The Git & Github Bootcamp

Section 1: Course Orientation

Section 2: Introducing …Git!

1.What really Matters in this Section

2.What Exactly is Git?

* Git is a version control system
* Version Control : Version control is software that

tracks and manages changes to files over time.

3.Visualizing Git

* Track changes across multiple files
* Compare versions of a project
* "Time travel" back to old versions
* Revert to a previous version
* Collaborate and share changes
* Combine changes

4.A quick History of Git

* Git development began in April 2005

5.Who uses Git?

* Engineers and coders.
* Tech Adjacent Roles.
* Governments.
* Scientists.
* Writers.

6.Git Vs. Github: What's The difference?

|  |  |
| --- | --- |
| Git | Github |
| Git is the version control software that runs locally on  your machine. You don't need to register for an account.  You don't need the internet to use it. You can use Git  without ever touching Github. | Github is a service that hosts Git repositories in the cloud  and makes it easier to collaborate with other people.  You do need to sign up for an account to use Github. It's  an online place to share work that is done using Git. |

Section 3: Installation & Setup

1.What really Matters in this section

2.Installing Git: Terminal Vs. GUIs

* Git is primarily a terminal tool
* It has GUI version too.

3.WINDOWS Git Installation

* Git bash : Bash is a command line interface that is widely used by

developers. It is the default shell for Linux and Mac. Git

"CIS designed to run on a Unix-based interface (like Bash)

4.MAC Git Installation

5.Configuring Your Git name and email

* Git config user.name - to know the username.
* Git config --global user.name "" - to set username as wrench

6.Installing GitKraken (Our GUI)

* Download git kraken from [www.gitkraken.com](http://www.gitkraken.com)

7.Terminal Crash Course: Introduction

* Basic linux commands
* Pwd
* ls
* Cd
* Rm
* Touch
* mkdir

8.Terminal Crash Course: Navigation

9.Terminal Crash Course: Creating Files and Folders

10.Terminal Crash Course: Deleting Files and folders

Section 4: The Very Basics of git: Adding & Committing

1.What really Matters in this section

2.What is a Git repo?

* A Git "Repo" is a workspace which tracks and manages files within a folder.

3.Our first Commands: git Init and Git status

* "git status" command gives information on the current status of a git repository and its contents
* Use "git init" to create a new repository. Before we can do anything git-related, we

must initialize a repo first!

4.The mystery .Git Folder

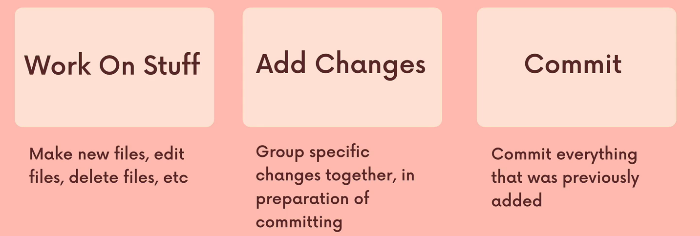
* .git folder is a hidden file containing useful files.

5.A Common Early Git Mistake

* Do not init a repo inside of a repo.
* Before running git init, use git status to verify that you are not currently inside of a repo.

6.The committing Workflow Overview





7.Staging Changes with Git Add

* We use the "git add" command to stage changes to be committed.

8.Finally, the git Commit Command

* We use the "git commit" command to actually commit changes from the staging area.
* Use "git commit -m "Message" command to provide a message while committing.

9.The Git Log Command (And More Committing)

* "git log" command provides the commit message on each commit.
* "git add ." command to stage all changes at once.

10.Committing Exercise