

Key Definitions

This cheat sheet contains git-specific terms and jargon. Here's a run-down of all the terms you may encounter:

Basic Definitions

- **Local repo or repository:** A local directory containing code and files for the project
- **Remote repository:** An online version of the local repository hosted on services like GitHub and GitLab
- **Cloning:** The act of making a clone or copy of a repository in a new director,
- **Commit:** A snapshot/save of the project you can come back to
- **Branch:** A copy of the project used for working in an isolated environment without affecting the main project
- **Git merge:** The process of combining two branches together

More advanced definitions

- **.gitignore file:** A file that lists other files you want git not to track (e.g. data folders, outputs, and any local files that shouldn't be seen by the public.
- **Staging area:** a cache that holds changes you want to commit next
- **Git stash:** another type of cache that holds unwanted changes you may want to come back later
- **HEAD:** a reference name for the latest commit. `HEAD~n` syntax is used to refer to older commits (e.g. `HEAD~2` refers to the second-to-last commit)

Starting a project

- | | |
|--|--|
| <code>git init [project name]</code> | Create a new local repository in the current directory names <code>[project name]</code> . |
| <code>git clone [project ssh url]</code> | Downloads a project with the entire history using the SSH URL from GitLab. |

Make a Change

- | | |
|-----------------------------|--|
| <code>git status</code> | Displays the status of your working directory. Options include new, staged, and modified files. It will retrieve branch name, current commit identifier, and changes pending commit. |
| <code>git add [file]</code> | Add a file to the staging area. Use. in place of the full file path to add all changed files from the current directory down into the directory tree. |
| <code>git add .</code> | Stage all files |

Make a Change (cont)

- | | |
|---|---|
| <code>git commit -m "commit message"</code> | Create a new commit from changes added to the staging area. The commit must have a message! |
|---|---|

Branches

- | | |
|--|--|
| <code>git branch</code> | List all local branches. Add <code>-r</code> flag for all remote branches. <code>a</code> flag for all branches. |
| <code>git branch new-branch</code> | Create a new branch called 'new-branch' |
| <code>git check out new-branch</code> | Switch to 'new-branch' branch & update the working directory |
| <code>git check out -b new-branch</code> | Create a new branch 'new-branch-' and switch to it |
| <code>git check out -D new-branch</code> | Delete 'new-branch' |

Merging

- | | |
|-----------------------------|---|
| <code>git checkout b</code> | Merge branch <code>a</code> into branch <code>b</code> . Add <code>--no-ff</code> |
| <code>git merge a</code> | option for no fast-forward merge |

Undoing Things

- | | |
|--|---|
| <code>git mv [existing_path] [new_path]</code> | Move (&/or rename) a file & stage move. |
| <code>git rm [file]</code> | Remove from staging area only. |
| <code>git checkout [commit_ID]</code> | View a previous commit (READ only) |
| <code>git revert [commit_ID]</code> | Create a new commit, reverting the changes from a specified commit |
| <code>git reset [commit_ID]</code> | Go back to a previous commit & delete all commits ahead of it (revert is safer). Add <code>--hard</code> flag to also delete workspace changes (BE CAREFUL) |

Share & Update

<code>git fetch</code>	Downloads commits and files from a remote repo into your local repo (no merge).
<code>git pull</code>	Downloads content from a remote repo and updates local repo to match content (git fetch + git merge).
<code>git push</code>	Upload local content to remote repo.

.gitignore Patterns

<code>*data/</code>	Ignore any directory named data .
<code>confidential.txt/</code>	Ignore file named confidential.txt .
<code>*.csv</code>	Ignore any file with the format .csv
<code>git config</code>	System wide ignore
<code>--global core.excludesfile [file]</code>	pattern for all local repos.

As best practice you should not upload any data or outputs (html, figs) to git. You can find some useful templates at GitHub [.gitignore Templates](#).

NB: You should not hold any sensitive data on GitLab as they may be accessed by others



By **hol**
cheatography.com/hol/

Not published yet.
 Last updated 23rd July, 2024.
 Page 2 of 2.

Sponsored by **ApolloPad.com**
 Everyone has a novel in them. Finish Yours!
<https://apollopad.com>