Git Cheat Sheet

Git Basics

`git init <directory>`	Create empty Git repo in specified directory. Run with no arguments to initialize the current directory as a git repository.
`git clone <repo>`	Clone repo located at <repo> onto local machine. Original repo can be located on the local filesystem or on a remote machine via HTTP or SSH.
`git config user.name <name>`	Define author name to be used for all commits in current repo. Devs commonly use --global flag to set config options for current user. +
`git add <directory>`	Stage all changes in <directory> for the next commit. Replace <directory> with a <file> to change a specific file.
`git commit -m "<message>"`	Commit the staged snapshot, but instead of launching a text editor, use <message> as the commit message.
`git status`	List which files are staged, unstaged, and untracked.
`git log`	Display the entire commit history using the default format. For customization see additional options. +
`git diff`	Show unstaged changes between your index and working directory +

Undoing Changes

`git revert <commit>`	Create new commit that undoes all of the changes made in <commit>, then apply it to the current branch.
`git reset <file>`	Remove <file> from the staging area, but leave the working directory unchanged. This unstages a file without overwriting any changes. +
`git clean -n`	Shows which files would be removed from working directory. Use the -f flag in place of the -n flag to execute the clean.

Rewriting Git History

`git commit --amend`	Replace the last commit with the staged changes and last commit combined. Use with nothing staged to edit the last commit's message.
`git rebase <base>`	Rebase the current branch onto <base>. <base> can be a commit ID, a branch name, a tag, or a relative reference to HEAD. +
`git reflog`	Show a log of changes to the local repository's HEAD. Add --relative-date flag to show date info or --all to show all refs.

Git Branches

`git branch`	List all of the branches in your repo. Add a <branch> argument to create a new branch with the name <branch>.
`git checkout -b <branch>`	Create and check out a new branch named <branch>. Drop the -b flag to checkout an existing branch.
`git merge <branch>`	Merge <branch> into the current branch.

Remote Repositories

`git remote add <name> <url>`	Create a new connection to a remote repo. After adding a remote, you can use <name> as a shortcut for <url> in other commands.
`git fetch <remote> <branch>`	Fetches a specific <branch>, from the repo. Leave off <branch> to fetch all remote refs.
`git pull <remote>`	Fetch the specified remote's copy of current branch and immediately merge it into the local copy. +
`git push <remote> <branch>`	Push the branch to <remote>, along with necessary commits and objects. Creates named branch in the remote repo if it doesn't exist. +