Git Version Control

Reference:

<https://git-scm.com/docs>

<https://guides.github.com/>

<https://help.github.com/en>

# What is Git and Github?

In summary Github provides a way to track changes to a project, share code, and network with other developers.

Git is a version control system that supports collaboration; it is a system that records changes to a file or set of files over time.

Developers make constant changes to code, releasing updated versions and fixing bugs. Version control tracks updates and bug fixes and documents these modifications in a central repository. This allows us to store our updates, so if required we can ‘roll back’ to a previous version.

It also allows developers to easily collaborate and share code. They can download a project source code make changes, and upload their revision. Other developers can see these changes, download them, and make their own contributions.

# How to access Git

We mostly use a command line interface (Terminal on Mac) although there is also git software with a graphic interface (such as [Github Desktop](https://desktop.github.com/) or [Source Tree](https://www.sourcetreeapp.com/)). Git can also be used within some editors (such as Atom, Dreamweaver).

# 

# Repository

A repository (usually abbreviated to “repo”) is a location where all the files for a particular project are stored. Each project has its own repo, and a unique URL.

# Forking a Repo

“Forking” is when you create a new project based off of another project that already exists. This encourages the further development of open source projects initially created by other developers, and improved by a community of developers.

If you find a project you’d like to contribute to, you can fork the repo, make the changes you’d like, and release the revised project as a new repo. If the original repository that you forked to create your new project gets updated, you can easily add those updates to your own forked repo.

# Pull Requests

You’ve forked a repository, made a revision to the project and want to merge the project back into the original source (or want it to be recognized by the original developers). You can do so by creating a pull request. The authors of the original repository can see your work, and then choose whether or not to accept it into the original project. Whenever you issue a pull request you send a message to the project owner.

# Social networking

Github also provides social networking and an online community for developers. Each user on GitHub has their own profile (that acts like a CV), which showcases your work and contributions to other projects.

Many developers can contribute and collaborate to advance a project forward, and revisions can be discussed publicly.

# Changelogs

When multiple people collaborate on a project, it’s hard to keep track of revisions (who changed what, when, and where those files are stored). GitHub keeps track of all the changes in a change log.

# Branches

The ‘Master’ branch is usually the source of truth and other branches are usually used for new versions or development testing. Once a revision branch is completed it is then merged back into the master. It means you can continue to change code without affecting the original. This means if you create an error in your new branch you haven’t lost your original working source.

# Process

We start with a new project or clone (copy) an existing project. Then we add new files or alter existing files. We ‘Stage’ (add) these files to changes, then when changes are complete we ‘commit’ with a message to document what the changes where, then finally we’ll push the files upto the remote branch.

# Git setup

1. Create a Github account
2. Create a repo
3. [Download Git](https://git-scm.com/downloads)

# Working with Git

It’s common to use CLI (Terminal) commands when working with Git. Here are some common commands:

Check the git version:

$ git --version

Set up a new git on your local computer:

$ git init

Copy an existing Git repository hosted remotely:

$ git clone <https://www.github.com/username/repo-name>

Show the current Git directory’s remote repository

$ git remote

Check the state of the local repo

$ git status

List all branches:

$ git branch -a

Create a new branch:

$ git checkout -b <branch>

Make some new changes:

$ git add -a

Finalise local changes:

$ git commit -a -m "A message describing the change"

Push all changes to the remote develop branch:

$ git push origin branchname