Short Git and GitHub Glossary

Official Glossary

The short glossary below was condensed and modified from the glossary in the official GitHub docs. The full glossary is at github-glossary.

branch

A branch is a parallel version of a repository. It is contained within the repository, but does not affect the primary or main branch allowing you to work freely without disrupting the "live" version. When you've made the changes you want to make, you can merge your branch back into the main branch.

checkout

A "checkout" specifies a branch that you will subsequently work on. Anything you do after a checkout will effect only that branch, and not mess with the main branch. Once you are satisfied with your work on the branch, you can submit a "pull request" to have your changes pulled into the main branch. You can use `git checkout` on the command line to create a new branch or change your current working branch to a different branch.

clone

A clone is (n) a copy of a repository that lives on your computer instead of on a website's server somewhere, or (v) the act of making that copy. When you make a clone, 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.

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 considers your newly modified files to be the current "official" versions - the head of the snake. Commits should always 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.

default branch (aka base or main branch)

The base branch for new pull requests and code commits in a repository. Each repository has at least one branch, which git creates when you initialize the repository. The first branch is usually called "main" by default, and is the default or primary branch. In days of yore, it was called the "master" branch, you may still see this term used in some places.

diff

A diff is the difference in changes between two commits, or saved changes. The diff will visually describe what was added or removed from a file since its last commit.

fetch

When you use git fetch, you're adding changes from the remote repository to your local working branch without committing them. Unlike git pull, fetching allows you to review changes before committing them to your local branch. See "merge" below.

force push

A Git push that overwrites the remote repository with local changes without regard for conflicts. It is the analog of the `git pull` (which works in the opposite direction) in that it asks you no questions and tells you no lies.

fork

A fork is a personal copy of another user's repository that lives on your account. Forks allow you to freely make changes to a project without affecting the original or "upstream" repository. In general, forks are used when you want to use an existing project as a base, but intend to go your own way. For example, humans, chimps, and the other great apes are forks of a common primate ancestor - divergent species that share some traits but have forged their own path.

If your goal is to work on stuff that will soon be re-integrated into a main project, "branches" (see above) are the way to go.

main

The default development branch. Whenever you create a Git repository, a branch named main is created, and becomes the active branch.

merge

Merging takes the changes from

```
* the origin repo and applies them to your local repo
    - this is done via `git fetch` + `git merge`
* a branch in your repo to another branch (usually "main")
    - this happens after a pull request
```

Merging will give you an opportunity to resolve and conflicts between origin & local or between branches.

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. You need to resolve the conflicet before you can merge the branches.

origin

The default "upstream" repository. Most projects have at least one upstream project that they track. By default, origin is used for that purpose. *This will almost always be the GitHub copy of your repo.*

pull

Pull refers to when you are fetching in changes and merging them in one go. Use caution. See also fetch.

pull request

Pull requests are proposed changes to a repository submitted by a user and accepted or rejected by a repository's collaborators.

push

To push means to send your committed changes to a remote repository on GitHub.com. For instance, if you change something locally, you can push those changes so that others may access them. *Caution*, push don't play. It's safest to do a fetch & merge just before doing a push unless you're absolutely sure that no changes have been made to your remote repo beyond your ken.

README

A text file containing information about the files in a repository that is typically the first file a visitor to your repository will see. Make sure it doesn't suck.

remote (aka origin)

This is the version of a repository or branch that is hosted on a server, most likely GitHub.com. Remote versions can be connected to local clones so that

changes can be synced.

repository (aka repo)

A repository is the most basic element of GitHub. They're easiest to imagine as a project's folder. A repository contains all of the project files (including documentation). Within each repo is the .git folder, which is where the magic happens. Unless you are a level 7 git wizard or above, *do not duck with the .git folder.*

resolve

The action of fixing up manually what a failed automatic merge left behind.

review

Reviews allow others with access to your repository to comment on the changes proposed in pull requests, approve the changes, or request further changes before the pull request is merged.

staged

A new file is "staged" if it is ready to be included in the repo on the next commit. Done with `git add`.

status

The state of your current local repo. Get it using `git status`. This will show you

```
* if you are synched with origin
* what files have changed but are not staged
* what files are staged for the next commit
* any files in the repo directory that are not being tracked
```

`git status` is free to use - so use it!

upstream (aka origin or main)

When talking about a branch or a fork, the primary branch on the original repository is often referred to as "upstream", since that is where the content of the branch or fork came from. The branch/fork you are working on is then called "downstream".

When your changes are pulled from your branch to the main branch (and hence travel upstream), they are called "salmon".

Kidding. I just made that up. The bit about the salmon I mean. The rest of the doc is legit.