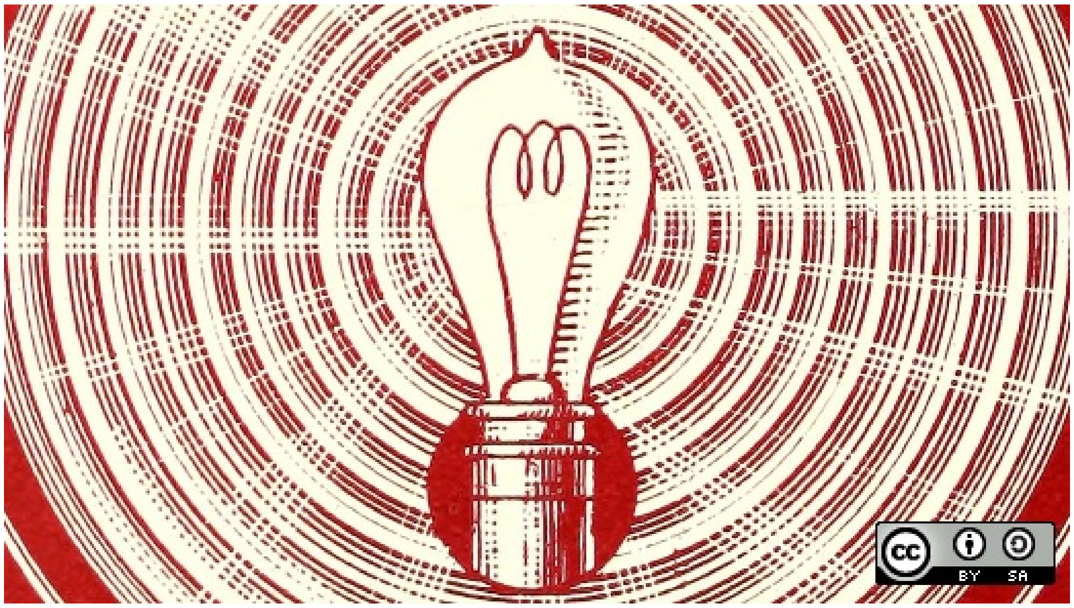
A step-by-step guide to Git

opensource.com/article/18/1/step-step-guide-git

25 Jan 2018 Kedar Vijay Kulkarni (Red Hat) Feed 214 up 10 comments



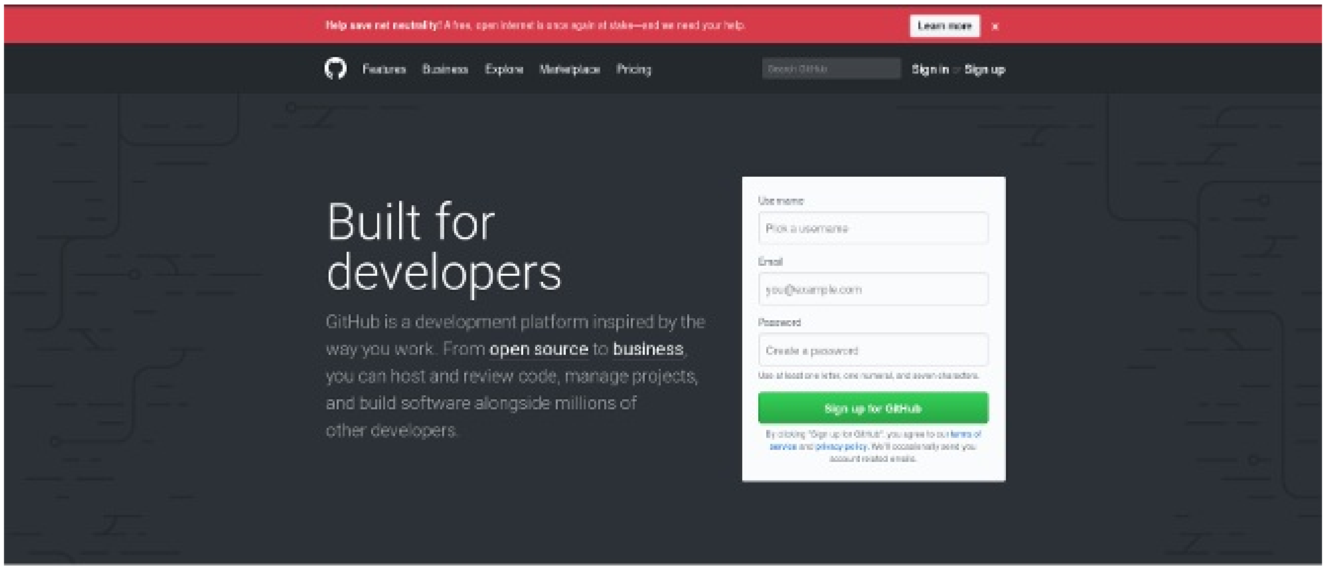
If you've never used Git, you may be nervous about it. There's nothing to worry about—just follow along with this step-by-step getting-started guide, and you will soon have a new Git repository hosted on GitHub.

Before we dive in, let's clear up a common misconception: Git isn't the same thing as GitHub. Git is a version-control system (i.e., a piece of software) that helps you keep track of your computer programs and files and the changes that are made to them over time. It also allows you to collaborate with your peers on a program, code, or file. GitHub and similar services (including GitLab and BitBucket) are websites that host a Git server program to hold your code.

Step 1: Create a GitHub account

The easiest way to get started is to create an account on GitHub.com (it's free).

git_guide1.png



Help save net neutrality! A free, open internet is once again at stake—and we need your help.

Built for developers

GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside millions of other developers.

Username: Pick a username

Email: you@example.com

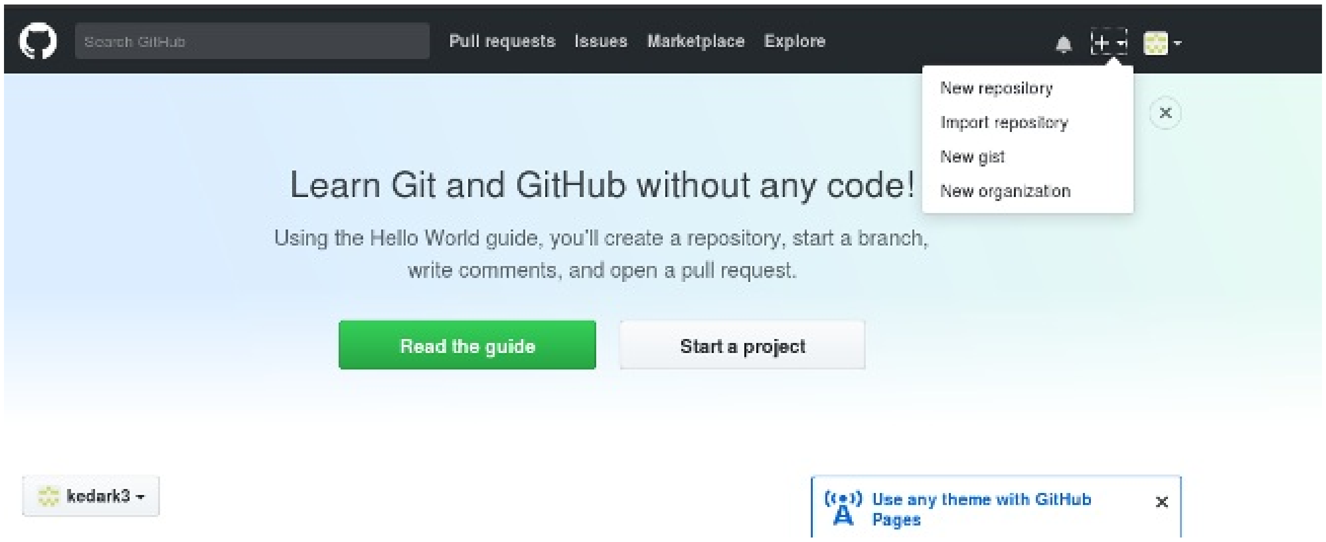
Password: Create a password

Use at least one letter, one numeral, and seven characters.

By clicking "Sign up for GitHub", you agree to our terms of service and privacy policy. We'll occasionally send you account related emails.

Pick a username (e.g., octocat123), enter your email address and a password, and click **Sign up for GitHub**. Once you are in, it will look something like this:

git_guide2.png



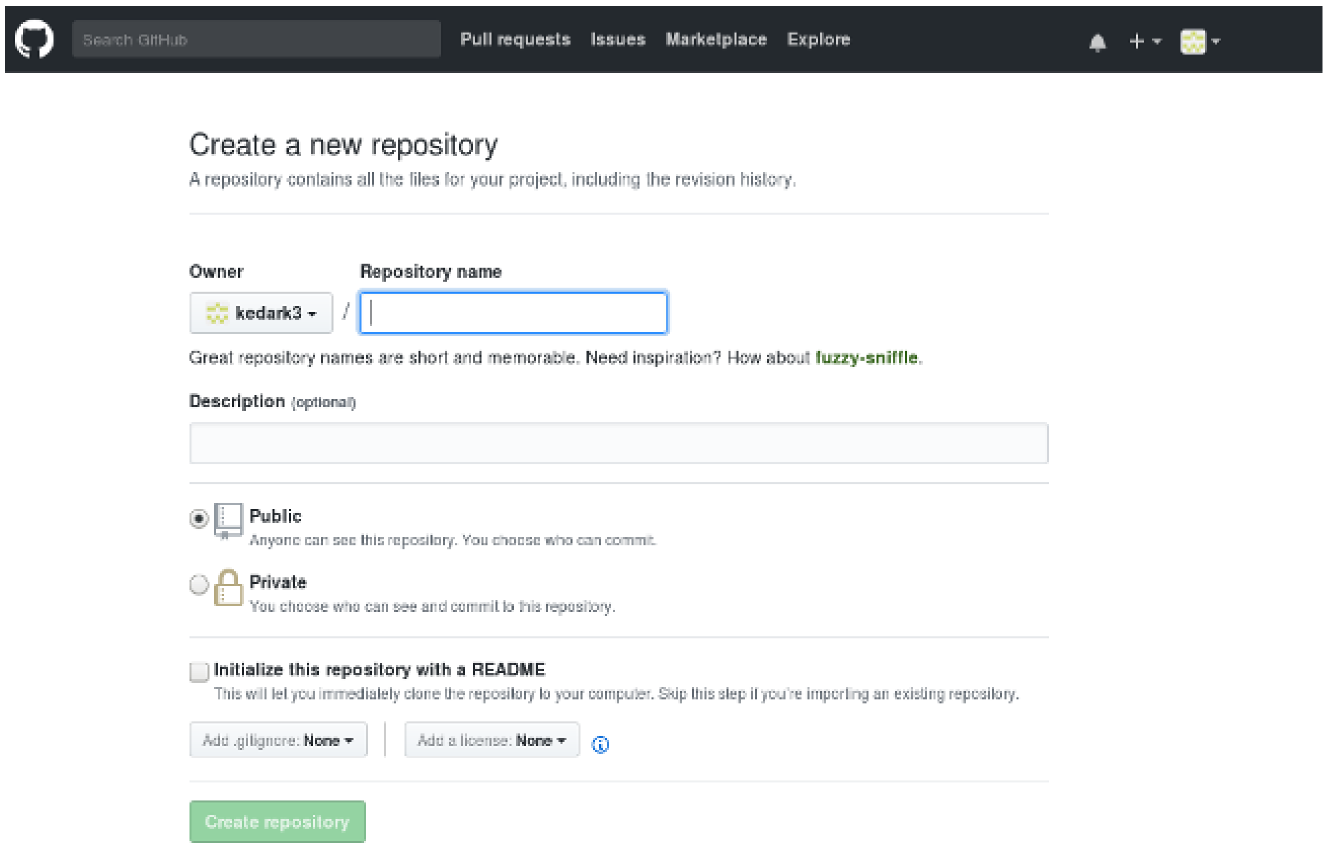
Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

Step 2: Create a new repository

A repository is like a place or a container where something is stored; in this case we're creating a Git repository to store code. To create a new repository, select **New Repository** from the `+` sign dropdown menu (you can see I've selected it in the upper-right corner in the image above).

git_guide3.png



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: kedark3 / Repository name:

Great repository names are short and memorable. Need inspiration? How about **fuzzy-sniffle**.

Description (optional)

● Public
Anyone can see this repository. You choose who can commit.

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

☐ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None

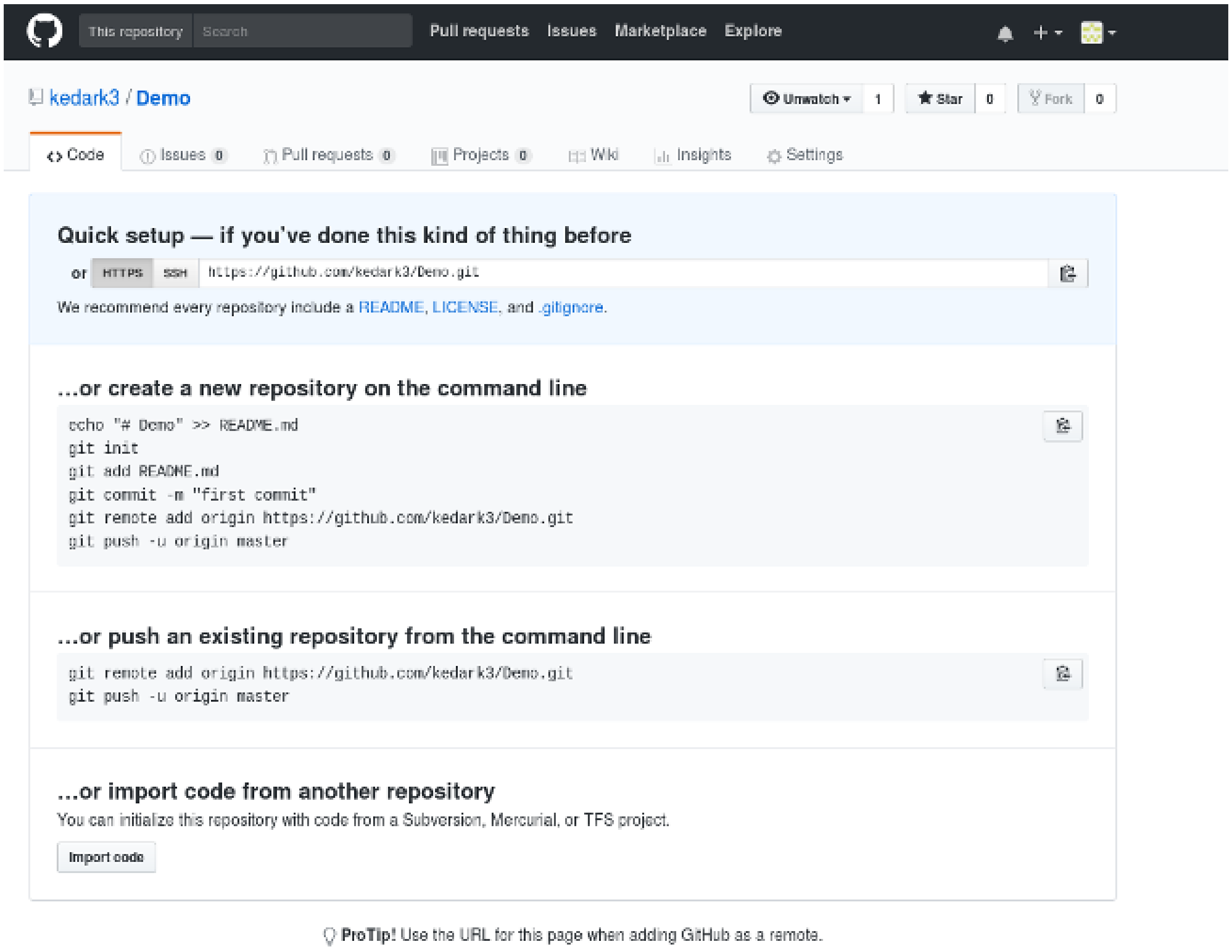
Enter a name for your repository (e.g, "Demo") and click **Create Repository**. Don't worry about changing any other options on this page.

Congratulations! You have set up your first repo on GitHub.com.

Step 3: Create a file

Once your repo is created, it will look like this:

git_guide4.png



Quick setup — if you've done this kind of thing before

or HTTPS SSH `https://github.com/kedark3/Demo.git`

We recommend every repository include a README, LICENSE, and .gitignore.

…or create a new repository on the command line

```
echo "# Demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/kedark3/Demo.git
git push -u origin master
```

…or push an existing repository from the command line

```
git remote add origin https://github.com/kedark3/Demo.git
git push -u origin master
```

…or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

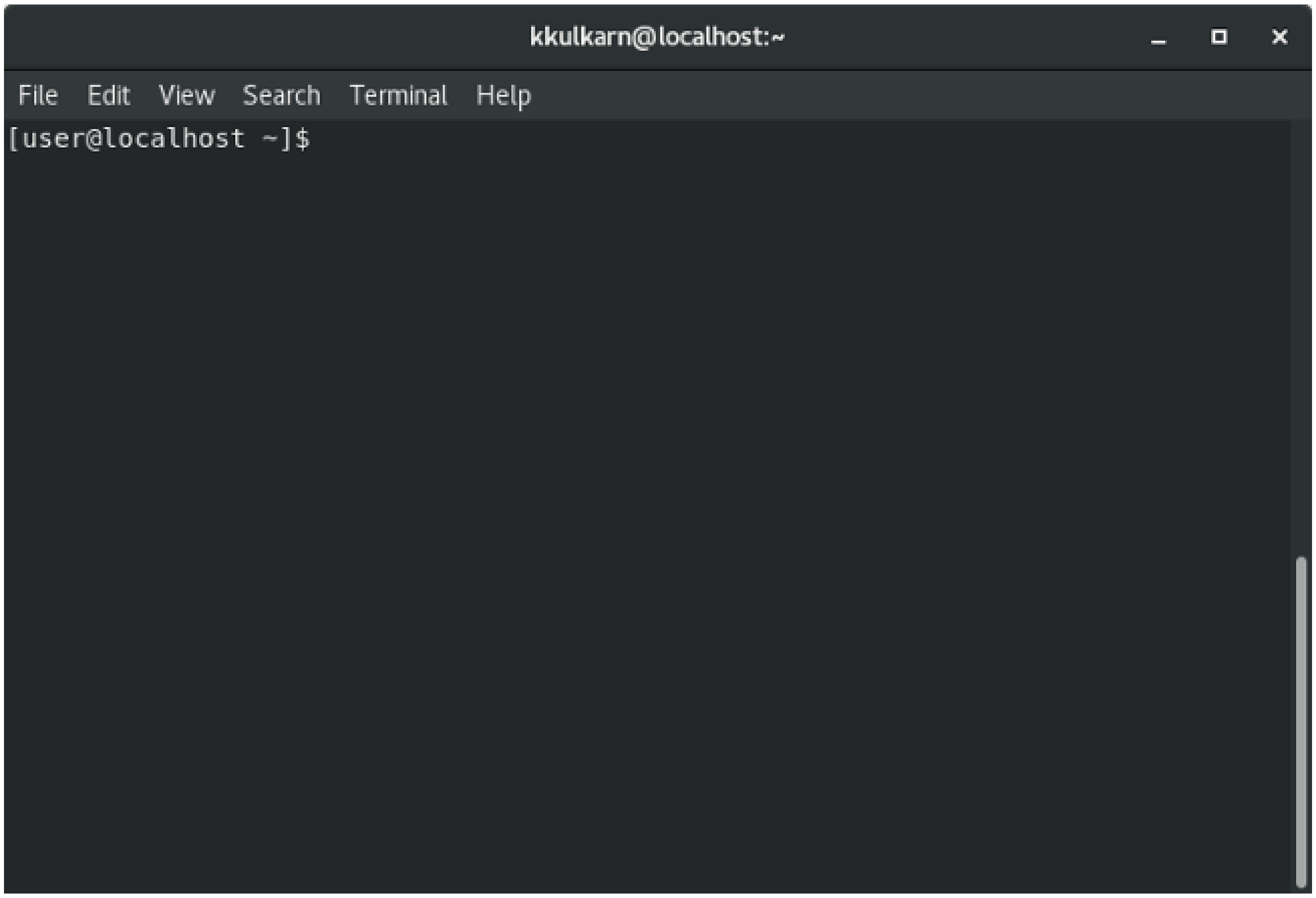
Import code

ProTip! Use the URL for this page when adding GitHub as a remote.

Don't panic, it's simpler than it looks. Stay with me. Look at the section that starts "...or create a new repository on the command line," and ignore the rest for now.

Open the *Terminal* program on your computer.

git_guide5.png



```
[user@localhost ~]$
```

Type `git` and hit **Enter**. If it says command `bash: git: command not found`, then install Git with the command for your Linux operating system or distribution. Check the installation by typing `git` and hitting **Enter**; if it's installed, you should see a bunch of information about how you can use the command.

In the terminal, type:

```
mkdir Demo
```

This command will create a directory (or folder) named *Demo*.

Change your terminal to the *Demo* directory with the command:

```
cd Demo
```

Then enter:

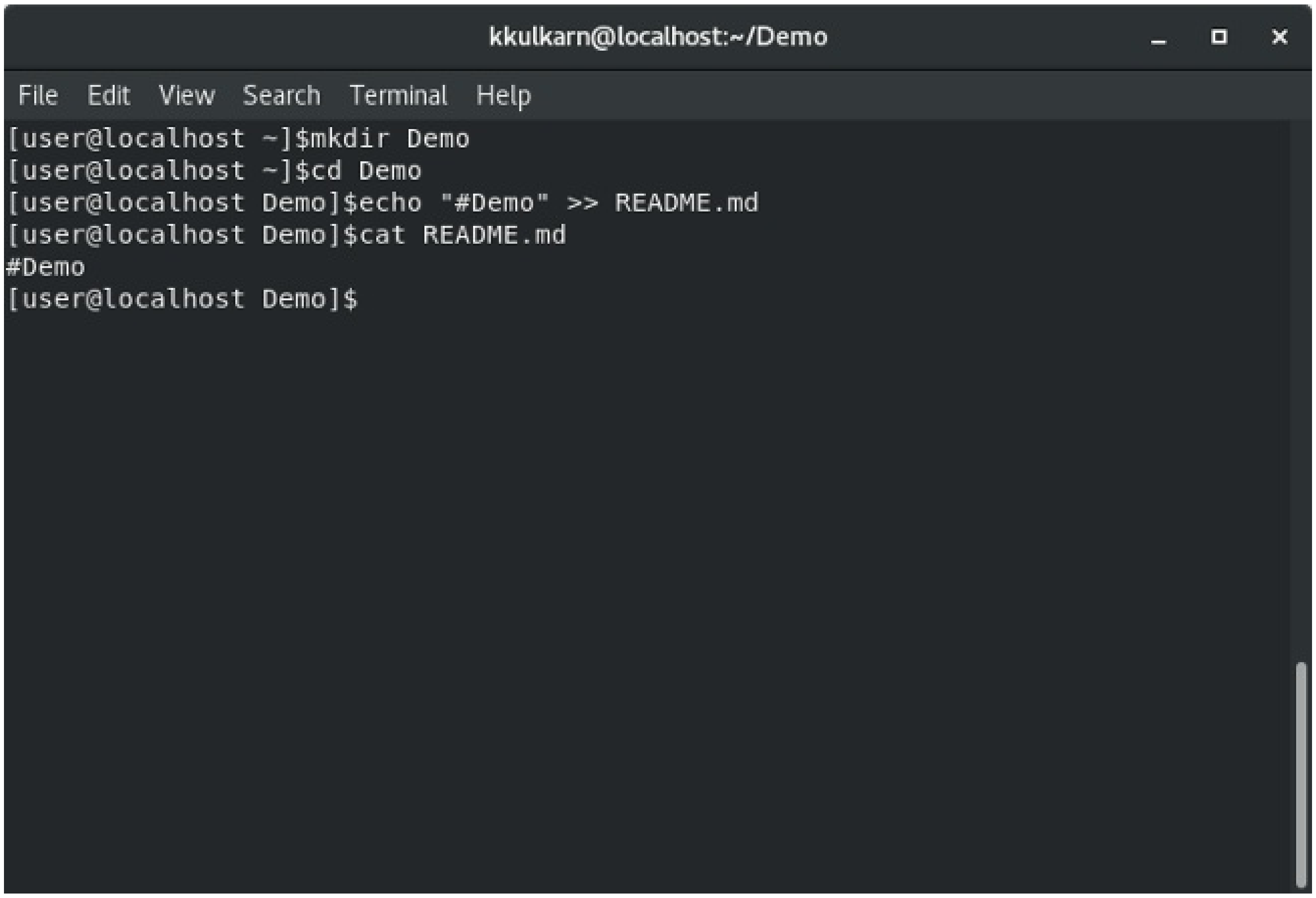
```
echo "#Demo" >> README.md
```

This creates a file named `README.md` and writes `#Demo` in it. To check that the file was created successfully, enter:

```
cat README.md
```

This will show you what is inside the `README.md` file, if the file was created correctly. Your terminal will look like this:

git_guide7.png


kkulkarn@localhost:~/Demo
File Edit View Search Terminal Help

```
[user@localhost ~]$mkdir Demo
[user@localhost ~]$cd Demo
[user@localhost Demo]$echo "#Demo" >> README.md
[user@localhost Demo]$cat README.md
#Demo
[user@localhost Demo]$
```

To tell your computer that *Demo* is a directory managed by the Git program, enter:

git init

Then, to tell the Git program you care about this file and want to track any changes from this point forward, enter:

git add README.md

Step 4: Make a commit

So far you've created a file and told Git about it, and now it's time to create a *commit*. Commit can be thought of as a milestone. Every time you accomplish some work, you can write a Git commit to store that version of your file, so you can go back later and see what it looked like at that point in time. Whenever you make a change to your file, you create a new version of that file, different from the previous one.
To make a commit, enter:

```
git commit -m "first commit"
```

That's it! You just created a Git commit and included a message that says *first commit*. You must always write a message in commit; it not only helps you identify a commit, but it also enables you to understand what you did with the file at that point. So tomorrow, if you add a new piece of code in your file, you can write a commit message that says, *Added new code*, and when you come back in a month to look at your commit history or Git log (the list of commits), you will know what you changed in the files.

Step 5: Connect your GitHub repo with your computer

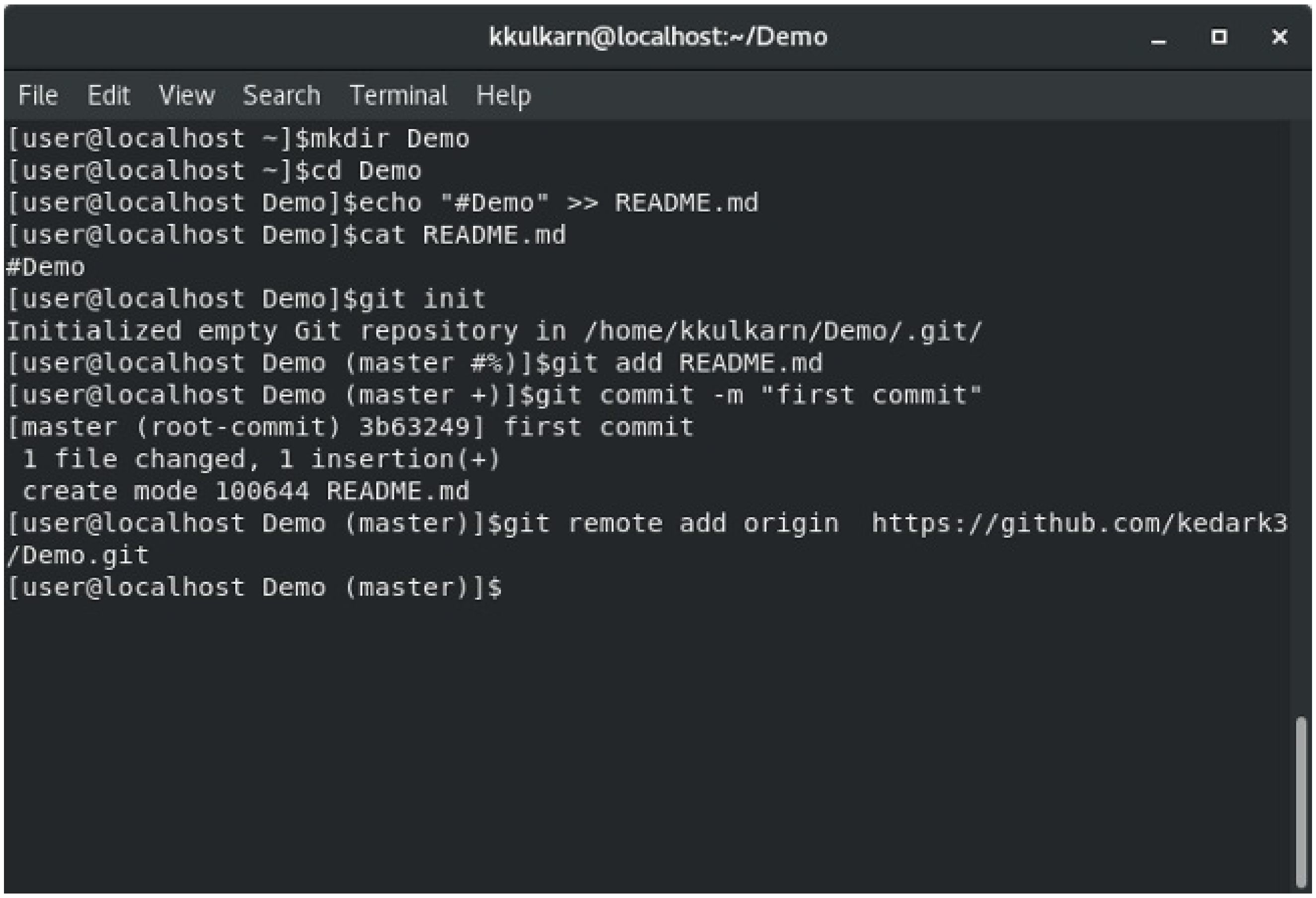
Now, it's time to connect your computer to GitHub with the command:

```
git remote add origin https://github.com/<your_username>/Demo.git
```

Let's look at this command step by step. We are telling Git to add a `remote` called `origin` with the address `https://github.com/<your_username>/Demo.git` (i.e., the URL of your Git repo on GitHub.com). This allows you to interact with your Git repository on GitHub.com by typing `origin` instead of the full URL and Git will know where to send your code. Why `origin`? Well, you can name it anything else if you'd like.

Now we have connected our local copy of the *Demo* repository to its remote counterpart on GitHub.com. Your terminal looks like this:

git_guide8.png

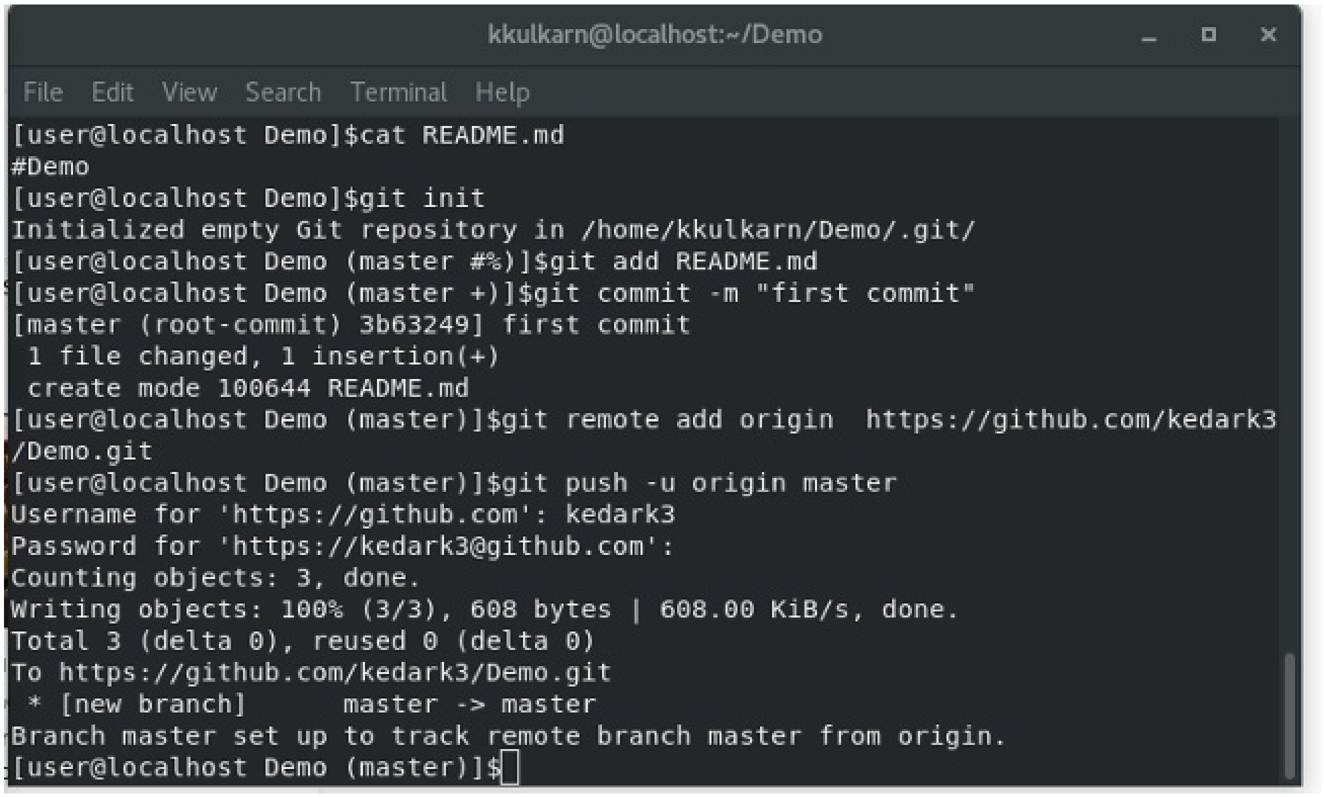

kkulkarn@localhost:~/Demo
File Edit View Search Terminal Help

```
[user@localhost ~]$mkdir Demo
[user@localhost ~]$cd Demo
[user@localhost Demo]$echo "#Demo" >> README.md
[user@localhost Demo]$cat README.md
#Demo
[user@localhost Demo]$git init
Initialized empty Git repository in /home/kkulkarn/Demo/.git/
[user@localhost Demo (master #%)]$git add README.md
[user@localhost Demo (master +)]$git commit -m "first commit"
[master (root-commit) 3b63249] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
[user@localhost Demo (master)]$git remote add origin  https://github.com/kedark3
/Demo.git
[user@localhost Demo (master)]$
```

Now that we have added the remote, we can push our code (i.e., upload our `README.md` file) to GitHub.com.

Once you are done, your terminal will look like this:

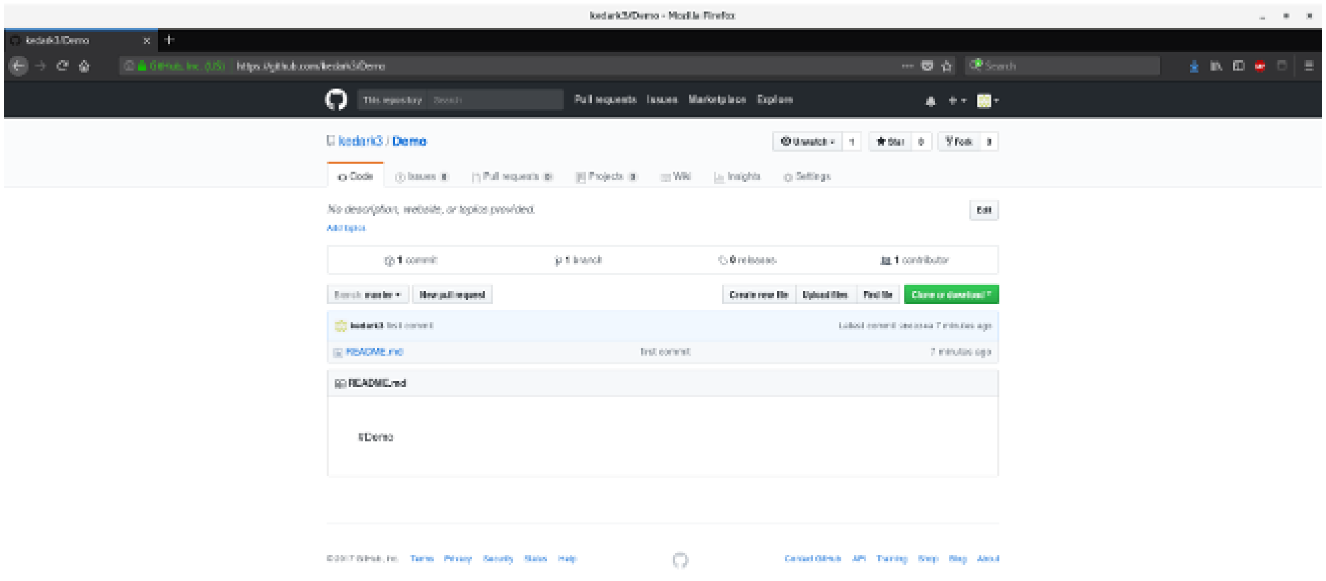
git_guide9.png


kkulkarn@localhost:~/Demo
File Edit View Search Terminal Help

```
[user@localhost Demo]$cat README.md
#Demo
[user@localhost Demo]$git init
Initialized empty Git repository in /home/kkulkarn/Demo/.git/
[user@localhost Demo (master #%)]$git add README.md
[user@localhost Demo (master +)]$git commit -m "first commit"
[master (root-commit) 3b63249] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
[user@localhost Demo (master)]$git remote add origin  https://github.com/kedark3
/Demo.git
[user@localhost Demo (master)]$git push -u origin master
Username for 'https://github.com': kedark3
Password for 'https://kedark3@github.com':
Counting objects: 3, done.
Writing objects: 100% (3/3), 608 bytes | 608.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/kedark3/Demo.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.
[user@localhost Demo (master)]$
```

And if you go to `https://github.com/<your_username>/Demo` you will see something like this:

git_guide10.png


kedark3/Demo - Mozilla Firefox
https://github.com/kedark3/Demo
Pull requests Issues Marketplace Explore
kedark3 / Demo
Code Issues Pull requests Projects Wiki Insights Settings
1 commit
1 branch
0 releases
1 contributor
New pull request
Create new file Upload files Find file Clone or download
README.md
first commit
README.md
#Demo

That's it! You have created your first GitHub repo, connected it to your computer, and pushed (or uploaded) a file from your computer to your repository called *Demo* on GitHub.com. Next time, I will write about Git cloning (downloading your code from GitHub to your computer), adding new files, modifying existing files, and pushing (uploading) files to GitHub.