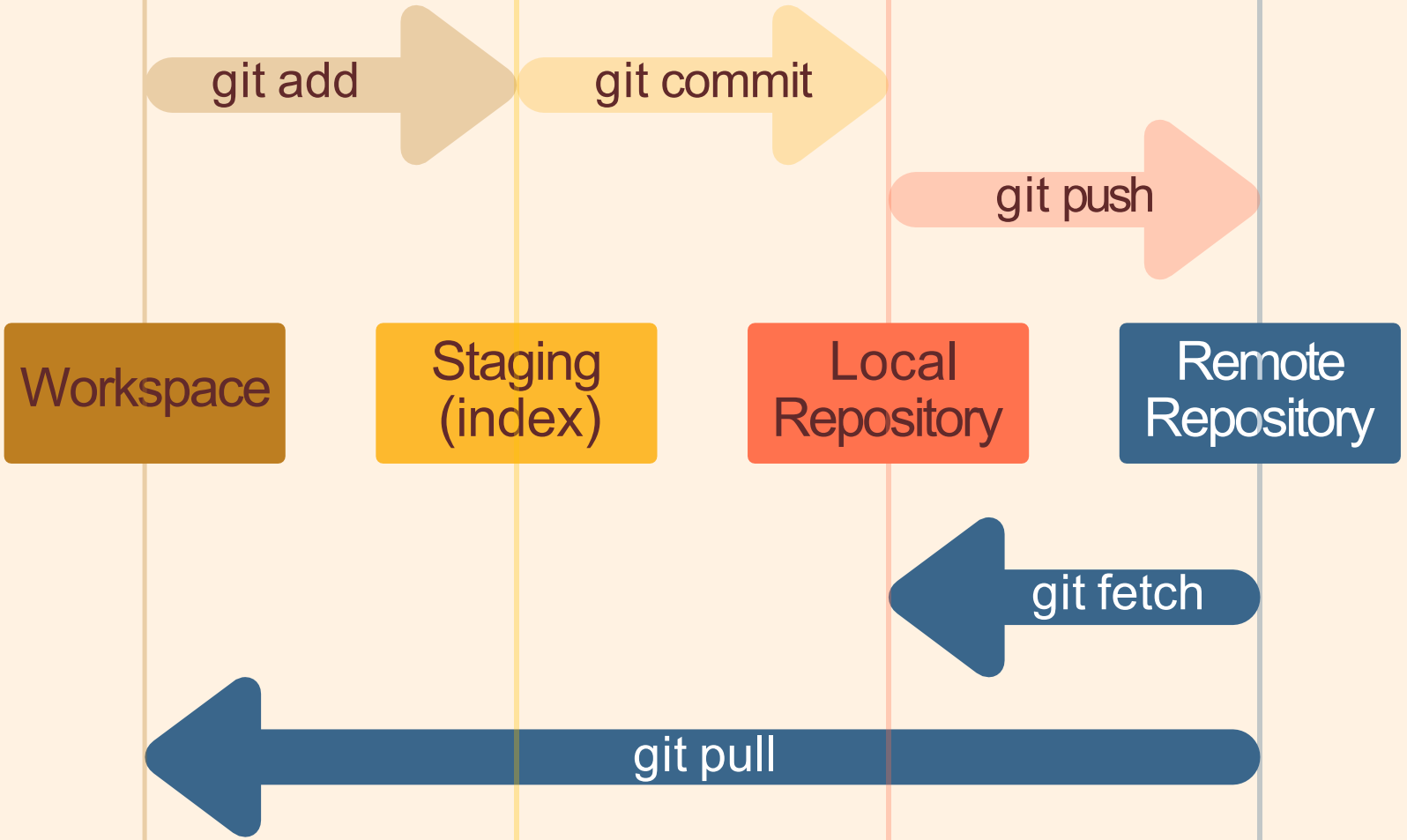
Week 2 Getting Started with Git



git pull = git fetch + git merge

update the remote tracking branch with the latest changes from the remote repository

update my current branch with whatever changes are on the remote tracking branch


git add
git commit
git push
Workspace
Staging (index)
Local Repository
Remote Repository
git fetch
git pull

git fetch

- Gets changes from remote branch(es)
- Updates the remote-tracking branches with the new changes
- Does not merge changes onto your current HEAD branch
- Safe to do at anytime

git pull

- Gets changes from remote branch(es)
- Updates the current branch with the new changes, merging them in
- Can result in merge conflicts
- Not recommended if you have uncommitted changes!

Week 2
Exercise