
INTRO TO SOFTWARE DEV

3 ESSENTIALS OF PROGRAMMING

PROGRAMMING

COMPUTER SKILLS

VERSION CONTROL AND COLLABORATION


COMPUTER BASICS AND
COMMANDLINE INTERFACE

COMMAND LINE INTERFACE(CLI)

What is CLI

Why learn it?

Why not GUI

WHAT IS CLI?

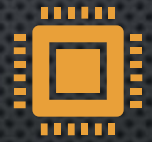
- Text-based interface for interacting with a computer
- Execute commands via typing
- Found in most OS
  - Linux/macOS Terminal
  - Windows Command Prompt or PowerShell

WHY USE IT

Efficiency – Faster than GUI for a lot of tasks

Automation – use scripts to automate your work

Remote Access- remotely connect to servers and manage remote systems



More control – Fine-grained control over systems and various processes

(Bonus) You look cool doing it

GETTING STARTED

- Look for 'Command Prompt' or 'PowerShell' Somewhere in the world
  - Or use git bash
- macOS: Open 'Terminal' from Applications -> Utilities (this is how it is on mine)
- Linux: Press 'Ctrl + Alt+ T'

BASIC COMMANDS

1. `pwd` - Print working directory (shows where you are)

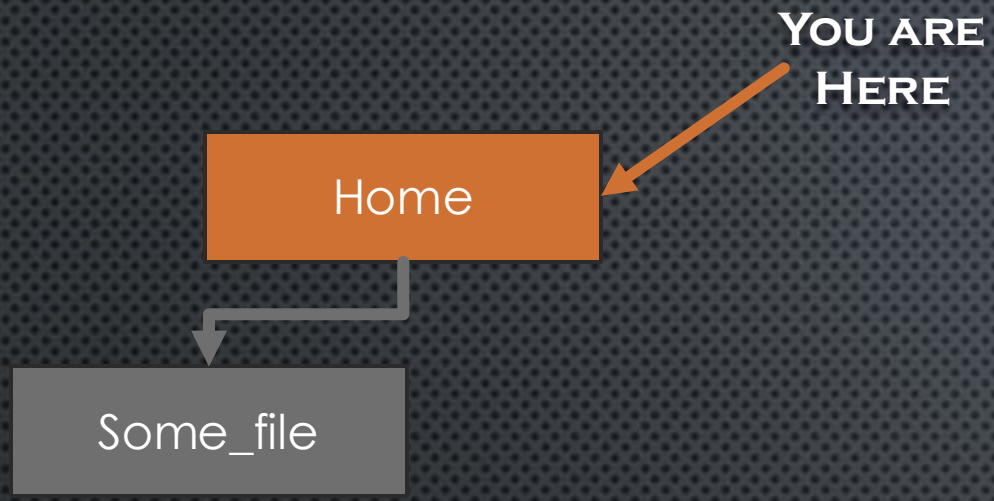
2. `ls` - List files and folders

3. `cd <directory>` - Change directory

4. `mkdir <folder_name>` - Create a new folder

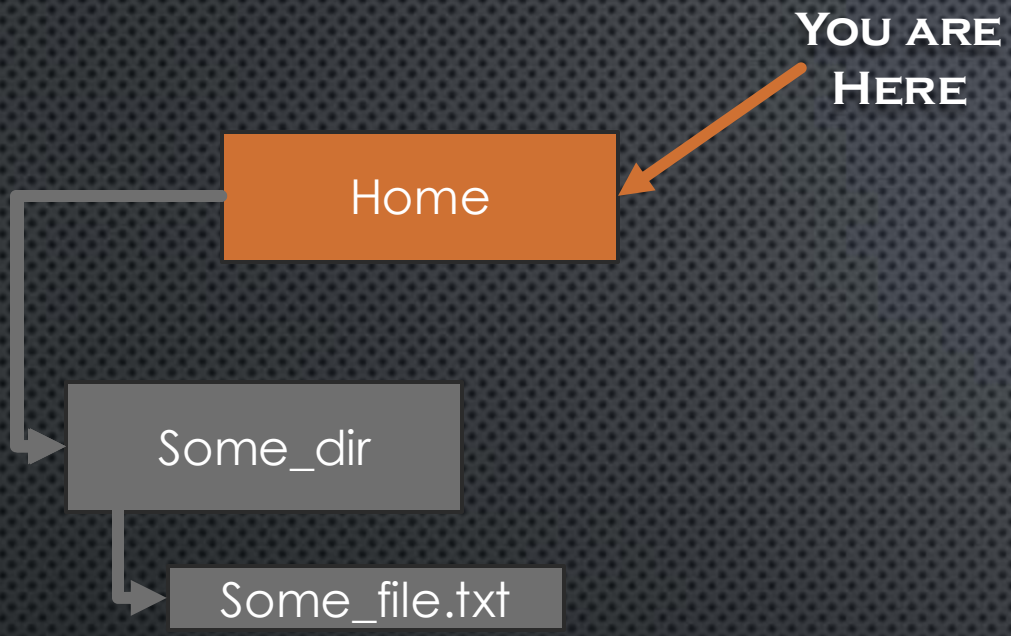
5. `touch <file_name>` - Create an empty file

6. `rm <file_name>` - Delete a file



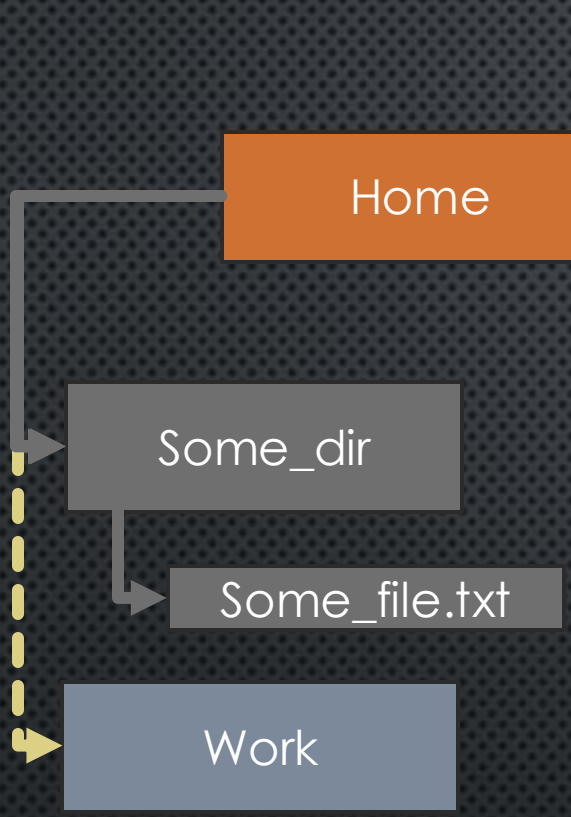
Navigating files and directories

- When you are in the terminal, you are in a file on your computer
  - Yes: just about everything is a file ESPECIALLY in Linux
- When moving around just remember you are in a file



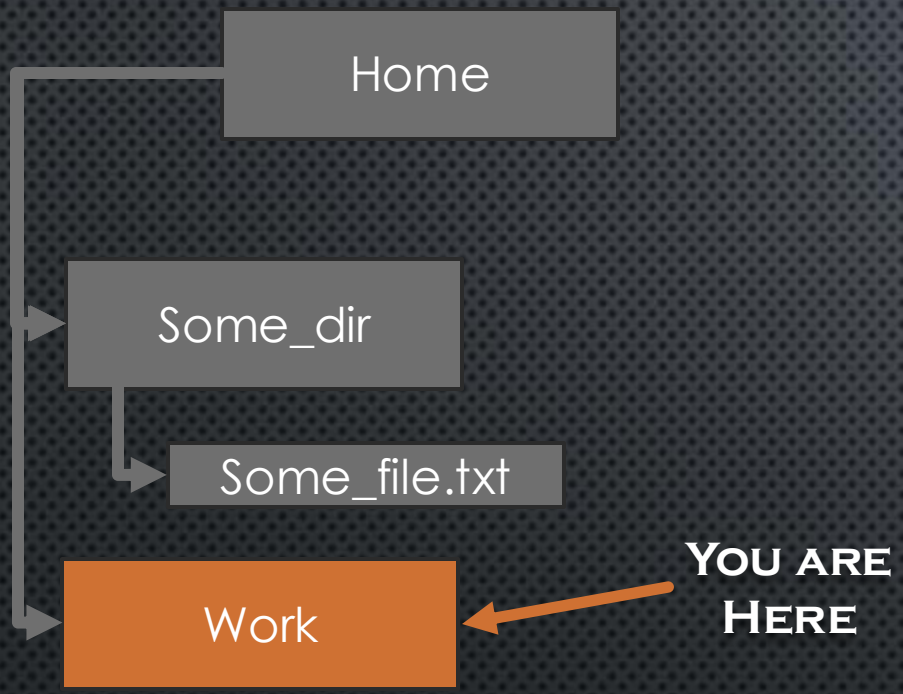
Navigating files and directories- example

- To make a work directory with a test file
  - 1) First make sure you are where you want to be
    - Type `pwd` to see where you are



Navigating files and directories- example

- To make a work directory with a test file
  - 1) First make sure you are where you want to be
    - Type `pwd` to see where you are
  - 2) If you are where you want to make the file, type `mkdir work`
    - This makes the directory



Navigating files and directories- example

- To make a work directory with a test file
  - 1) First make sure you are where you want to be
    - Type `pwd` to see where you are
  - 2) If you are where you want to make the file, type `mkdir work`
    - This makes the directory
  - 3) To move to work and make your file
    - Type 'cd work'

Navigating files and directories- example

- To make a work directory with a test file
  - 1) First make sure you are where you want to be
    - Type `pwd` to see where you are
  - 2) If you are where you want to make the file, type `mkdir work`
    - This makes the directory
  - 3) To move to work and make your file
    - Type 'cd work'
  - 4) now you can Run `touch test.txt` to make empty file to edit.


VERSION CONTROL AND
COLLAB

So what is GitHub?

Cloud software dev platform

- Uses git for version control
- Place to create/store/manage/share code
- Provides many, many useful features to manage projects

Why Should I care?

- Very useful/powerful
- It will save you
- Fomo
  - everyone does it
- great For Resume and Employers love it
  - (The Real Reason)

Introduction to Git

What is it?

- Distributed, open-Source Version Control system

What does it do?

- Collaborate/Manage Projects
- Track Code Changes
- Supports Nearly Every
  - OS
  - Commandline tool
  - Development Env

GIT BASICS – GETTING STARTED

- First Step- Install Git

To use Git on your local machine, you need to install Git. Here are the steps to install Git on your machine:

Windows

1. Download the latest version of Git from the official Git website.
2. Run the installer and follow the instructions.
3. Make sure to enable the option to use Git from the Windows Command Prompt. This will allow you to use Git from the Command Prompt or Git Bash.

Mac

0. You could use the same method as windows, but I would recommend using Homebrew to install Git.
1. Open the Terminal application.
2. Use the following command to install Git using Homebrew:

```
brew install git
```

Linux

1. Use the package manager to install Git. For example, on Ubuntu/Debian, you can run the following command:

```
sudo apt-get update
sudo apt-get install git
```

GIT BASICS – GETTING STARTED

- First Step- Install Git

To use Git on your local machine, you need to install Git. Here are the steps to install Git on your machine:

Windows

1. Download the latest version of Git from the official Git website.
2. Run the installer and follow the instructions.
3. Make sure to enable the option to use Git from the Windows Command Prompt. This will allow you to use Git from the Command Prompt or Git Bash.

Mac

0. You could use the same method as windows, but I would recommend using Homebrew to install Git.
1. Open the Terminal application.
2. Use the following command to install Git using Homebrew:

```
brew install git
```

Linux

1. Use the package manager to install Git. For example, on Ubuntu/Debian, you can run the following command:

```
sudo apt-get update
sudo apt-get install git
```

- Next – Configure your Git

After installing Git, you need to set up your Git username and email address. Here are the steps to set up Git:

1. Open the Git Bash or Terminal application.
2. Run the following commands to set up your Git username and email address:

```
git config --global user.name "Your Name"
git config --global user.email "
```

GIT BASICS – GETTING STARTED

- 2 METHODS TO CREATE REPO
  - CREATE/INITIALIZE PROJECT
    - CAN BE ENTIRE DIRECTORY
    - CAN BE EMPTY
  - CLONE/FORK
    - MAKES COPY OF REPO FROM GITHUB ONTO YOUR MACHINE

```
##########################
# Create Repo from scratch #
##########################

#Bash stuff to create new directory
mkdir new_project #make new directory
cd new_project    #change directory to new_project

git init #initialize git repository

#This is output after running init
>>> Initialized empty Git repository in /Users/username/new_project/.git/

#Now we have to connect the local repository to a remote repository
git remote add origin https://github.com/username/new_project.git #TA-DA! Now we have a remote repository
```

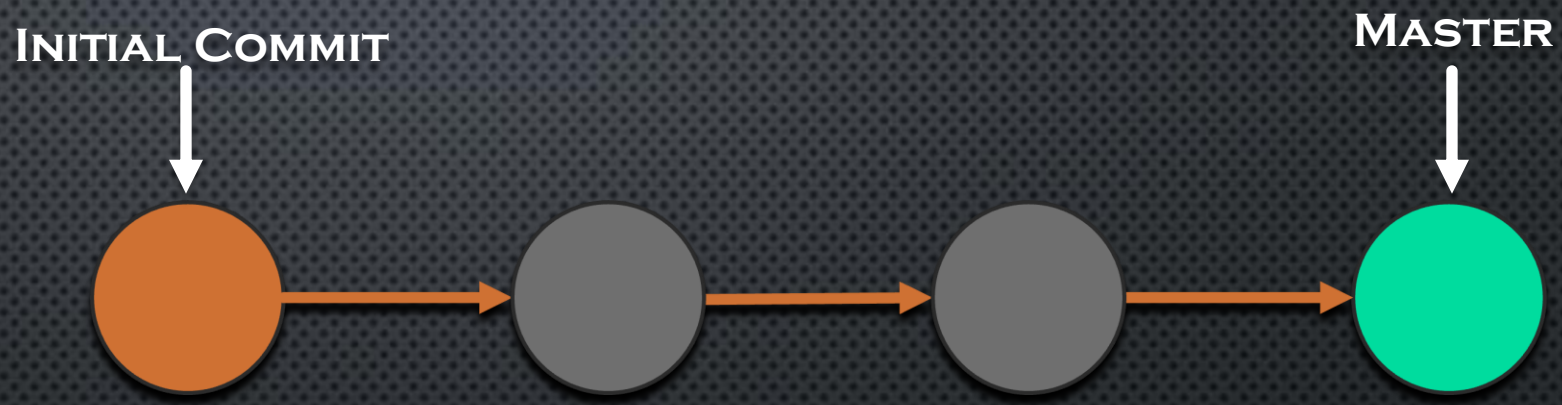
```
##########################
# Create Repo from existing #
##########################

#make sure you move to dir where you want the repo
git clone https://<url> #clones the repo to your local machine

>> Cloning into 'repo_name'...
>> some message about cloning

cd repo_name #move into the repo
```

GIT BASICS – BIG PICTURE

- KEEPS HISTORY OF YOUR ENTIRE PROJECT
  - THINK TIMELINE
- EACH NODE IS A SAVEPOINT
  - SNAPSHOT OF DIRECTORY
  - SAVES ALL FILES AT THAT POINT IN TIME
- WE CAN ALWAYS GO BACK TO A PREVIOUS SAVEPOINT

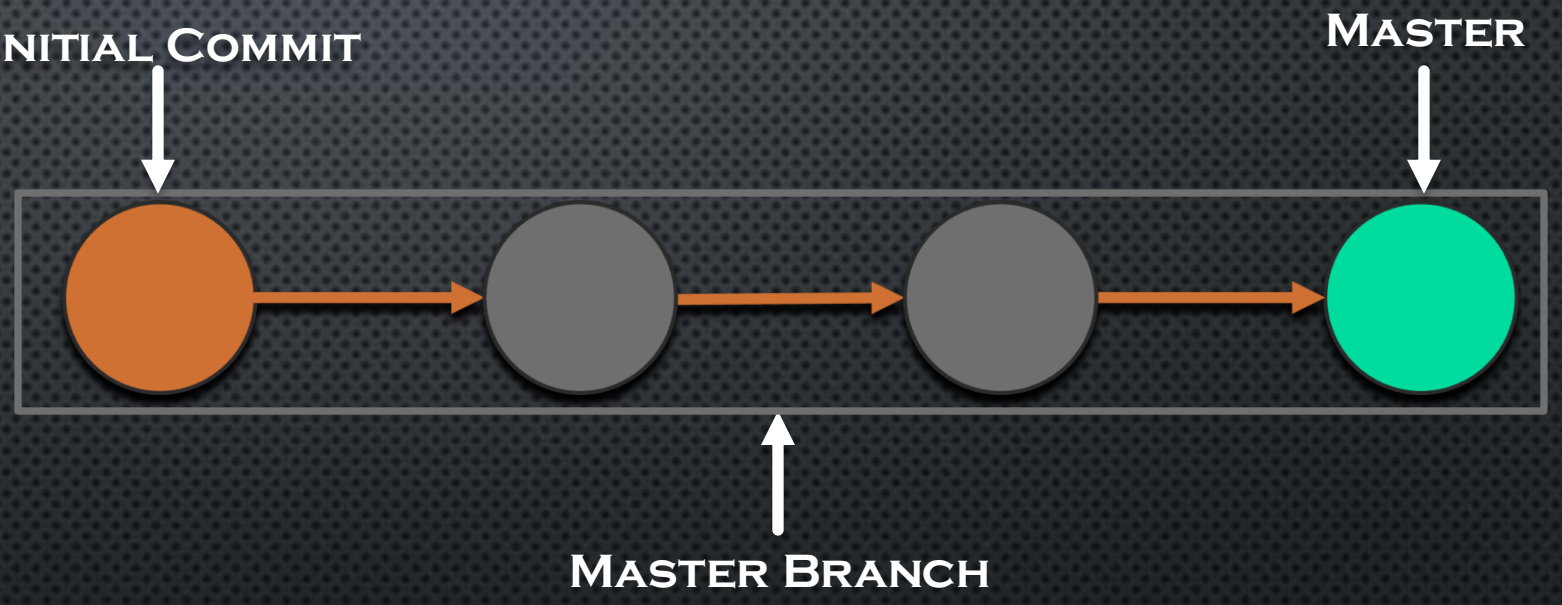
GIT BASICS – LINGO

- SAVEPOINTS CALLED "COMMITS"
  - "INITIAL COMMIT" TO START PROJECT
  - MOST RECENT STABLE VERSION TYPICALLY CALLED MASTER



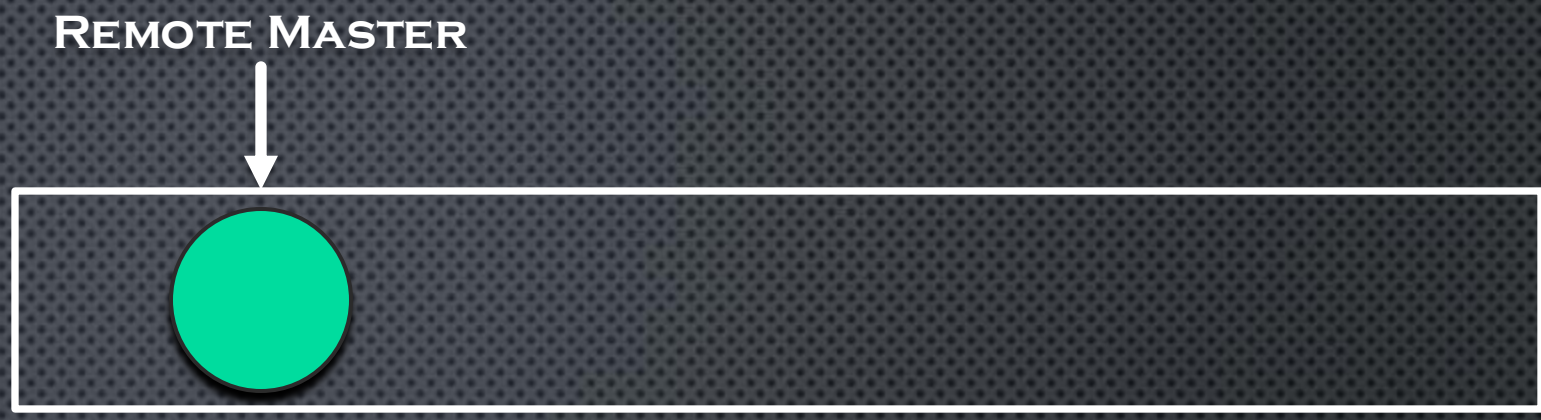
Git Basics – Lingo

- Savepoints called "Commits"
  - "Initial commit" to start project
  - Most recent stable version typically called Master
- Master Branch
  - Chain of commits from Initial → master commit



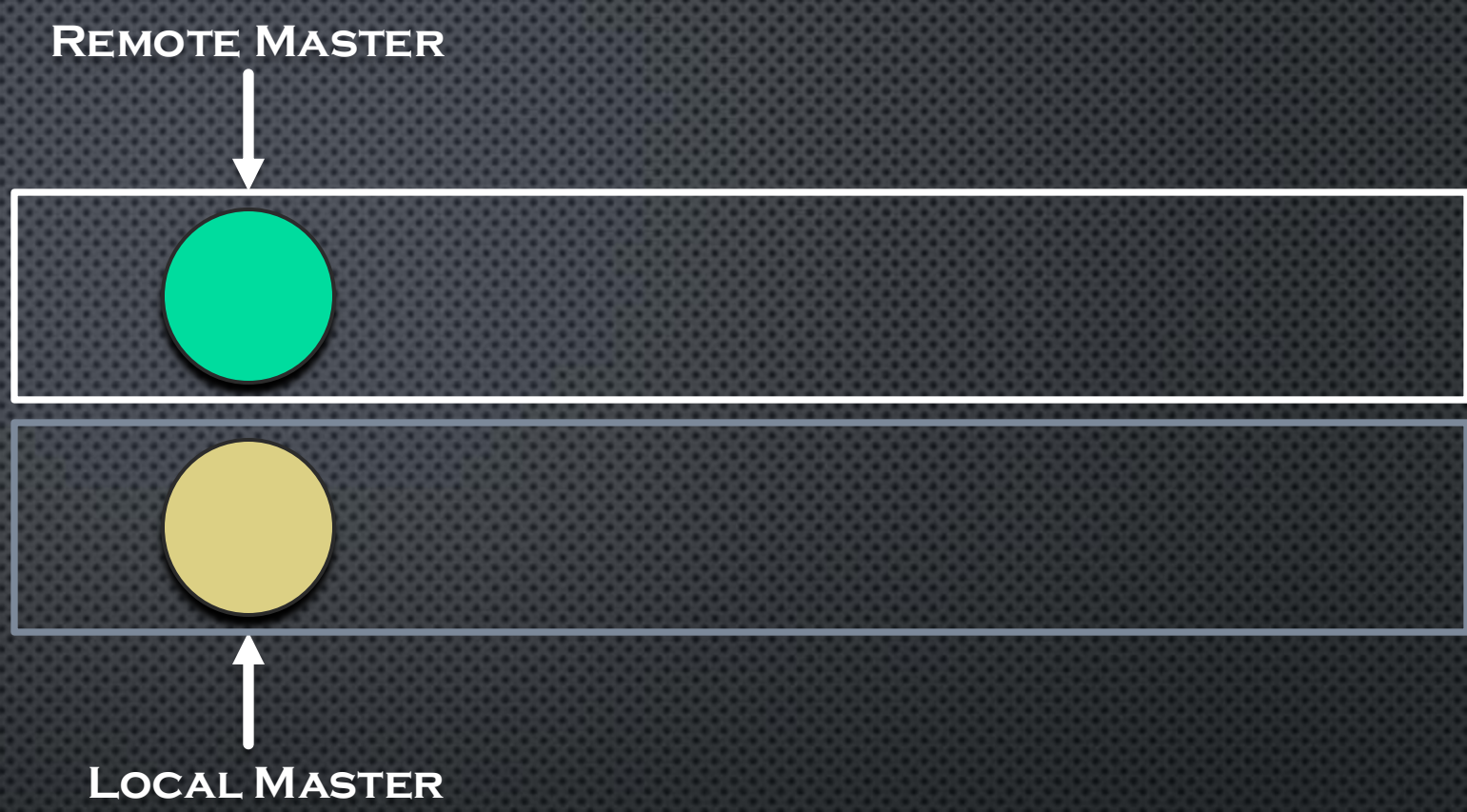
GIT BASICS – WORKING

- MASTER IS REPRESENTED IN GREEN
  - THIS IS THE REMOTE REPO- IT IS NOT ON YOUR COMPUTER



GIT BASICS – WORKING

- MASTER IS REPRESENTED IN GREEN
  - THIS IS THE REMOTE REPO- IT IS NOT ON YOUR COMPUTER
- HOW TO UPDATE REMOTE WITH LOCAL CHANGES?

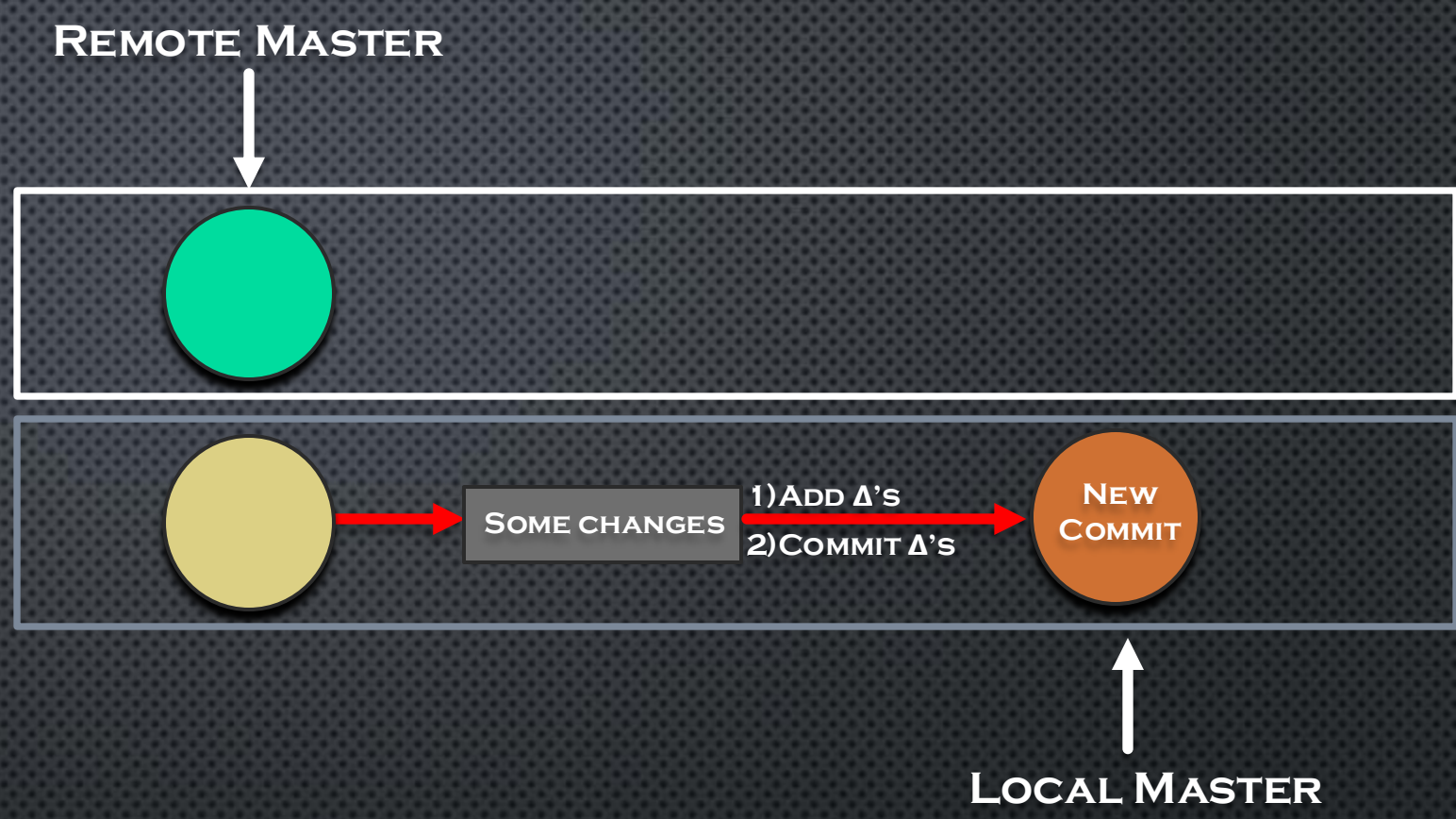


GIT BASICS – WORKING

- Master is represented in green
  - This is the remote repo- IT IS NOT ON YOUR COMPUTER
- How to update Remote with local changes?
  - Run `git status` → this shows you:
    - Branch
    - Commit
    - Changes to me committed

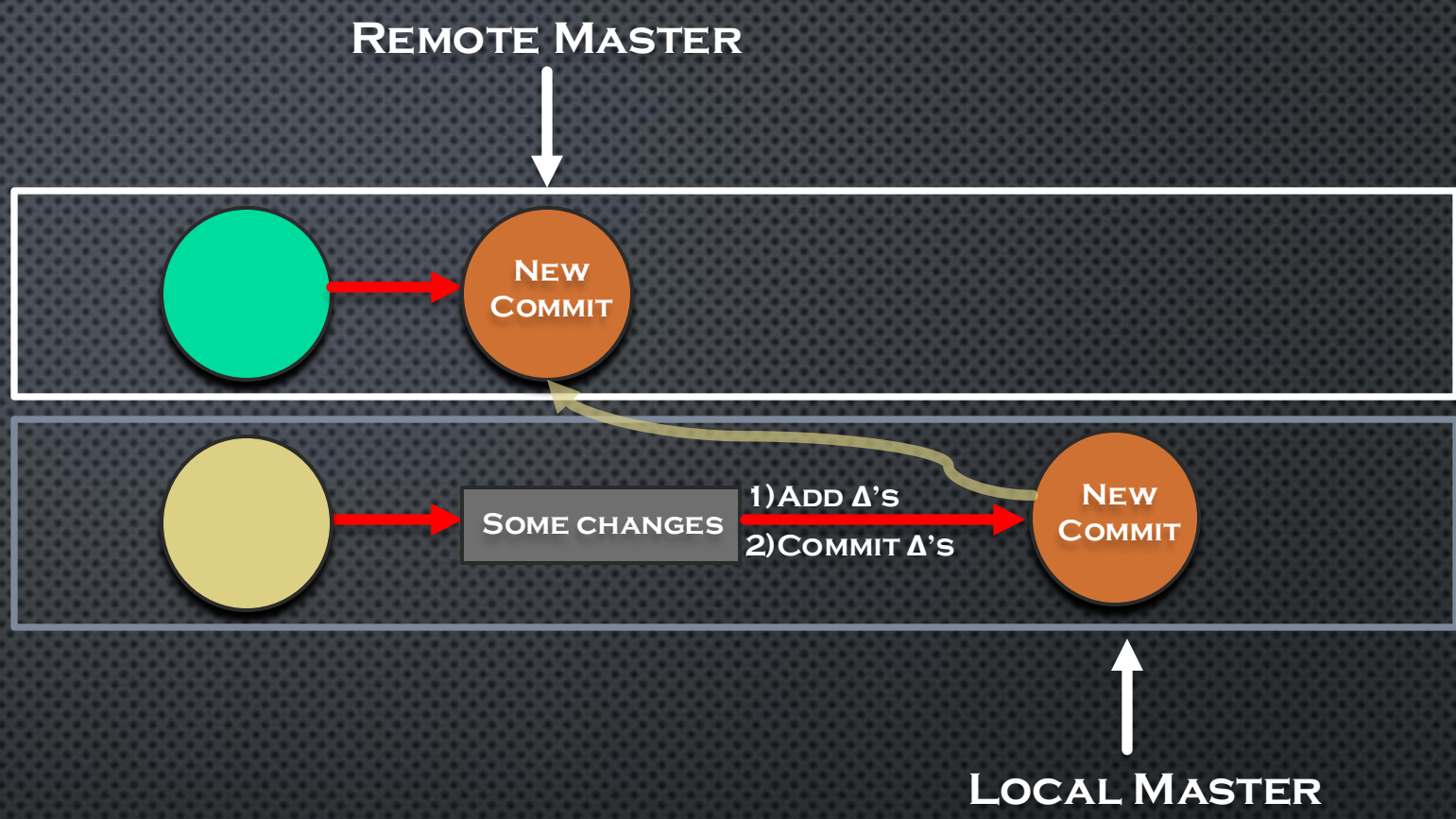
GIT BASICS – WORKING

- MASTER IS REPRESENTED IN GREEN
  - THIS IS THE REMOTE REPO- IT IS NOT ON YOUR COMPUTER
- HOW TO UPDATE REMOTE WITH LOCAL CHANGES?
  - RUN `git status` → THIS SHOWS YOU:
    - BRANCH
    - COMMIT
    - CHANGES TO ME COMMITTED
  - RUN ` git add ` → THIS WILL ADD FILES TO THE STAGING AREA I.E. CHANGES TO BE APPLIED
  - RUN `git commit –m “<some message>” ` THIS MAKES A COMMIT



GIT BASICS – WORKING

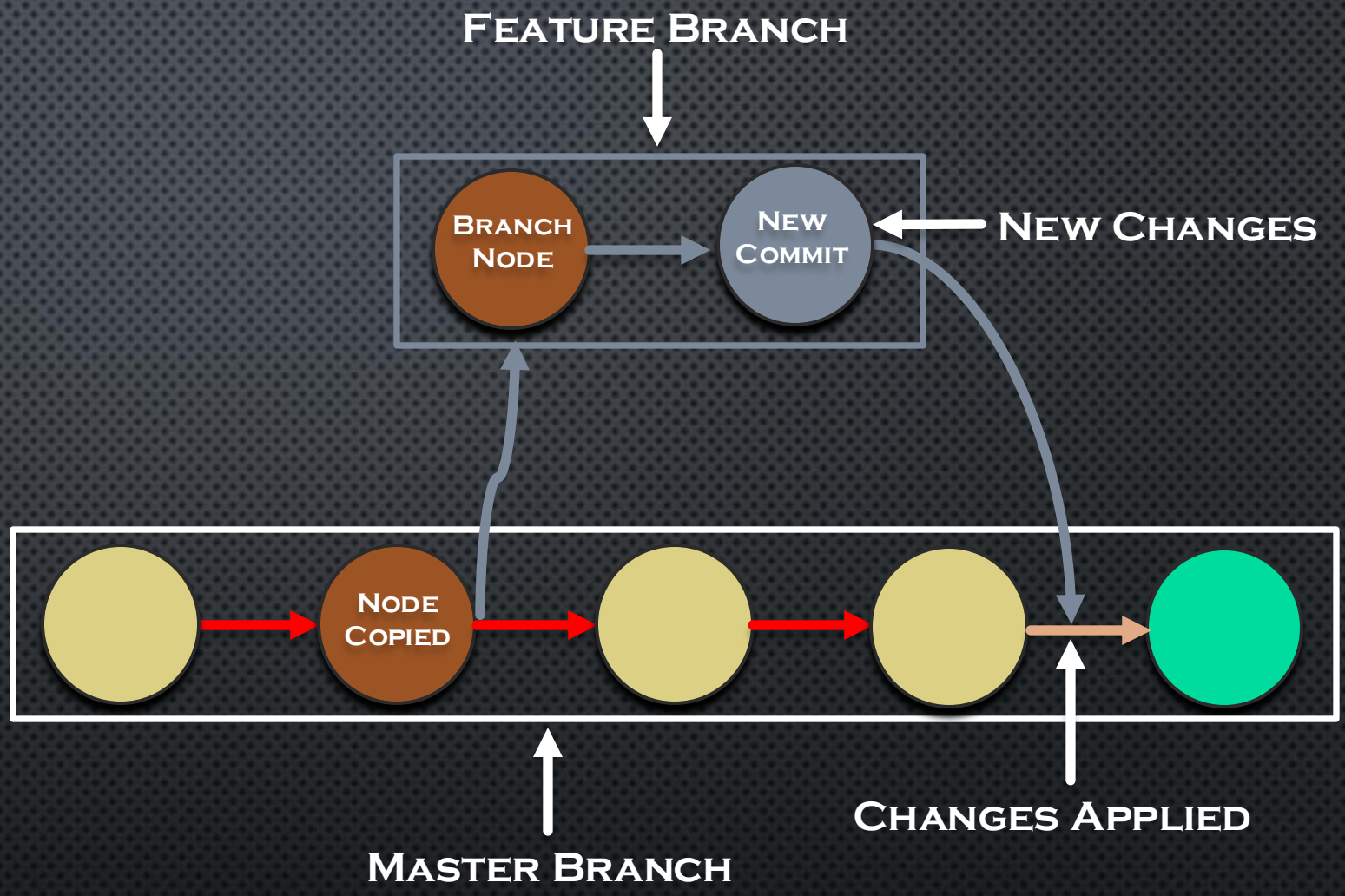
- MASTER IS REPRESENTED IN GREEN
  - THIS IS THE REMOTE REPO- IT IS NOT ON YOUR COMPUTER
- HOW TO UPDATE REMOTE WITH LOCAL CHANGES?
  - RUN `git status` → THIS SHOWS YOU:
    - BRANCH
    - COMMIT
    - CHANGES TO ME COMMITTED
  - RUN ` git add ` → THIS WILL ADD FILES TO THE STAGING AREA I.E. CHANGES TO BE APPLIED
  - RUN `git commit –m "<some message>" ` THIS MAKES A COMMIT
  - TO APPLY THIS TO THE REMOTE:
    - RUN ` git push` → THIS WILL PUSH YOUR CHANGES TO THE REMOTE



WORKING AS A GROUP

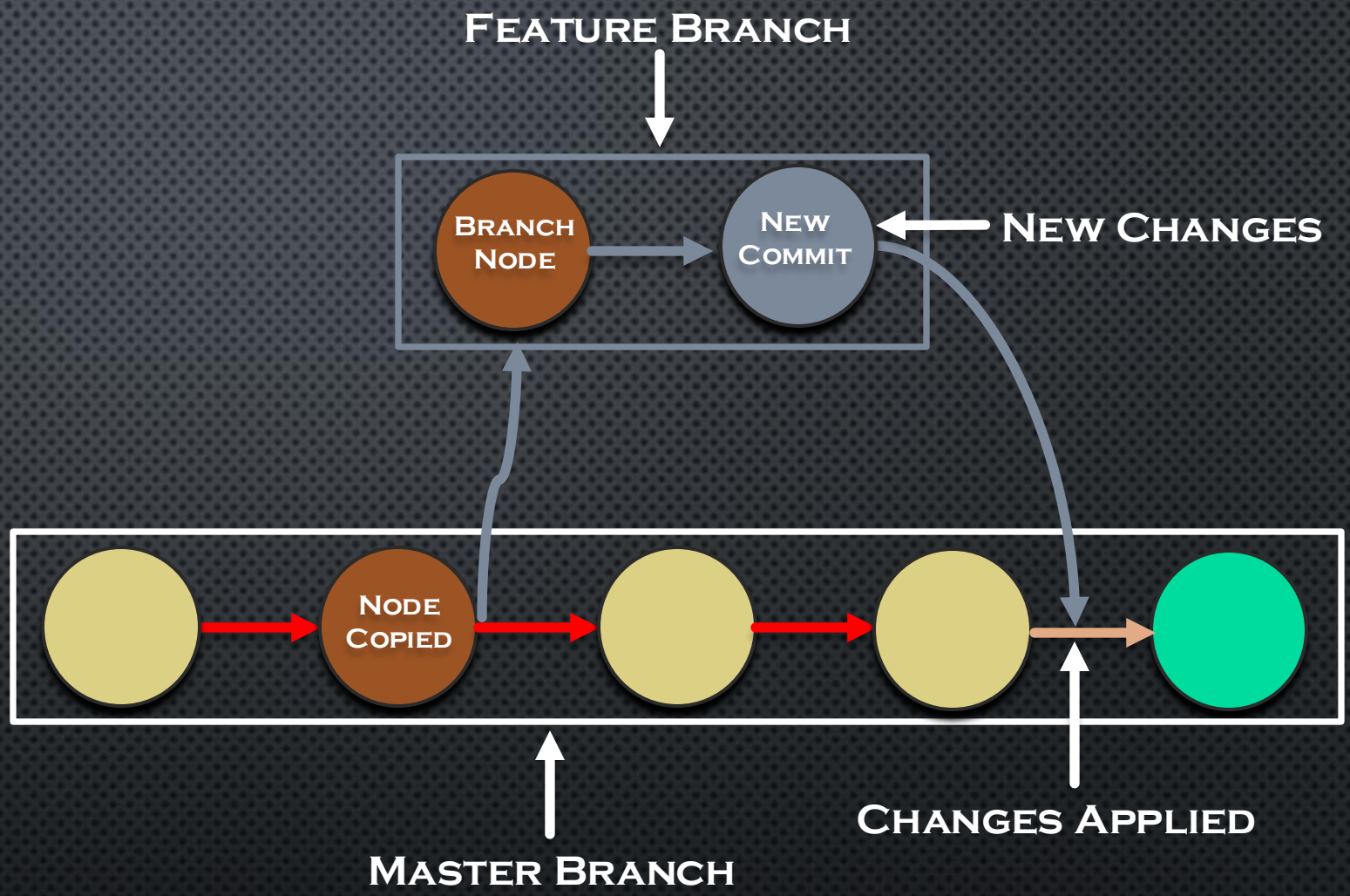
- THINGS GET HAIRY WHEN PUSHING TO A SINGLE BRANCH AS PART OF A TEAM
  - WHAT HAPPENS IF YOUR "FRIEND" PUSHES CHANGES RIGHT BEFORE YOU AND NOW NOTHING IN YOUR CODE WORKS.

- SOLUTION?
  - BRANCHES
  - FORKS(WE WONT DO THIS)



Branches- Working with Others

- Branches duplicate the timeline and allow you to work from a point in time independent of the Master
- Branches can be merged to the master
  - Warning: Before pushing branch changes, you must update said branch with changes to master
    - This is done via "git fetch <remote name>"
  - Next run "git merge"




BRANCH BASED COLLAB

STEP 1: CLONE THE REPO

Make Sure you are in the dir where you want the repo to land

If you name it something that already exists, bad things will happen

>>git clone <url>

>> cd <name or repo>

STEP 2: CREATE BRANCH

Check your current branch prior to creation

#Create

>>git checkout –b <name>

CHECK BRANCH AGAIN

STEP 3: MAKE CHANGE + COMMIT

Make your changes

>> git add .

>> git commit –m “<message>”

TAKE A SECOND AND THINK→ DO NOT PUSH

STEP 4: SYNC WITH MAIN DEV BRANCH

>> git fetch origin

>>git checkout <your branch>

MAKE SURE YOU ARE IN YOUR BRANCH

>>git merge origin/<branch>

#there may be conflicts to resolve

>>git commit

>> git push origin <your branch>

STEP 4: SYNC WITH MAIN DEV BRANCH

>> git fetch origin

>>git checkout <your branch>

MAKE SURE YOU ARE IN YOUR BRANCH

>>git merge origin/<branch>

#there may be conflicts to resolve

>>git commit

>> git push origin <your branch>

STEP 5: PULL REQUEST – UPDATE MAIN

Go to the repo in github.com

Click on Pull Requests→ new Pull Request

Use <your branch> as the source

Use main as target

If everything is good, then you can merge changes