



Let The Data Confess

Follow

Oct 6, 2021 · 7 min read · Listen



Save



Learn Git in just 10 easy steps



In this tutorial, we are going to discuss what is Git, why do people use it in the industry, how to learn git in just 10 easy steps and what is the workflow for Git. Git is the most frequently used open-source tool that every company uses for its software development.

Let's start!!

Why do we need Git?





Suppose you have a project but don't know how to collaborate with others, what will you do?

And if you are changing the code every week, how will you track the changes i.e. how will you do the version control?

Git is the solution for that.

Nowadays, Software Development takes place in a distributive way which means not just one developer work on one solution, there are multiple people work on the same solution independently. It is necessary to have tools that allow Distributed Software Development and Git is one solution to that problem.

Table of Content

We are going to learn Git in 10 easy steps as follows:

1. What is Git
2. What is GitHub
3. How to Create an account on GitHub
4. How to install Git
5. How to set up the configuration
6. How to initialize a Git repository
7. How to clone a GitHub repository
8. Git Workflow
9. Workflow Demo
10. Branching and merging

So first, let's understand what is Git?

What is Git?

Git is a Free and Open Source Distributed Version Control System.



[Open in app](#)

Let's say 10 people are working on the same project. Now, if they want to stay updated with the latest code, one way is to keep exchanging the latest code. But this might be time-consuming.

Another way is to use Git.

It allows multiple developers to access the source code of the project and can modify the code which can also be seen by other developers.

Version Control allows the user to have versions of the project. **Git also stores the history of changes that were made at a particular time.** You can go back to any version if needed. In this way, even if you make any mistake, you have the flexibility of reverting.

Every code is stored in a repository. You can consider a repository as a folder that stores all the code files.

What is GitHub?

Oops! Is there any difference between Git and GitHub? I used to consider that both are the same.

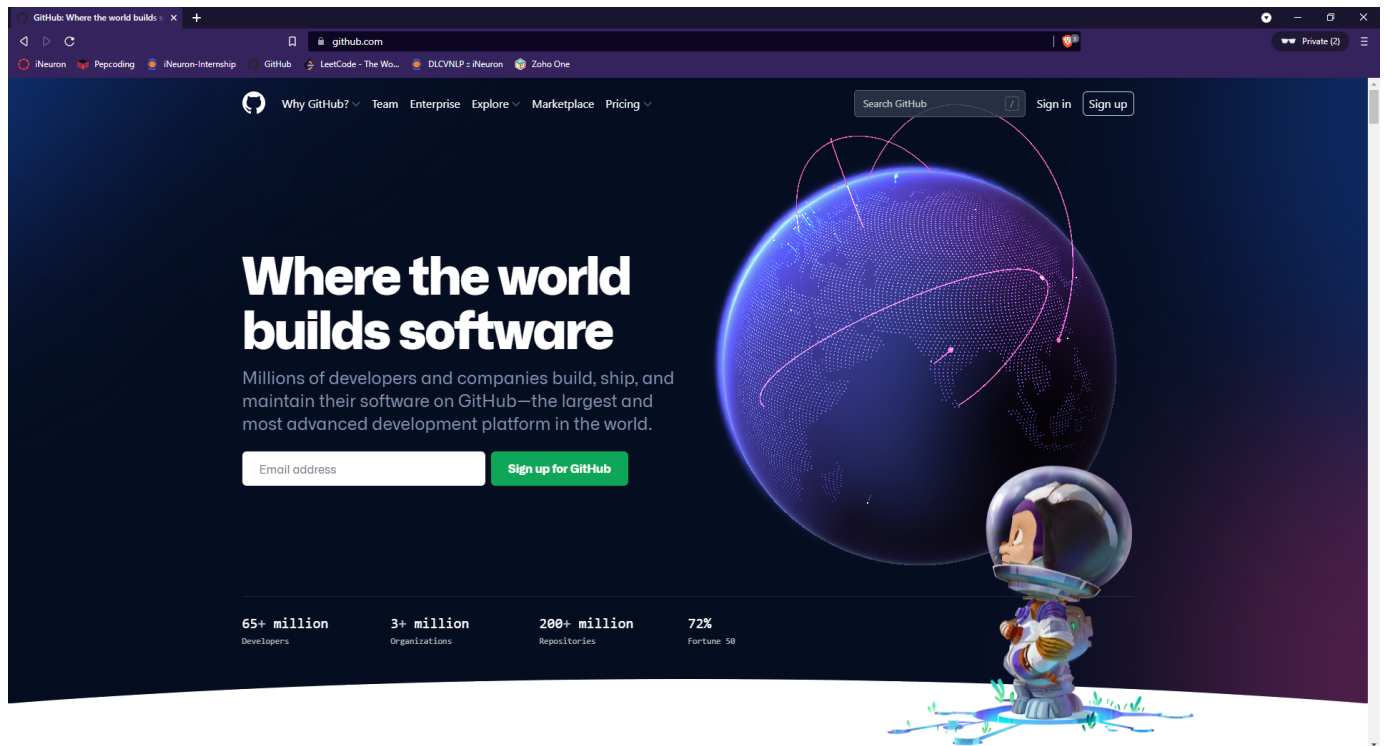


[Open in app](#)

GitHub is a cloud-based hosting service that allows you to manage git repositories.

2. How to create an account on GitHub?

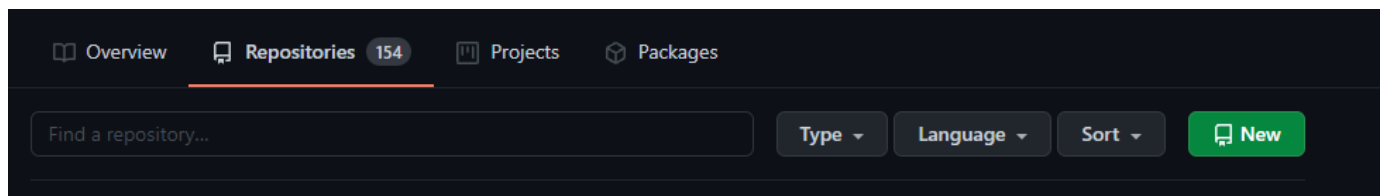
1. Visit [GitHub](#) and sign up with the required details.



GitHub login Page

3. Creating Repository on GitHub

1. Sign in to your GitHub account.
2. Click on New Repository as shown in the below image.




1. Enter the name of the repository you wish to give.
2. Choose whether the repository is public or private. Private repositories can only be accessed by you and to whom you allow access. Public repositories can be seen by anyone.



[Open in app](#)


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner * **Repository name ***

 /

Great repository names are short and memorable. Need inspiration? How about **congenial-pancake?**

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

[Create repository](#)

Congrats, your first repository has been created.

Now we'll learn how to manage the repository using Git. First, let us install git.

4. How to install Git

1. Visit [Git](#) and install according to your Operating System.
2. Now, you can open Terminal (For Mac, Linux) to use Git or Git Bash for Windows.



[Open in app](#)

```
WINDOWS 10@DESKTOP-SO1NR34 MINGW64 ~ ((cbc6cd7...))  
$ |
```

(This is how Git looks like)

Now that we have completed the installation. Let us configure Git to interact with.

5. How to configure Git

Before using Git, you need to initialize it with your GitHub Credentials, i.e User Name, and Email that you used while registering.

```
$ git config — global user.name “YOUR_USERNAME”
```

```
$ git config — global user.email “YOUR_EMAIL”
```

6. How to initialize a Git repository

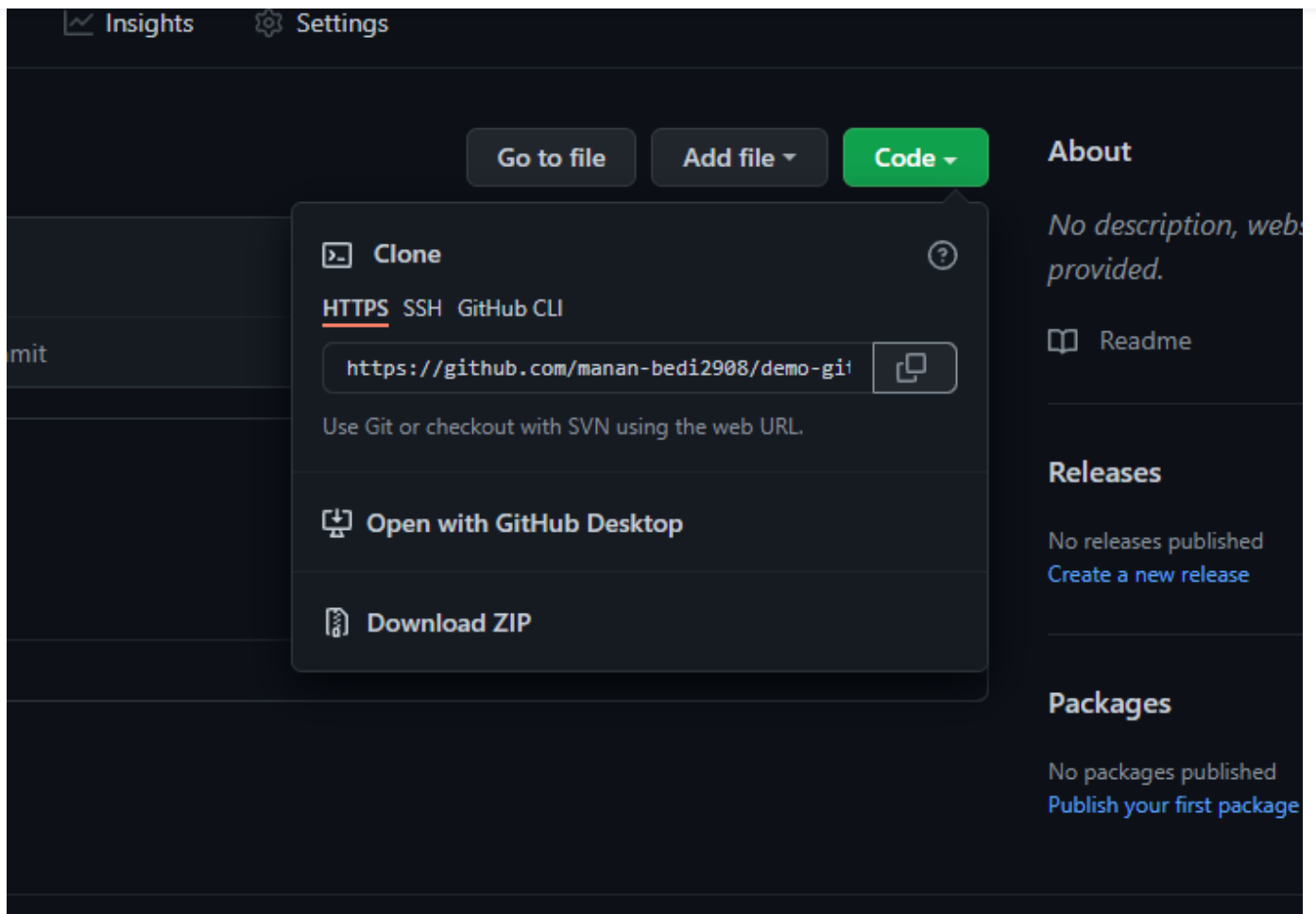
Initializing creates an empty git repository or re-initializes an empty one.

7. Clone a GitHub Repository

Cloning a repository will download the code to your local system so that you can work locally.

1. Go to the GitHub Repository you want to clone.
2. Click on the Code button on the right side of the page.
3. You can either download the code as a zip file or also download using Git Bash.
4. For downloading using Bash, copy the link provided.



[Open in app](#)

```
$ git clone <link>
```

MINGW64:/c/Users/WINDOWS 10/Desktop

```
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop ((cbc6cd7...))
$ git clone https://github.com/manan-bedi2908/demo-git.git
Cloning into 'demo-git'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop ((cbc6cd7...))
$ |
```



[Open in app](#)

**REGISTER NOW
& GET EARLY BIRD
DISCOUNT**
*Early bird offer is valid till 11th October
Visit our website to Register : www.letthedataconfess.com

Time Series Analysis

Through Demand Forecasting Project

Date: **16TH & 17TH**
October

Time: **04:00 PM – 7:30 PM IST**
Both the days

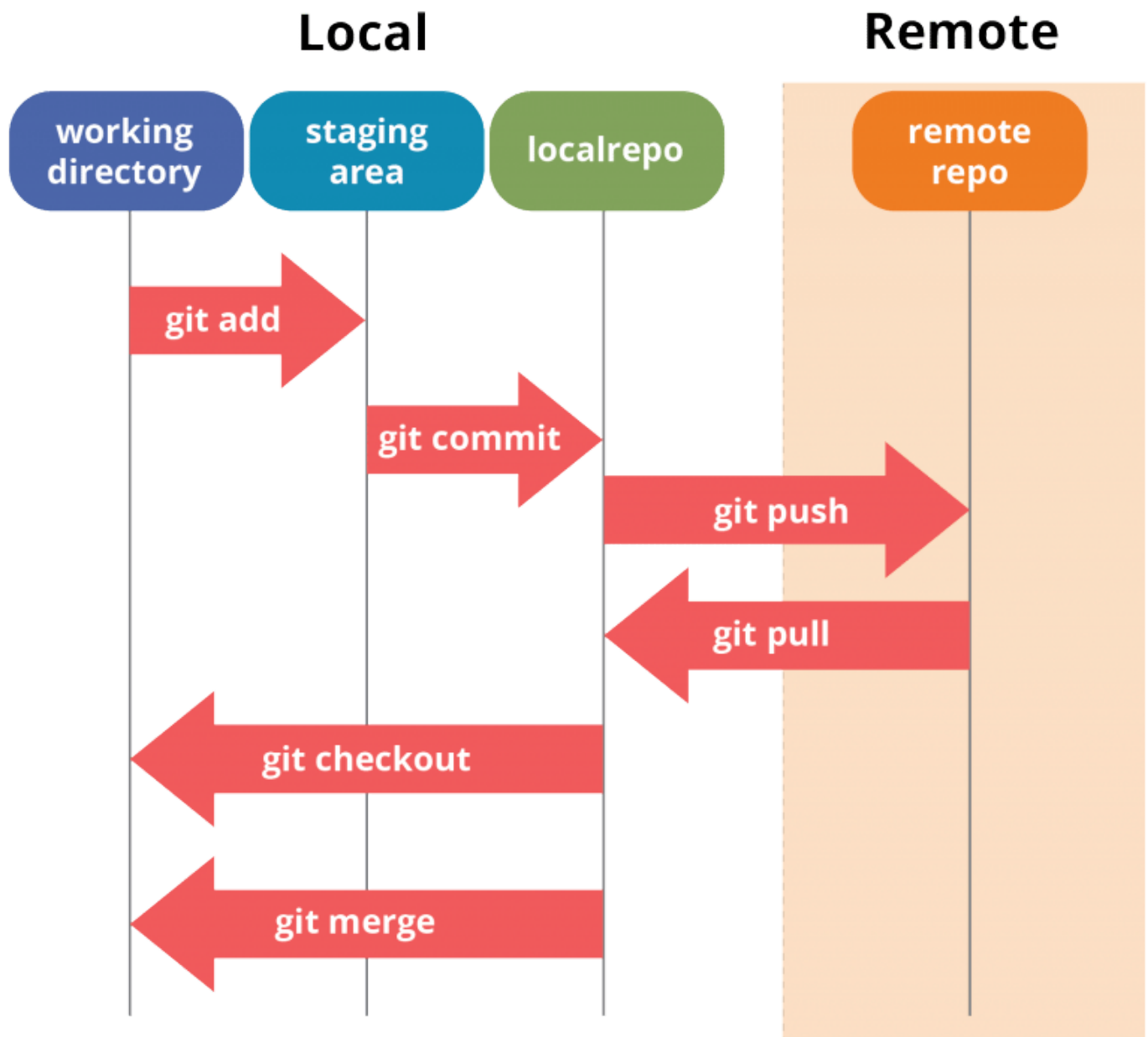
*Additional mentoring session to clear your job related doubts

Follow Us :

8. Git Workflow

There are 4 fundamental elements in the Git Workflow:

1. Working Area
2. Staging Area
3. Local Repository
4. Remote Repository

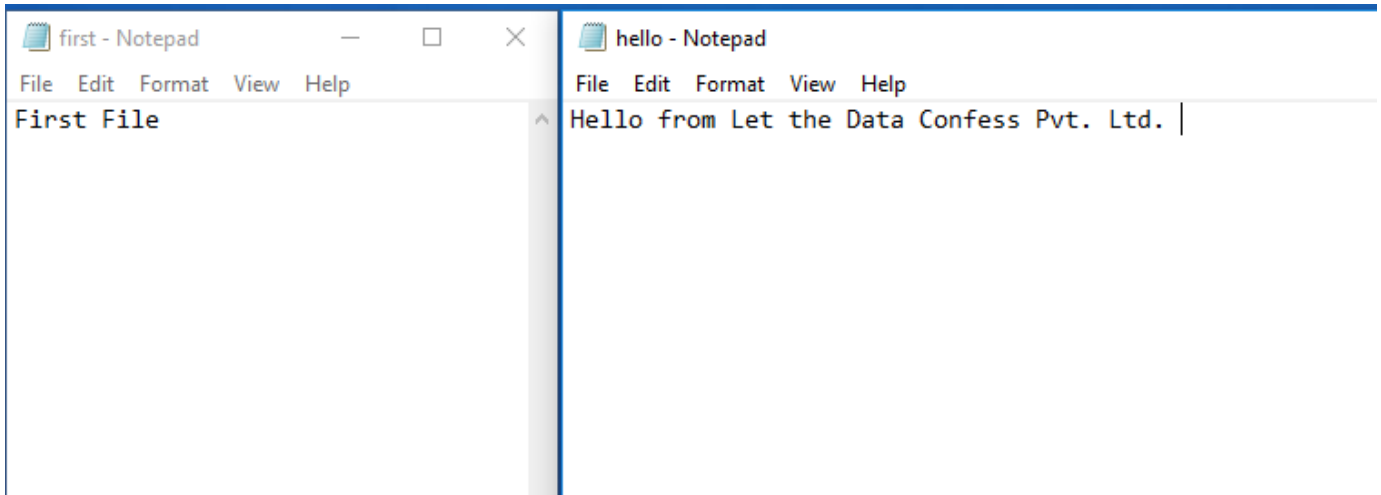


[Open in app](#)

repository later.

```
$ git add .
```

This command adds the files in the working directory to the staging area. Let's suppose I have made two files "hello.txt" and "first.txt" and I want to add them to the staging area.



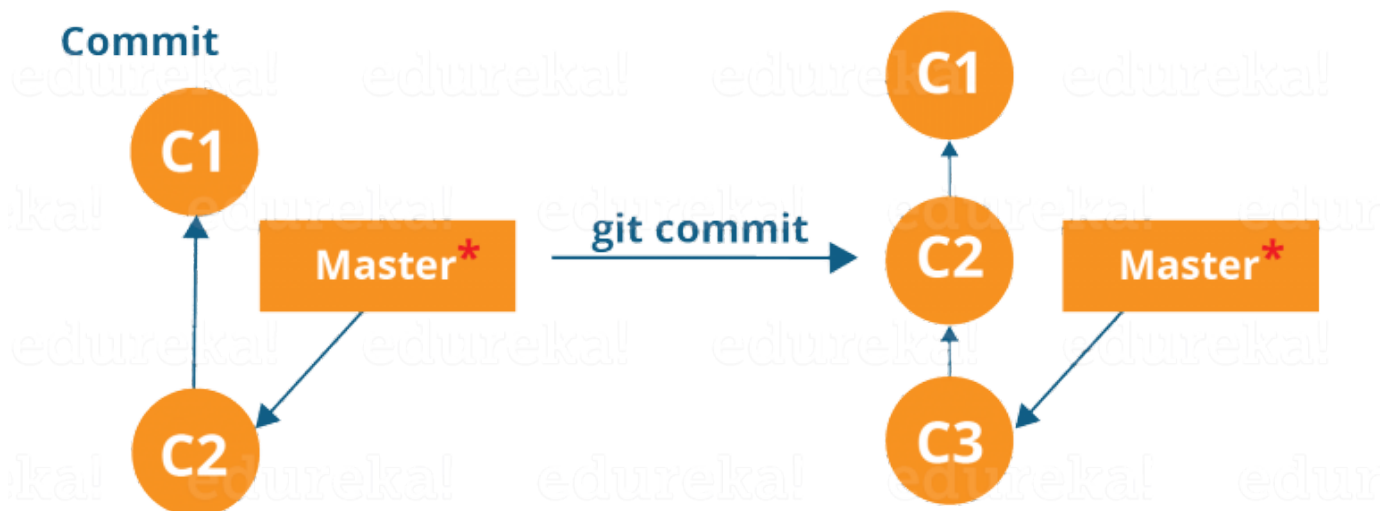
```
MINGW64:/c/Users/WINDOWS 10/Desktop/demo-git
```

```
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
$ git add .
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
```

Now you can push the changes to the local repository.

```
$ git commit -m"add changes"
```

This command pushes the changes to the local repository. -m flag is used to pass a message saying the change you did.



[Open in app](#)

```
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   first.txt
        new file:   hello.txt

(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
$ git commit -m"add two files"
[main f2c512a] add two files
 2 files changed, 2 insertions(+)
 create mode 100644 first.txt
 create mode 100644 hello.txt
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
$ |
```

You can also unstage the files if you realize that you have pushed the wrong changes.

```
$ git reset HEAD~1
```

Now that we have the changes pushed to the local repository, finally, it's time to push the changes to the remote repository also.

```
$ git push origin main
```

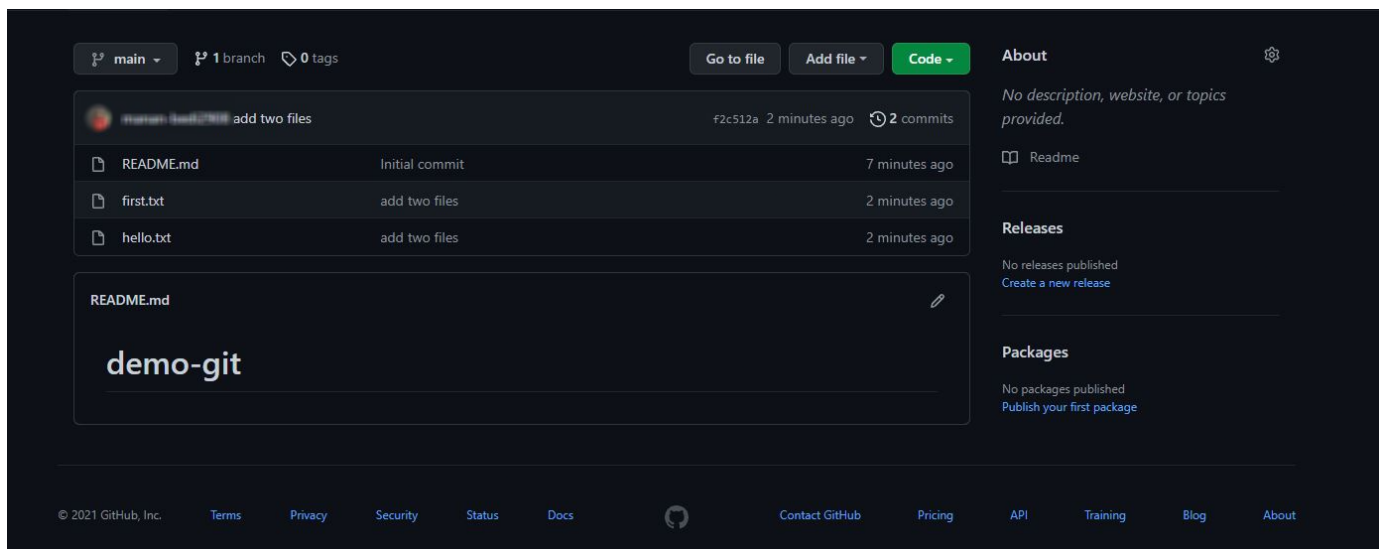


[Open in app](#)

```
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 374 bytes | 93.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/manan-bedi2908/demo-git.git
   16883cf..f2c512a  main -> main
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
$ |
```

The changes are now finally pushed to the remote repository also.

Now, you can see the changes on the repository also.



How to find what is the status of your changes?

```
$ git status
```

Let us suppose you push quite many changes to the remote repository, but now you think that those were not useful and you want to revert to the previous version. How would you do it?

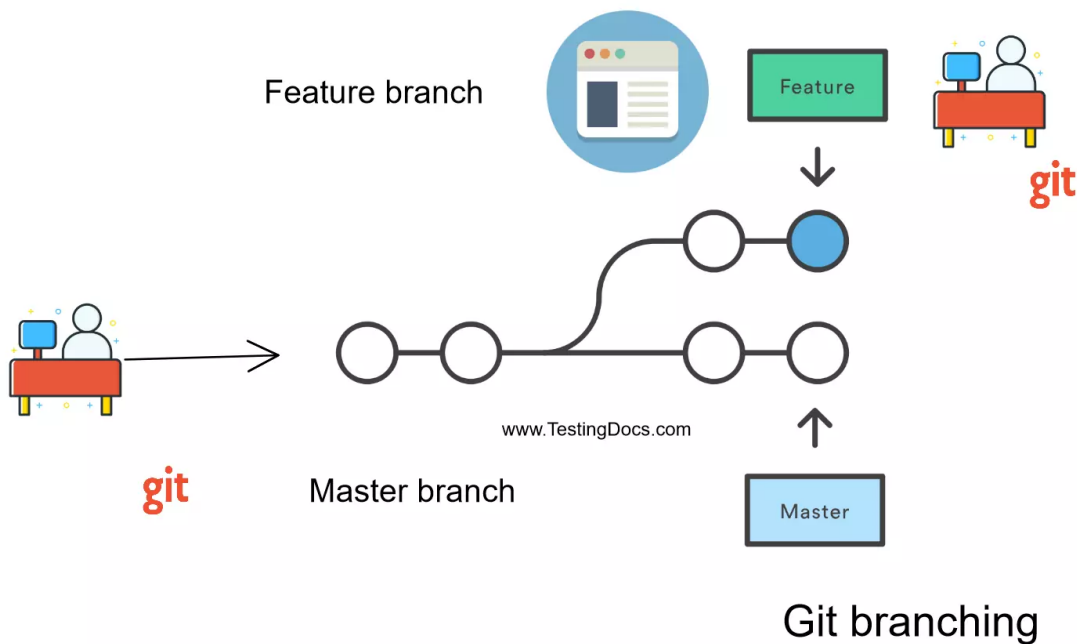
Git also provides functionality to revert to the previous version or any version previously made. The following command will revert to the previous version.




[Open in app](#)

Branching

Git Branches are a pointer to a snapshot of your changes. You can have separate branches for different modules of a project. And at last, you can combine these changes and merge them into the main branch.



[Source: testingdocs.com]

Steps to create branch

1. Creating a new branch first.

```
$ git branch <BranchName>
```

```
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
$ git branch dev
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
```

1. Checking out to the new branch. Let's assume the name of the new branch is dev

```
$ git checkout dev
```

```
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
$ git checkout dev
Switched to branch 'dev'
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (dev)
$ |
```



[Open in app](#)

the main branch. You can do so using

```
$ git merge dev
```

Now, we have added some text in hello.txt, and let us push into the dev branch.

```
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (dev)
$ git status
On branch dev
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   hello.txt

no changes added to commit (use "git add" and/or "git commit -a")
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (dev)
$ git add .
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (dev)
$ git commit -m"add"
[dev 2f234df] add
 1 file changed, 2 insertions(+), 1 deletion(-)
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (dev)
$ git push origin dev
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 308 bytes | 154.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'dev' on GitHub by visiting:
remote:   https://github.com/manan-bedi2908/demo-git/pull/new/dev
remote:
To https://github.com/manan-bedi2908/demo-git.git
 * [new branch]      dev -> dev
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (dev)
```

Now, let us merge the dev branch into the main branch



[Open in app](#)

```
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
$ git merge dev
Updating f2c512a..2f234df
Fast-forward
 hello.txt | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)
(base)
WINDOWS 10@DESKTOP-SOINR34 MINGW64 ~/Desktop/demo-git (main)
$ |
```

Conclusion

Git is a very important tool used in IT Industry for collaboration between teams. It allows to maintain versions and also distribute the code among each member seamlessly. Knowing Git is a must nowadays. I hope, through this tutorial you will be able to implement Git in your projects.

Let us know your feedback in the comments!!

Also, don't forget to check out our upcoming Bootcamp on "Time Series Analysis" where you will learn forecasting fundamentals through the guided project. Here is the link for the details:

Forecasting through time series analysis with the guided project (Live) - Let The Data Confess

This course will teach you the practical skills that would allow you to land a job as a quantitative finance analyst, a...

www.letthedataconfess.com

