1. Create a new branch:  
   git checkout -b feature\_branch\_name
2. Edit, add and commit your files.
3. Push your branch to the remote repository:  
   git push -u origin feature\_branch\_name

*It’s as simple as that!*

**What’s going on here? Git Branch explained in more detail**

**Git Branch**

Git’s branching functionality lets you create new branches of a project to test ideas, isolate new features, or experiment without impacting the main project.

**View Branches**

To view the branches in a Git repository, run the command:

git branch

To view both remote-tracking branches and local branches, run the command:

git branch -a

There will be an asterisk (\*) next to the branch that you’re currently on.

There are a number of different options you can include with git branch to see different information. For more details about the branches, you can use the -v (or -vv , or --verbose ) option. The list of branches will include the SHA-1 value and commit subject line for the HEAD of each branch next to its name.

You can use the -a (or --all ) option to show the local branches as well as any remote branches for a repository. If you only want to see the remote branches, use the -r (or --remotes ) option.

**Checkout a Branch**

To checkout an existing branch, run the command:

git checkout BRANCH-NAME

Generally, Git won’t let you checkout another branch unless your working directory is clean, because you would lose any working directory changes that aren’t committed. You have three options to handle your changes:

1. trash them (see [Git checkout for details 137](https://guide.freecodecamp.org/git/git-checkout/)) or
2. commit them (see [Git commit for details 258](https://guide.freecodecamp.org/git/git-commit/)) or
3. stash them (see [Git stash for details 125](https://guide.freecodecamp.org/git/git-stash/)).

**Create a New Branch**

To create a new branch, run the command:

git branch NEW-BRANCH-NAME

Note that this command only creates the new branch. You’ll need to run git checkout NEW-BRANCH-NAME to switch to it.

There’s a shortcut to create and checkout a new branch at once. You can pass the -b option (for branch) with git checkout . The following commands do the same thing:

*# Two-step method*

git branch **NEW**-BRANCH-NAME

git checkout **NEW**-BRANCH-NAME

*# Shortcut*

git checkout -b **NEW**-BRANCH-NAME

When you create a new branch, it will include all commits from the parent branch. The parent branch is the branch you’re on when you create the new branch.

**Rename a Branch**

To rename a branch, run the command:

git branch -m OLD-BRANCH-NAME **NEW**-BRANCH-NAME

*# Alternative*

git branch --move OLD-BRANCH-NAME **NEW**-BRANCH-NAME

**Delete a Branch**

Git won’t let you delete a branch that you’re currently on. You first need to checkout a different branch, then run the command:

git branch -d BRANCH-TO-**DELETE**

*# Alternative:*

git branch *--delete BRANCH-TO-DELETE*

The branch that you switch to makes a difference. Git will throw an error if the changes in the branch you’re trying to delete are not fully merged into the current branch. You can override this and force Git to delete the branch with the -D option (note the capital letter) or using the --force option with -d or --delete :

git branch -D BRANCH-TO-**DELETE**

*# Alternatives*

git branch -d *--force BRANCH-TO-DELETE*

git branch *--delete --force BRANCH-TO-DELETE*

**Compare Branches**

You can compare branches with the git diff command:

git diff FIRST-BRANCH..SECOND-BRANCH

You’ll see colored output for the changes between branches. For all lines that have changed, the SECOND-BRANCH version will be a green line starting with a “+”, and the FIRST-BRANCH version will be a red line starting with a “-”. If you don’t want Git to display two lines for each change, you can use the --color-words option. Instead, Git will show one line with deleted text in red, and added text in green.

If you want to see a list of all the branches that are completely merged into your current branch (in other words, your current branch includes all the changes of the other branches that are listed), run the command git branch --merged .

**Update a Branch from Remote**

**To update a local branch from remote:**

git stash (optional, to save local changes which differs from the remote repository **if** any)

**If you weren’t already on the branch you want to work on:**

git checkout my\_local\_branch

**Finally pull from the remote branch**

git pull

**Track a Remote Branch**

If you already have a branch and you want to track a remote branch, then you use set-upstream-to command:

git branch --set-upstream-to origin/BRANCH

Or you can use the -u flag (upstream) when you make your first push:

git push -u origin BRANCH

**Help with Git Branch**

If you forget how to use an option, or want to explore other functionality around the git branch command, you can run any of these commands:

git help branch

git branch --help

man git-branch