Version Control with Git

# Lesson 01: What is version control?

* Version Control System is just a software that helps you control (or manage) the different versions of something (typically the source code).
* There are number of version controls systems (<https://en.wikipedia.org/wiki/List_of_version-control_software>) but (03) are most popular:
  + Git [<https://git-scm.com/>]
  + Subversion [<https://subversion.apache.org/>]
  + Mercurial [<https://www.mercurial-scm.org/>]
* Two categories of version control systems:
  + Centralized [<https://www.atlassian.com/blog/software-teams/version-control-centralized-dvcs>]
    - There’s one all powerful central computer that hosts the project; and every interaction must go through this central computer.
  + Distributed [<https://en.wikipedia.org/wiki/Distributed_version_control>]
    - There’s no central repository of information; Each developer has a complete copy of the project on their own computer.
    - This is cool because you can also work offline.
* We will be using version control system “Git” which is distributed version control system.
* Git and GitHub are quite different.
  + Git is a version control tool while GitHub is the service that hosts Git projects.
  + To work with git projects, you’re not required to use GitHub.
* Abilities for powerful revision history:
  + The ability to label a change.
  + The ability to give a detailed explanation of why a change was made.
  + The ability to move between different versions of the same document.
  + The ability to undo change A, make edit B, then get back change A without affecting edit B.

## Terminology

* **Version Control System (VCS)**: It’s a tool that manages different versions of source code.
* **Source Code Manager (SCM)**: It’s another name for VCS.
  + Git is an SCM (and therefore a VCS!).
* **Commit**:
  + Git thinks of its data like a set of snapshots of a mini filesystem.
  + Every time you commit (save the state of your project in Git), it basically takes a picture of what all your files look like at that moment and stores a reference to that snapshot.
  + You can think of it as a save point in a game – it saves your project’s files and any information about them.
  + Everything you do in Git is to help you make commits, so ***commit is the fundamental unit in Git***.
* **Repository/Repo**: It’s a directory which contains your project work, as well as few files (hidden by default on Mac OS X) which are used to communicate with Git.
  + Repositories can exist either locally on your computer or as a remote copy on another computer.
  + A repository is made up of commits.
* **Working Directory**: It’s the files that you see in your computer’s file system.
* **Checkout**: It’s when content in the repository has been copied to the Working Directory.
* **Staging Area / Staging Index / Index**:
* **SHA**:
* **Branch**:

# Lesson 02: Create a Git Repo

# Lesson 03: Review a Repo’s History

# Lesson 04: Add commits to a Repo

# Lesson 05: Tagging, Branching, and Merging

# Lesson 06: Undoing Changes