Cheatography

Git & GitHub Cheat Sheet

by Shaishav Trivedi via cheatography.com/192964/cs/40122/

Create a Repository

$ git init [project name]	The git init command is used to initialize a new Git repository in a directory. When you run this command in a folder, it sets up the necessary Git infrastructure, creating a hidden .git directory where Git stores its configuration files and history.
$ git clone git_url	Git will create a new directory with the name of the repository by default.
$ git clone git_url my_directory	The git clone command is used to create a copy of a Git repository from a remote source, such as a URL, into a local directory.

Configuration

$ git config --global user.name "name"	It is used to configure your Git username globally on your computer. This global configuration sets your Git username for all Git repositories on your system.
$ git config --global user.email "email"	It is used to configure your Git email address globally on your computer. This global configuration sets your Git email address for all Git repositories on your system.
$ git config --global color.ui auto	It is used to enable automatic colorization of Git's output in the terminal or command prompt. It enhances the readability of Git's commands and output by applying color to different elements like branch names, file statuses, and commit information.

Configuration (cont)

$ git config --global --edit	It opens the global Git configuration file in your default text editor, allowing you to edit it directly. This configuration file stores various settings that apply to all Git repositories on your system.

Working with Branches

$ git branch	It is used to list all the branches in your Git repository and show which branch you are currently on.
$ git branch -av	It is used to display a more detailed list of branches in your Git repository, including both local and remote branches. It provides information about the branches' names, commit SHAs, and the branches' relationship to remote repositories (if any).
$ git checkout my_branch	It is used to switch to a different branch in your Git repository.
$ git checkout -b new_branch	It is a convenient way to create and switch to a new branch in Git in a single step.
$ git branch -d my_branch	It is used to delete a local Git branch. However, it will only delete the branch if the changes on that branch have already been merged into the branch you are currently on. This is a safe way to clean up branches that are no longer needed.

Working with Branches (cont)

$ git checkout branchB $ git merge branchA	They are used to switch to "branchB" and then merge changes from "branchA" into "branchB".
$ git tag my_tag	It is used to create a lightweight or annotated tag in Git, which allows you to mark a specific commit with a label, making it easier to reference or identify later.

Synchronize

$ git fetch [alias]	It is used to fetch changes from a remote repository using a specific remote alias. In Git, a remote alias typically represents a remote repository, such as a repository on GitHub or another server.
$ git merge [alias]/[branch]	It merges the specified remote branch into your current branch. It performs a regular merge, which may create a new merge commit if there have been changes in both branches. This is the standard way to merge changes from a remote branch into your current branch.
$ git merge --no-ff [alias]/[branch]	Adding the --no-ff (no fast-forward) flag to the merge command ensures that a merge commit is always created, even if it could be fast-forwarded. This can be useful to preserve a clear history of feature branches and indicate when a branch was merged.

Synchronize (cont)

$ git merge --ff-only [alias]/[branch]	Using the --ff-only (only fast-forward) flag instructs Git to perform a fast-forward merge if possible. A fast-forward merge occurs when the branch you're merging into has no new commits since the branch you're merging from. If a fast-forward merge isn't possible (i.e., there are new commits in the current branch), Git will not perform the merge.
$ git push [alias] [branch]	It is used to push your local branch to a remote repository represented by the specified remote alias. This command is commonly used to share your local changes with others or to update the remote branch with your local commits.
$ git pull	It is used to fetch changes from a remote repository (commonly referred to as "fetching") and then automatically merge those changes into your current branch.
$ git cherry-pick [commit_id]	It is used to apply a specific commit's changes to your current branch. It allows you to select and pick individual commits from one branch and apply them to another branch. This can be useful for incorporating specific changes or fixes from one branch into another.

Rename branch

$ git branch -m <new_name>	It is used to rename the current branch in a Git repository to a new name.
$ git push origin -u <new_name>	It is used to rename a branch and push it to the remote repository with a new name.
$ git push origin --delete <old>	It is used to delete a branch from a remote Git repository.

Rewriting history

$ git commit --amend -m "new message"	It is used to amend (modify) the most recent Git commit with a new commit message. This is useful when you want to change the commit message of the last commit you made.

Make a change

$ git status	It is used to show the current status of your Git working directory.
$ git add [file]	It is used to stage changes in a file or files for the next commit. It tells Git to include the specified file(s) in the commit.
$ git add .	It stages all changes in the current directory and its subdirectories for the next commit. This means it includes any modifications, additions, or deletions of files in the working directory in the staging area.
$ git commit -m "commit message"	It is used to save the staged changes in your Git repository with a descriptive commit message. This message helps explain the purpose of the commit.

Make a change (cont)

$ git commit -am "commit message"	It is a convenient way to commit changes in Git. It combines two steps: staging changes (git add .) and committing them with a message (git commit -m "-commit message").
$ git restore [file]	It is used to undo changes in your working directory or unstage changes from the staging area in Git.
$ git restore --staged [file]	It is used to unstage specific file from the staging area.
$ git reset [file]	It is used to unstage changes for a specific file in the Git staging area, effectively removing it from the staging area
$ git diff	It is used to view the differences (changes) between the current state of your working directory and the last committed version in your Git repository. It provides a line-by-line comparison of changes in your files.
$ git diff --staged	It is used to view the differences (changes) between the last commit and the files currently staged in the Git staging area. This command shows you what changes you are about to commit.
$ git rebase [branch]	It is used to reapply the commits from your current branch on top of another branch. It effectively moves your branch's starting point to the tip of the specified branch, incorporating all the changes from the target branch.

Cheatography

Git & GitHub Cheat Sheet

by Shaishav Trivedi via cheatography.com/192964/cs/40122/

Observe your Repository

$ git log	It is used to display a log of the commit history in your Git repository. When you run this command, Git will show a list of commits in reverse chronological order, with the most recent commits appearing first.
$ git log branchB..branchA	It is used to view the commit history that exists in "branchA" but not in "branchB". In other words, it shows the commits that are unique to "branchA" when compared to "branchB".
$ git log --follow [file]	It is used to view the commit history of a specific file while also following the file's history when it has been renamed or moved. This is particularly useful when you want to see the complete history of a file, even if it has been renamed at some point.
$ git diff branchB...branchA	It is used to compare the differences between two branches, "branchB" and "branchA", in Git. It specifically shows the changes that have occurred on each branch compared to their common ancestor.
$ git show [SHA]	It is used to display detailed information about a specific commit in your Git repository.

Remote

$ git remote add [alias] [url]	It is used in Git to add a new remote repository as an alias with a specified URL.
$ git remote	It lists the names of all the remote repositories that are associated with your local Git repository.

Remote (cont)

$ git remote -v	It is used to list the remote repositories associated with your local Git repository along with their corresponding URLs. The '-v' flag stands for "verbose," and it displays both the remote's alias (nickname) and the URL of the remote repository.
$ git remote rm [remote repo name]	It is used to remove a remote repository from the list of remote repositories associated with your local Git repository.
$ git remote set-url origin [git_url]	It is used to change the URL associated with an existing remote repository.

Temporary Commits

$ git stash	It is used in Git to temporarily save changes that you've made to your working directory but don't want to commit immediately.
$ git stash list	It is used to display a list of stashes that you have saved in your Git repository.
$ git stash pop	It is used to apply the most recent stash from the stash list and remove it from the stash list.
$ git stash drop	It is used to remove a specific stash from the stash list.

Tracking path Changes

$ git rm [file]	It is used in Git to remove a file from the current working directory and the staging area (index), effectively deleting it from the Git repository.

Tracking path Changes (cont)

$ git mv [existing-path] [new-path]	It is used to rename or move a file or directory within a Git repository.
$ git log --stat -M	It is used to display the commit history of a Git repository with some additional statistics about the changes in each commit, including information about moved or renamed files.

Log

$ git log -S'<a term in the source>'	It is used to search the commit history for changes that introduced or removed a specific string (or term) in the source code. This can be useful for tracking when a particular piece of code was added or removed from the codebase.
$ git log -p <file_name>	It is used to view the commit history of a specific file and see the changes (diffs) made to that file in each commit.
$ git log --pretty=oneline --graph --decorate --all	It is used to display a concise and visually informative representation of the commit history in a Git repository.