

# 12 Days of Git

DAY 7  
GIT BRANCHES

# What are Git branches?

Branches are an everyday part of the process when using Git. Effectively they are a point to your changes. You might create a new branch when you want to work on a new feature, or bug fix and keep those changes completely separate until you are ready to release them.

There are a few commands that you can use to work with branches, let's take a look at them.

# Create a new branch

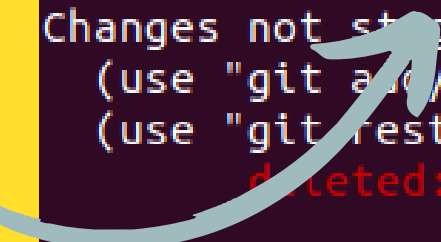

If you have a copy of a repository and you want to make changes then creating a new branch is best practice.

To create a new branch you can run the command:

```
(base) → PyPi-connectdb git:(main) X git checkout -b update-query-structure  
Switched to a new branch 'update-query-structure'  
(base) → PyPi-connectdb git:(update-query-structure) X █
```

# See what branch you are on

When you are working within a repository, and you want to check what branch you are on you can use the following command:



```
(base) → PyPi-connectdb git:(update-query-structure) X git status
On branch update-query-structure
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    update.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        connection/

no changes added to commit (use "git add" and/or "git commit -a")
(base) → PyPi-connectdb git:(update-query-structure) X
```

Its also visible in terminal base line

# Switch to a local branch

If you have a branch within your local repository that you've created then you can easily switch to it using the following command:

```
(base) → PyPi-connectdb git:(update-query-structure) X git checkout main
D      update.py
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)
(base) → PyPi-connectdb git:(main) X
```

# Switch to a branch that's came from a remote repo

If you have cloned a Git repository from a remote location and it's come with a bunch of branches, then you can use the following command:

```
git checkout --track origin/main
```

# Push a branch

When you create a branch on your local copy of the repository it won't automatically create within the remote location. When you come to push your changes to that remote location you can't just use the `git push` command you need to use a slightly different command, either of these:

Using **branch name**

```
git push -u origin branchname
```

Using **HEAD**

Referencing HEAD saves you from having to type out the exact name of the branch.

```
git push -u origin HEAD
```

*Of course, if your branch already exists in the remote location, you can just run `git push`.*

# Merge a branch

So, you've created a branch, and you are ready to merge that branch into the main one, ready for production or the next step in your development phrase. How can you do it?

The first thing you need to do is switch to the branch you want to merge your changes into. In this example I want to merge my changes from the branch "update-query-structure" into my main branch.

```
(base) → PyPi-connectdb git:(update-query-structure) X git checkout main
D      update.py
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)
(base) → PyPi-connectdb git:(main) X git merge update-query-structure
```

*When you issue this command you may receive merge conflicts, we will be looking at how to deal with them in coming days.*

*I now have to push this merge from my local repository to my remote repository, I do that with the command:*

```
PyPi-connectdb git:(main) X git push
```

# Delete a local branch

If you want to delete a branch that has already been merged you can use the command:

**git branch -d branchname**

If you want to delete a local branch regardless of whether it has been merged or not, then the command to use is:

**git branch -D branchname**

There is a subtle difference, the capital **D** is really a shortcut for **--delete --force**.



For more visit:  
<https://www.techielass.com/git-branches/>

