What is GIT?

Git is a DevOps tool used for source code management. It is one of the most prominent version control systems (VCS) today that is widely used to handle small and large projects efficiently. It helps in tracking changes in source code, enabling different people to collaborate on different parts of the same program.

Before diving into our how Git works tutorial, let's understand a little more about what Git is in general.

**What Is Git?**

Git is a version control system for tracking changes in computer files. It helps in coordinating work amongst several people in a project and tracks progress over time. Unlike the centralized version control system, Git branches can be easily merged. A new branch is created every time a developer wants to start working on something. This ensures that the master branch always has a production-quality code.

Git is a distributed version control system, so here, every developer gets their local repository with full commit history. The commit history makes Git fast, as now a network connection is not needed to create commits or perform diffs between commits.

Now that we know more about Git let us explore GitHub.

What Is GitHub?

GitHub is a Git repository hosting service that provides a web-based graphical interface (GUI). It helps every team member work together on a project from anywhere, making it easy to collaborate.

GitHub is one place where project managers and developers coordinate, track, and update their work, so projects stay transparent and on schedule. The packages can be published privately, within the team, or publicly for the open-source community. Downloading packages from GitHub enables them to be used and reused. GitHub helps all team members stay on the same page and stay organized. Moderation tools, like issue and pull request locking, helps the team focus on the code.

Post Graduate Program in DevOps

Designed in collaboration with Caltech CTMEEXPLORE COURSEPost Graduate Program in DevOps

Next in this how Git works tutorial, let's look at the several commands used in Git.

Different Commands in Git

So far, in the how Git works tutorial you read all about Git and GitHub. Next in the how Git works tutorial comes Git commands.

Git config

Git init

Git add

Git diff

Git commit

Git reset

Git status

Git merge

Git push

Git pull

Next in this how Git works tutorial, we will go over the Git push command in detail.

Git Push Command

The Git push command is used to push the local repository content to a remote repository. After a local repository has been modified, a push is executed to share the modifications with remote team members. Pushing is the way commits are transferred from the local repository to the remote repository.

Git-push-command

Now that we have learned about the Git push command in this Git Bash tutorial let us take a look at a Git push command demo.

FREE DevOps Certification Training

Master the fundamnetals of DevOpsENROL NOWFREE DevOps Certification Training

Demo of Git Push Command

Next in the how Git works tutorial, let’s start with opening Git Bash and configuring it with a user name and email ID. To configure, we use the following commands:

To configure, we use the following commands:

Git config --global user.name "Simplilearn GitHub"

Git config --global user.email siddam.bharat@simplilearn.net

Git config –list

working-push-command

Then, let's check the current working directory:

pwd

taha

To create a repository in the working directory, use the following commands:

mkdir Git\_Demo

cd Git\_Demo

pwd

We can go to the directory location and check the Git\_demo folder.

taha

The directory "Git\_demo" will be empty for now.

Let's create a folder for the repository.

mkdir FirstRepo

cd FirstRepo

pwd

first repo

The folder "FirstRepo" is empty. We will now initialize a repository to our folder.

Git init

git-init

Something called the "master" appears on the screen. Whenever a Git repository is created for the first time, it creates a branch, and the name of the branch is master. Navigate to the folder; you can find a hidden ".git" folder.

If you check the folder, you can see several directories and configurations. Make sure you don't make any changes to any of the directories.

directories

This is created when a repository is initialized.

Moving further, let's make some commits. For that, I will create two notepads and commit them one by one.

For the first notepad, the commands are as follows:

touch alpha.txt

notepad alpha.txt

A notepad opens on the screen. Type anything inside it, save it and close it. notepad

Next, let's check the status of the file that was created.

git status

This shows that there isn't a file committed yet, and there are untracked files. The untracked files can be seen in red.

seen-in-red.

For Git to track that file, add command is used. If you know the exact name of the file, you can specify it and simply type the following command:

git add .

untracked-files.

The next step is to commit the file.

git commit -m "alpha"

branch-master

Let's check the status of the file again.

git-status

git status

You'll notice that there are no more commits to be made, as there was a single notepad and that was committed in the previous step.

Next, check all the information regarding the commits that were made.

git log

git-log

This displays the commit ID, author's name, and email ID used. You can also find the date and commit message on the screen.

Let's make one more commit.

Repeat the same process again. I will make a notepad, add something to it, and close it.

touch beta.txt

notepad beta.txt

git add .

git commit -m "beta"

git-comit

git log

git-log-2

We can see the commit number and order of the commits.

Now, let's push the two notepads on GitHub. Open your GitHub account, and create a new repository. The name of the repository will be "FirstRepo."

firstrepo.J

Copy the "git remote add origin" URL.

Paste the copied URL onto the Git Bash.

git-bash

git remote -v

content

Now, let's push the content on to the remote repository.

git push -u origin master

origin-master

The repository is created on the server, and the content is pushed into that repository. It links the master branch on the local repository to the master branch on the server.

Next, refresh the GitHub page, and you can find all the commits there.

Each commit has a hash ID, which contains the details of each commit.

github\_simpli.

You can open each notepad and check the content inside.

github-simpli-2

Learn the basics of Git VCS and understand how to setup Git in your system with the Git Training Course. Check out the course now.

Conclusion

With this how Git works tutorial we went through the Git Push command, and also learned the basics of the push command and followed a hands-on demo of the same. In the demo, we saw how files from the local repository could be pushed to the remote repository. The process makes it possible for the team to stay updated on different people performing different tasks in the same program.