

 **static** (ডিফল্ট)

 **relative**

 **absolute**

 **fixed**

 **sticky**

**সংক্ষেপে:**

* **static**: ডিফল্ট, কোনও পজিশনিং প্রভাব নেই।
* **relative**: তার সাধারণ অবস্থানের তুলনায় সরানো হয়।
* **absolute**: নিকটস্থ পজি শনড পূর্বপুরুষের তুলনায় সরানো হয়।
* **fixed**: ভিউপোর্টের তুলনায় সরানো হয় এবং স্থির থাকে।
* **sticky**: স্ক্রোলিং এর উপর ভিত্তি করে relative এবং fixed এর মধ্যে পরিবর্তিত হয়।

<https://www.nobledesktop.com/learn/git/git-branches>

Learn how Git branches can be useful for various website features and updates, with detailed command instructions provided for different operational aspects. This content outlines how to create and manage Git branches, switch between branches, merge branches, and delete branches with hands-on examples.

## Key Insights

* Git allows for the creation of new branches for separate tasks, enabling users to switch between different tasks easily with commands such as 'git status' and 'git branch'.
* Users can create a new branch in Git by running the 'git checkout -b my-branch-name' command, where 'my-branch-name' can be replaced with the desired name.
* To switch between local branches, the 'git checkout my-branch-name' command is used. For branches from a remote repository, the 'git checkout --track origin/my-branch-name' command is utilized.
* Pushing to a branch that does not exist on the remote can be done using 'git push -u origin my-branch-name' or 'git push -u origin HEAD', where HEAD refers to the top of the current branch. If the branch already exists on the remote, the 'git push' command can be used.
* Merging branches involves ensuring the working tree is clean, checking out the branch to merge into another and then running the 'git merge my-branch-name' command.
* Branches can be deleted using 'git push origin --delete my-branch-name' for remote branches and 'git branch -d my-branch-name' or 'git branch -D my-branch-name' for local branches. The '-D' option forcefully deletes the branch regardless of its merged status.

Git lets you branch out from the original code base. This lets you more easily work with other developers, and gives you a lot of flexibility in your workflow.



Here's an example of how Git branches are useful. Let's say you need to work on a new feature for a website. You create a new branch and start working. You haven't finished your new feature, but you get a request to make a rush change that needs to go live on the site today. You switch back to the master branch, make the change, and push it live. Then you can switch back to your new feature branch and finish your work. When you're done, you merge the new feature branch into the master branch, and both the new feature and rush change are kept!

## For All the Commands Below

The commands below assume you've navigated to the folder for the Git repo.

### See What Branch You're On

* Run this command:
  + **git status**

### List All Branches

**NOTE:**The current local branch will be marked with an asterisk (\*).

* To see **local branches**, run this command:
  + **git branch**
* To see **remote branches**, run this command:
  + **git branch -r**
* To see **all local and remote branches**, run this command:
  + **git branch -a**

### Create a New Branch

* Run this command (replacing **my-branch-name**with whatever name you want):
  + **git checkout -b my-branch-name**
* You're now ready to commit to this branch.

### Switch to a Branch In Your Local Repo

* Run this command:
  + **git checkout my-branch-name**

### Switch to a Branch That Came From a Remote Repo

1. To get a list of all branches from the remote, run this command:
   * **git pull**
2. Run this command to switch to the branch:
   * **git checkout --track origin/my-branch-name**

### Push to a Branch

* If your local branch **does not exist**on the remote, run either of these commands:
  + **git push -u origin my-branch-name**
  + **git push -u origin HEAD**

**NOTE:**HEAD is a reference to the top of the current branch, so it's an easy way to push to a branch of the same name on the remote. This saves you from having to type out the exact name of the branch!

* If your local branch **already exists**on the remote, run this command:
  + **git push**

### Merge a Branch

1. You'll want to make sure your working tree is clean and see what branch you're on. Run this command:
   * **git status**
2. First, you must check out the branch that you want to merge another branch into (changes will be merged into this branch). If you're not already on the desired branch, run this command:
   * **git checkout master**
   * **NOTE:** Replace **master**with another branch name as needed.
3. Now you can merge another branch into the current branch. Run this command:
   * **git merge my-branch-name**
   * **NOTE:** When you merge, there may be a conflict. Refer to **Handling Merge Conflicts**(the next exercise) to learn what to do.

### Delete Branches

* To delete a **remote branch**, run this command:
  + **git push origin --delete my-branch-name**
* To delete a **local branch**, run either of these commands:
  + **git branch -d my-branch-name**
  + **git branch -D my-branch-name**
* **NOTE:** The -d option only deletes the branch if it has already been merged. The -D option is a shortcut for --delete --force, which deletes the branch irrespective of its merged status.