Github Glossary

Basic Terminology

repository

This is easiest to imagine as a project's folder. A repository contains all of the project files (including documentation), and stores each file's revision history. Repositories can have multiple collaborators and can be either public or private.

remote

This is the version of a repository or branch that is hosted on a server, most likely GitHub.com or BitBucket.com.

clone

A clone is a copy of a repository that lives on your computer instead of on a website's server (like GitHub or BitBucket). In other words, when you clone a remote repository, you're downloading your own copy of the code onto your computer. With your copy of the code, you can edit the files in your preferred editor and use Git to keep track of your changes without having to be online. The repository you cloned is still connected to the remote version so that you can push your local changes to the remote to keep them synced when you're online.

branch

A branch is a parallel version of a repository. You can think of this in terms of an actual tree. The trunk (repository) and a branch (still part of the tree), but its own version of the repository. This branch allows the developer to work freely without disrupting the “live” version of the app (repository). Developers use “branches” to build and test new features or fix bugs. When finished writing and testing your code, this “branch” can be merged back into the repository.

main

The default development branch. Whenever you create a Git repository, a branch named “main” is created, and becomes the active branch. This branch usually contains the code that is “live” (website, app, game, software, etc). Using “main” as the default development branch name is purely a naming convention and is not required.

checkout

You can use “git checkout” on the command line to create a new branch or change your current working branch to a different branch.

commit
A commit, or "revision", is an individual change to a file (or set of files). When you make a commit to save your work, Git creates a unique ID (a.k.a. the "SHA" or "hash") that allows you to keep record of the specific changes committed along with who made them and when. Commits usually contain a commit message which is a brief description of what changes were made.

commit message
Short, descriptive text that accompanies a commit and communicates the change the commit is introducing.

merge
Merging takes the changes from one branch (in the same repository), and applies them into another. This often happens as a "pull request" (which can be thought of as a request to merge), or via the command line. A merge can be done through a pull request via the web interface (on Github, BitBucket, etc) if there are no conflicting changes, or can always be done via the command line.

merge conflict
A difference that occurs between merged branches. Merge conflicts happen when people make different changes to the same line of the same file, or when one person edits a file and another person deletes the same file. Git doesn’t know which change is correct, so it flags it as a “merge conflict” so the developers can decide whose change should be kept. The merge conflict must be resolved before you can merge the branches.

push
To push means to send your committed changes to a remote repository. For instance, if you change something locally, you can push those changes to the remote repository so that others may access them.

pull
Pull refers to when you are fetching changes and merging them in. For instance, if someone has edited the remote file you're both working on, you'll want to pull in those changes to your local copy of the code so that it's up to date.