

Git and Github for Software Engineers

Benson K. Njunge
Tech Lead, Surv Tech.

Github Bootcamp, June 2020.

What is Git?

is a free and **open source distributed version control system**.

What is a version control system

Version control systems refers to a category of software tools that help a software teams manage changes to source code over time.

Version control software keeps track of every modification to the code in a special kind of database.

If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members.

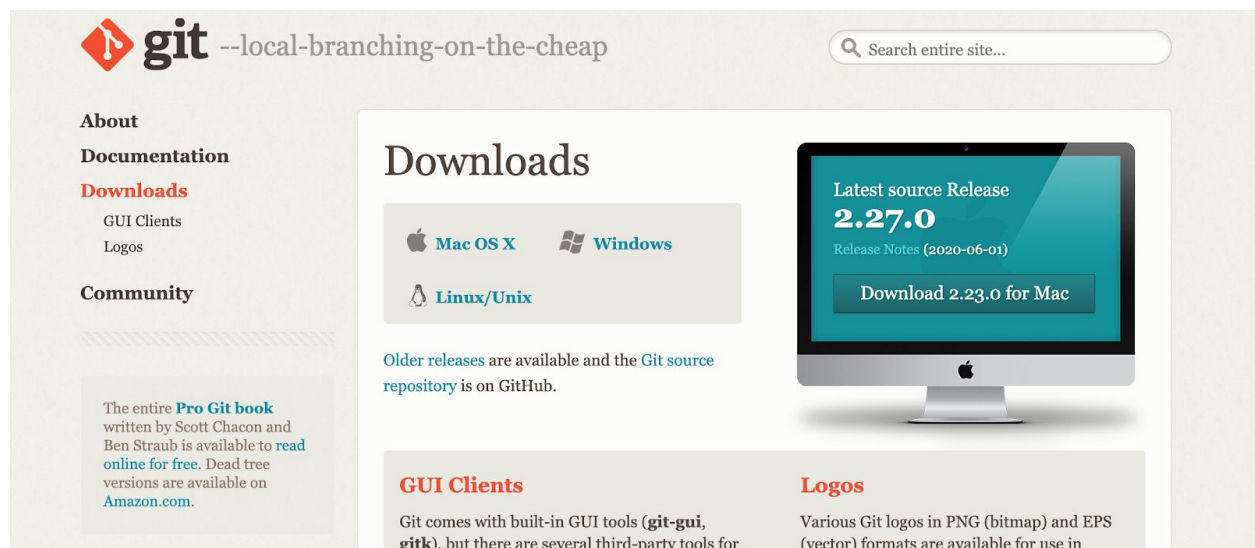
Git, being a version control system is designed to handle everything from small to very large projects with speed and efficiency.

Installing Git

Git is available to install for various platforms/OS's available, if you are on Windows, MacOS, Linux, this is how you install on each.

Download a release of Git for your machine on the link below and follow install

<https://git-scm.com/downloads>



Note that for this tutorial we will be using git on the command line only. This way, it's easier to learn git using git-specific commands first and then to try out a git GUI once you're more comfortable with the command.

Linux, open terminal and type
`sudo apt install git-all`

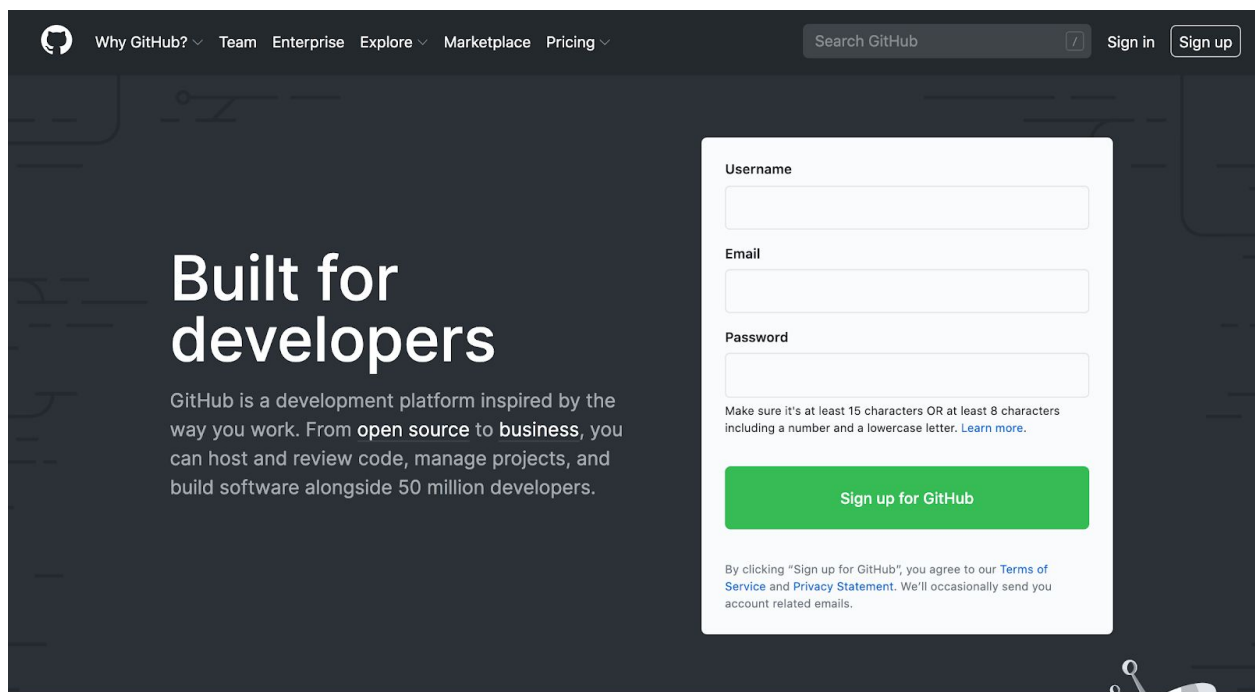
Mac OS

Recent versions of MacOS comes with Git preinstalled, and if not, then its easier, open terminal and type:

`git --version`

If git is not already installed, it will prompt you to install, follow the steps and the installation will complete as usual.

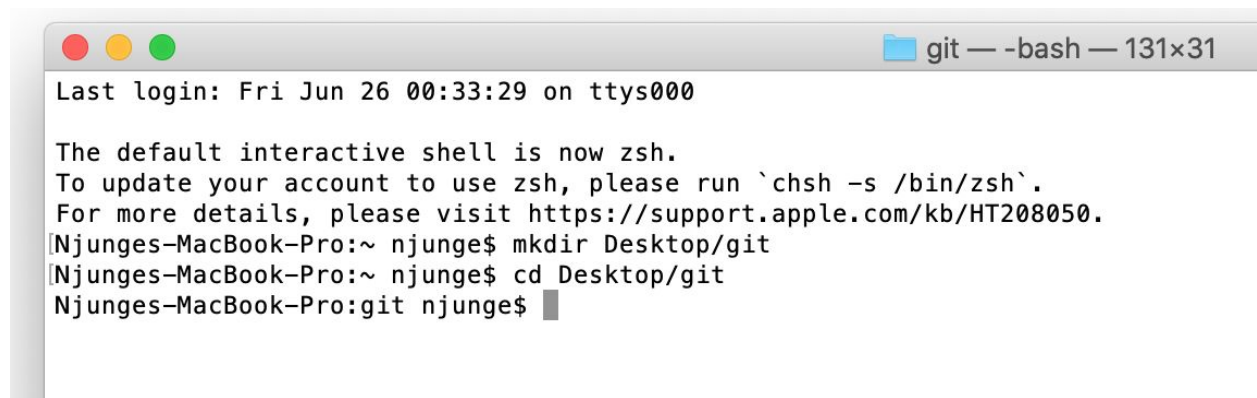
Once you are done with the installation, create an account at github.com

The image is a screenshot of the GitHub website's sign-up page. The background is dark with a faint, stylized pattern of lines and dots. On the left side, the text "Built for developers" is prominently displayed in a large, white, sans-serif font. Below this, a smaller line of text describes GitHub as a development platform inspired by the way you work, mentioning open source, business, code hosting, project management, and 50 million developers. On the right side, there is a white rectangular box containing the sign-up form. The form has three input fields labeled "Username", "Email", and "Password". Below the password field, there is a note about password requirements: "Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)". A large green button with the text "Sign up for GitHub" is positioned below the form. At the bottom of the form, there is a small line of text stating that by clicking "Sign up for GitHub", the user agrees to the Terms of Service and Privacy Statement, and that they will occasionally receive account-related emails. The top of the page features a navigation bar with links for "Why GitHub?", "Team", "Enterprise", "Explore", "Marketplace", and "Pricing", along with a search bar and "Sign in" and "Sign up" buttons.

Creating a git project: Local Copy

The project we are creating today has nothing to do with code, however, we will create a simple html project and we will use it for demonstration purposes.

So, create a folder on your desktop, for me, I will create

A terminal window titled 'git — -bash — 131x31' with a blue folder icon. The window shows the following text: 'Last login: Fri Jun 26 00:33:29 on ttys000', 'The default interactive shell is now zsh.', 'To update your account to use zsh, please run `chsh -s /bin/zsh`.', 'For more details, please visit https://support.apple.com/kb/HT208050.', '[Njunge-MacBook-Pro:~ njunge\$ mkdir Desktop/git]', '[Njunge-MacBook-Pro:~ njunge\$ cd Desktop/git]', and '[Njunge-MacBook-Pro:git njunge\$]' with a cursor.

```
Last login: Fri Jun 26 00:33:29 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[Njunge-MacBook-Pro:~ njunge$ mkdir Desktop/git
[Njunge-MacBook-Pro:~ njunge$ cd Desktop/git
[Njunge-MacBook-Pro:git njunge$ ]
```

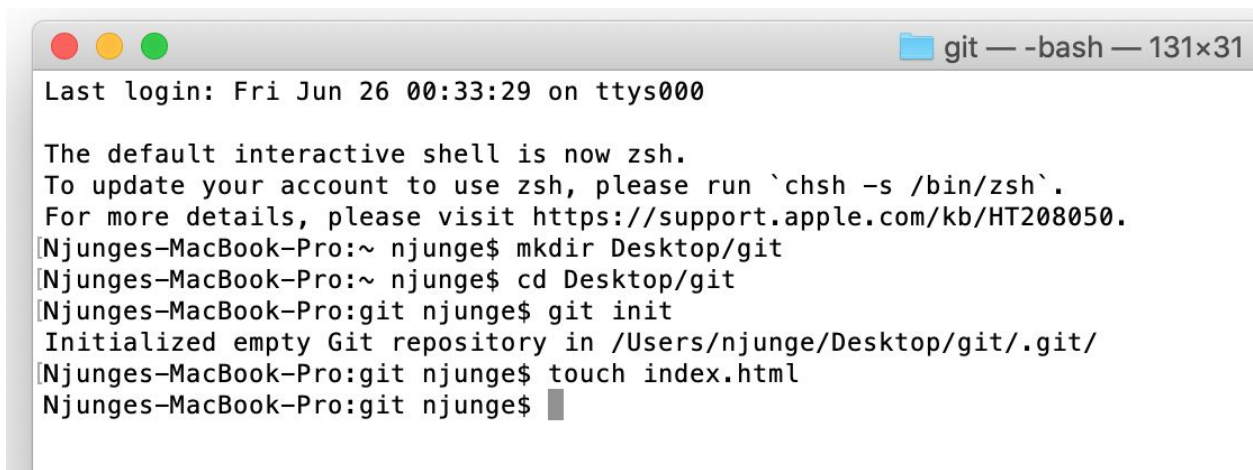
To make it a github managed project, we will run git init

A terminal window titled 'git — -bash — 131x31' with a blue folder icon. The window shows the following text: 'Last login: Fri Jun 26 00:33:29 on ttys000', 'The default interactive shell is now zsh.', 'To update your account to use zsh, please run `chsh -s /bin/zsh`.', 'For more details, please visit https://support.apple.com/kb/HT208050.', '[Njunge-MacBook-Pro:~ njunge\$ mkdir Desktop/git]', '[Njunge-MacBook-Pro:~ njunge\$ cd Desktop/git]', '[Njunge-MacBook-Pro:git njunge\$ git init]', 'Initialized empty Git repository in /Users/njunge/Desktop/git/.git/', and '[Njunge-MacBook-Pro:git njunge\$]' with a cursor.

```
Last login: Fri Jun 26 00:33:29 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[Njunge-MacBook-Pro:~ njunge$ mkdir Desktop/git
[Njunge-MacBook-Pro:~ njunge$ cd Desktop/git
[Njunge-MacBook-Pro:git njunge$ git init
Initialized empty Git repository in /Users/njunge/Desktop/git/.git/
[Njunge-MacBook-Pro:git njunge$ ]
```

Now, we add our first file to our project by typing touch index.html

A terminal window titled 'git — -bash — 131x31' with standard macOS window controls (red, yellow, green buttons). The terminal text shows a login message, a message about switching to zsh, and a series of commands to create a git repository on the desktop and create an initial file.

```
Last login: Fri Jun 26 00:33:29 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[Njunge-MacBook-Pro:~ njunge$ mkdir Desktop/git
[Njunge-MacBook-Pro:~ njunge$ cd Desktop/git
[Njunge-MacBook-Pro:git njunge$ git init
Initialized empty Git repository in /Users/njunge/Desktop/git/.git/
[Njunge-MacBook-Pro:git njunge$ touch index.html
[Njunge-MacBook-Pro:git njunge$
```

Once you've added or modified files in a folder containing a git repo, git will notice that changes have been made inside the repo. But, git won't officially keep track of the file unless you explicitly do so.

To know what files have been added, you will type in your terminal git status

```
git — -bash — 131x31
Last login: Fri Jun 26 00:33:29 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Njunge-MacBook-Pro:~ njunge$ mkdir Desktop/git
Njunge-MacBook-Pro:~ njunge$ cd Desktop/git
Njunge-MacBook-Pro:git njunge$ git init
Initialized empty Git repository in /Users/njunge/Desktop/git/.git/
Njunge-MacBook-Pro:git njunge$ touch index.html
Njunge-MacBook-Pro:git njunge$ git status
On branch master

No commits yet

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

        index.html

nothing added to commit but untracked files present (use "git add" to track)
Njunge-MacBook-Pro:git njunge$
```

At this point, git is able to see your file but since you haven't 'asked' it to track, it will just stay there as is. Now the next step is called STAGING.

During the staging stage* we will now ask git to keep track of the file as we write code to it.

Here, we will look at the first step of staging, the **commit**.

A **commit** is a record of what files you have changed since the last time you made a commit.

Essentially, you make changes to your repo (for example, adding a file or modifying one) and then tell git to put those files into a commit.

Commits make up the essence of your project and allow you to go back to the state of a project at any point.

So, how do you tell git which files to put into a commit? This is where the staging environment or index comes in. As seen when we added a file in our project or when we make changes to our repo, git notices that a file has changed but won't do anything with it (like adding it in a commit).

To add a file to a commit, you first need to add it to the staging environment. To do this, you can use the `git add <filename>` command

Once you've used the `git add` command to add all the files you want to the staging environment, you can then tell git to package them into a commit using the `git commit` command.

Note: The staging environment, also called 'staging', is the new preferred term for this, but you can also see it referred to as the 'index'.

Add a file to the staging environment

Add a file to the staging environment using the `git add` command.

If you rerun the `git status` command, you'll see that git has added the file to the staging environment (notice the "Changes to be committed" line).

So, lets add our file to the staging phase or the index

We type `git add index.html`, but when you have modified a lot of files for a specific file//method/process, you can just type `git add .` note the dot, stands for all in this case

In this case, I will use the `add all` command

```
git — -bash — 131x31
Last login: Fri Jun 26 00:33:29 on ttys000

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Njunge-MacBook-Pro:~ njunge$ mkdir Desktop/git
Njunge-MacBook-Pro:~ njunge$ cd Desktop/git
Njunge-MacBook-Pro:git njunge$ git init
Initialized empty Git repository in /Users/njunge/Desktop/git/.git/
Njunge-MacBook-Pro:git njunge$ touch index.html
Njunge-MacBook-Pro:git njunge$ git status
On branch master

No commits yet

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

    index.html

nothing added to commit but untracked files present (use "git add" to track)
Njunge-MacBook-Pro:git njunge$ git add .
Njunge-MacBook-Pro:git njunge$
```

And now if we check the status of our repo, this is what we will get

```
Njunge-MacBook-Pro:git njunge$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   index.html

Njunge-MacBook-Pro:git njunge$
```


Creating a commit

A commit is a record of files that you have changed since the last commit. It keeps track of what, where has changed.

The message at the end of the commit should be something related to what the commit contains - maybe it's a new feature, maybe it's a bug fix, maybe it's just fixing a typo.

Don't put a message like "asdfasdf" or "foobar". That makes the other people who see your commit sad. Very, very, sad.

So now, lets put in our first commit, on our terminal, we will type, `git commit -m "My first commit"`

A screenshot of a macOS terminal window. The title bar shows three colored window control buttons (red, yellow, green) on the left and a title "git — -bash — 131x31" on the right. The terminal text shows the command `git commit -m "My first commit"` being executed. The output shows the commit hash `2df8a2c`, the message `My first commit`, and a summary of changes: `1 file changed, 0 insertions(+), 0 deletions(-)`. It also shows `create mode 100644 index.html`. The prompt returns to `Njunges-MacBook-Pro:git njunge$`.

```
Njunges-MacBook-Pro:git njunge$ git commit -m "My first commit"
[master (root-commit) 2df8a2c] My first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 index.html
Njunges-MacBook-Pro:git njunge$
```

Now that we're here, its time we take this to another level. We need to push this code to our online repo

A repo is a github repository that will keep our code, allow other developers to participate/collaborate with us. At the same time, we can choose to have our own repository that is private and only invite collaborators or other programmers/engineers we want to work with on the project.

So, lets login to our github account and create a repo.

Sign in to GitHub

Username or email address

Password


[Forgot password?](#)

Sign in


New to GitHub?


[Create an account.](#)


To create a repository, click the **new** button

 **bnjunge** ▼

Repositories

 **New**

 [Repo: Detik/medias...](#)



Choose a repository name, put a description if any is available. Choose if you want your repository to be public or private.

Public means anyone can see it, while private repositories means only you and the invited collaborators can see or make changes to it.

Create a new repository

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

Owner *

Repository name *

 bnjunge /

Great repository names are short and memorable. Need inspiration? How about [psychic-meme?](#)

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.

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

☐ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer.

Add .gitignore: **None** ▼


Add a license: **None** ▼



Create repository

Am going to let the names be the way they are for now, and do not initialize the repo with a readme yet.


Owner ^{*} Repository name ^{*}


 bnjunge / my repository name ✓

Great repository names ^{*} Your new repository will be created as my-repository-name. psychic-meme?

Description (optional)

This is what am using to teach

☒  **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.

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



☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer.

Add .gitignore: **None** Add a license: **None** ⓘ


Create repository

When you click the create button, you will see the following page


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

 Set up in Desktop or **HTTPS** **SSH** `https://github.com/bnjunge/my-repository-name.git` 

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).


...or create a new repository on the command line 

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

...or push an existing repository from the command line 

```
git remote add origin https://github.com/bnjunge/my-repository-name.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.



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

As of now, we have covered a few of the commands shown here, so its time to link our local repository with our remote repository. To do that, we will add remote origin by using the following command

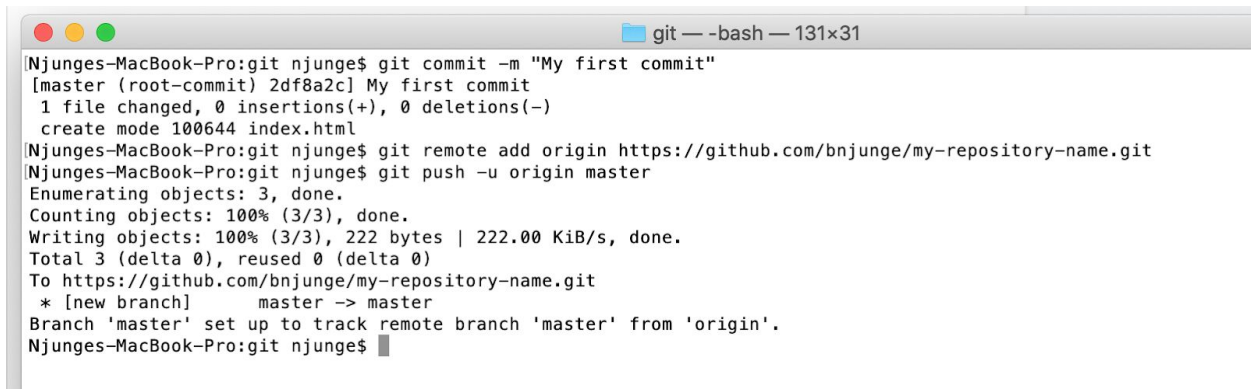
```
git remote add origin https://github.com/bnjunge/my-repository-name.git
```

A terminal window titled 'git — -bash — 131x31' showing the execution of two git commands. The first command is 'git commit -m "My first commit"', which shows the commit details: '[master (root-commit) 2df8a2c] My first commit', '1 file changed, 0 insertions(+), 0 deletions(-)', and 'create mode 100644 index.html'. The second command is 'git remote add origin https://github.com/bnjunge/my-repository-name.git', which is executed successfully.

```
Njunge-MacBook-Pro:git njunge$ git commit -m "My first commit"
[master (root-commit) 2df8a2c] My first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 index.html
Njunge-MacBook-Pro:git njunge$ git remote add origin https://github.com/bnjunge/my-repository-name.git
Njunge-MacBook-Pro:git njunge$
```

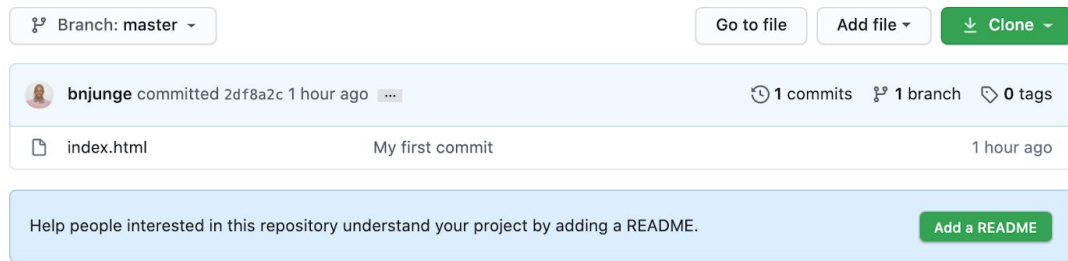
And just like that, we are ready to push our code to our repo.

To send our code to the repo, we use the command `git push -u origin master`

A terminal window titled 'git — -bash — 131x31' showing the execution of 'git push -u origin master'. The output shows the push process: 'Enumerating objects: 3, done.', 'Counting objects: 100% (3/3), done.', 'Writing objects: 100% (3/3), 222 bytes | 222.00 KiB/s, done.', 'Total 3 (delta 0), reused 0 (delta 0)', and 'To https://github.com/bnjunge/my-repository-name.git'. It also shows that a new branch 'master' is set up to track the remote branch 'master' from 'origin'.

```
Njunge-MacBook-Pro:git njunge$ git commit -m "My first commit"
[master (root-commit) 2df8a2c] My first commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 index.html
Njunge-MacBook-Pro:git njunge$ git remote add origin https://github.com/bnjunge/my-repository-name.git
Njunge-MacBook-Pro:git njunge$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 222 bytes | 222.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/bnjunge/my-repository-name.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
Njunge-MacBook-Pro:git njunge$
```

Congratulations! You have just pushed your first code. If you reload your github page you will see your first commit and your file already on your repo



This far is good, but we still haven't covered a few more concepts, and am going to do so now, the next is creating branches. If you are multiple developers working on a single project, or you would like to maintain another version of the project so that you can verify before you let it go live, you will need to create a branch.

Branches allow you to move back and forth between 'states' of a project. For instance, if you want to add a new page to your website you can create a new branch just for that page without affecting the main part of the project.

Once you're done with the page, you can merge your changes from your branch into the master branch. When you create a new branch, Git keeps track of which commit your branch 'branched' off of, so it knows the history behind all the files.

To create a branch on your project, you type **git checkout -b BranchName**

To check if the branch was created, we type **git branch**

```
git — -bash — 131x31
Njunge-MacBook-Pro:git njunge$ git checkout -b firstbranch
Switched to a new branch 'firstbranch'
Njunge-MacBook-Pro:git njunge$ git branch
* firstbranch
  master
Njunge-MacBook-Pro:git njunge$
```

The branch name with the asterisk next to it indicates which branch you're pointed to at that given time.

Now, if you switch back to the master branch and make some more commits, your new branch won't see any of those changes until you **merge** those changes onto your new branch.

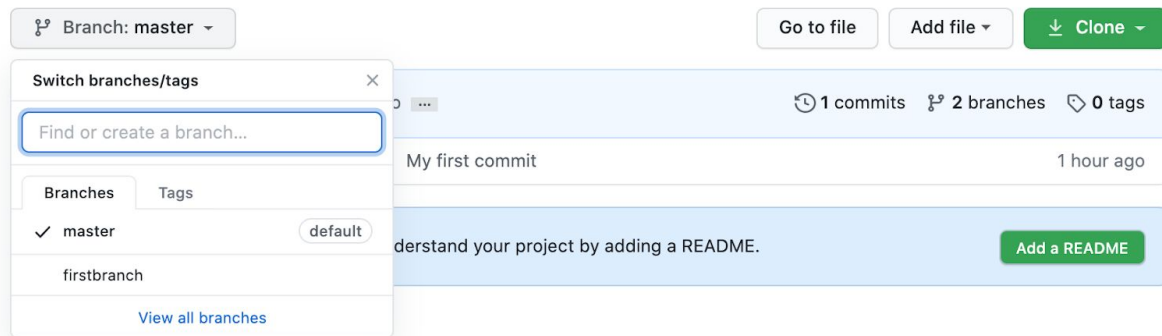
Pushing a branch to github

To push your branch to github, we type **git push -u origin branchname**

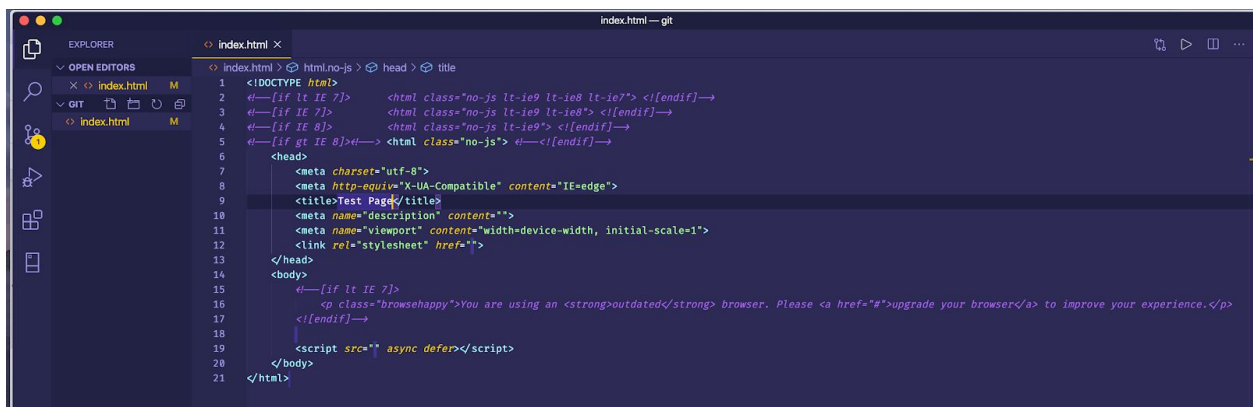
```
git — -bash — 131x31
Njunge-MacBook-Pro:git njunge$ git checkout -b firstbranch
Switched to a new branch 'firstbranch'
Njunge-MacBook-Pro:git njunge$ git branch
* firstbranch
  master
Njunge-MacBook-Pro:git njunge$ git push -u origin firstbranch
Total 0 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'firstbranch' on GitHub by visiting:
remote:   https://github.com/bnjunge/my-repository-name/pull/new/firstbranch
remote:
To https://github.com/bnjunge/my-repository-name.git
 * [new branch]      firstbranch -> firstbranch
Branch 'firstbranch' set up to track remote branch 'firstbranch' from 'origin'.
Njunge-MacBook-Pro:git njunge$
```

Origin to just point out, is an alias for your remote repo, you could use the repo url and it will still work fine.

If you refresh your github page, you will see the branch has reflected.



Lets do a little change to our html page and push as the brach and see what will happen. We will add html5 boilerplate to the code



Then we will add the changes and commit, then push to our repo
Lets now go to our repo and see what will happen.


```

git — -bash — 131x31
Njunge-MacBook-Pro:git njunge$ git add .
Njunge-MacBook-Pro:git njunge$ git commit -m "added html5 boilerplate to our project"
[firstbranch 451460f] added html5 boilerplate to our project
1 file changed, 21 insertions(+)
Njunge-MacBook-Pro:git njunge$ git push -u origin firstbranch
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% (3/3), 665 bytes | 665.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/bnjunge/my-repository-name.git
2df8a2c..451460f firstbranch -> firstbranch
Branch 'firstbranch' set up to track remote branch 'firstbranch' from 'origin'.
Njunge-MacBook-Pro:git njunge$

```

The screenshot shows a GitHub repository page for a branch named 'firstbranch'. At the top, a yellow banner indicates that 'firstbranch' has recent pushes less than a minute ago, with a 'Compare & pull request' button. Below this, the branch is selected, and buttons for 'Go to file', 'Add file', and 'Clone' are visible. A message states 'This branch is even with master.' with links for 'Pull request' and 'Compare'. The commit history shows a single commit by 'bnjunge' 2 hours ago, with a file named 'index.html' and the message 'My first commit'. A blue banner at the bottom encourages adding a README file. On the right sidebar, sections for 'About' (describing the project as 'This is what am using to teach'), 'Releases' (no releases published), and 'Packages' (no packages published) are visible.

Now its time for us to meet another term called PR or simply Pull Request. A pull request is a way of notifying the owner of the project that there are some changes you have made and would like to send the changes to the main codebase. This way, the owner of the project can review the code before accepting the changes to the main project.

To accept a pull request we say its a merge. Or you merge with your main project/master branch.

Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

base: master ← compare: firstbranch ✓ Able to merge. These branches can be automatically merged.

added html5 boilerplate to our project

Write

Preview

H B I ≡ <> 🔗 ≡ ≡ ☑ @ ↻ ↶

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Create pull request

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Reviewers

No reviews

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked issues

Use [Closing keywords](#) in the descr

You can choose to accept the pull request or to just close the pull request altogether if the updates would not be in line with the coding standards or its something that maybe may introduce bugs etc, in our todays class we are going to accept and merge.

added html5 boilerplate to our project #1

Edit Open with

Open

bnjunge wants to merge 1 commit into master from firstbranch

Conversation 0

Commits 1

Checks 0

Files changed 1

+21 -0

bnjunge commented now

Owner

No description provided.

added html5 boilerplate to our project

451460f

Add more commits by pushing to the firstbranch branch on bnjunge/my-repository-name.

Continuous integration has not been set up

GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

This branch has no conflicts with the base branch

Merging can be performed automatically.

Merge pull request You can also open this in GitHub Desktop or view command line instructions.

Write

Preview

H B I ≡ <> 🔗 ≡ ≡ ☑ @ ↻ ↶

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Close pull request

Comment

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

Reviewers

No reviews

Still in progress? Convert to draft

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked issues

Successfully merging this pull request may close these issues.

None yet

Notifications

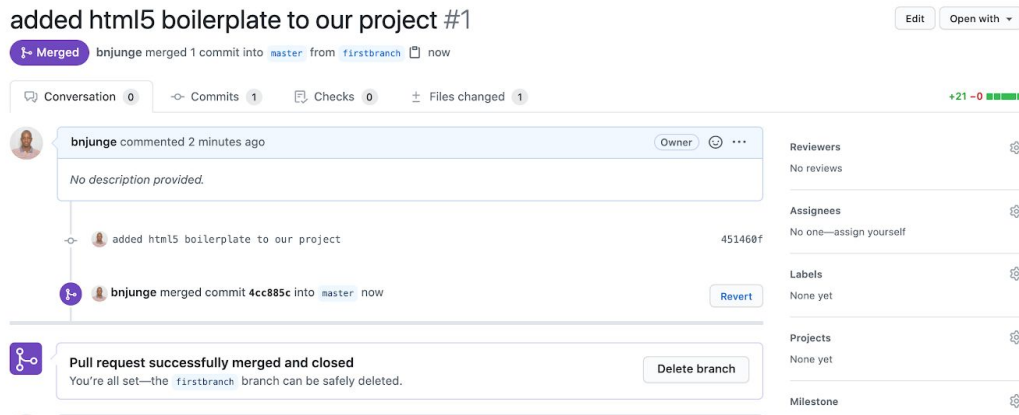
Customize

Unsubscribe

You're receiving notifications because you're watching this repository.

1 participant

And when we have merged, our master branch will be updated accordingly.



Get changes on GitHub back to your computer

To get the changes back to your computer you will need to pull the code.

On terminal type:

git pull -a origin BranchYouWish

You can use **git log** to see the changes

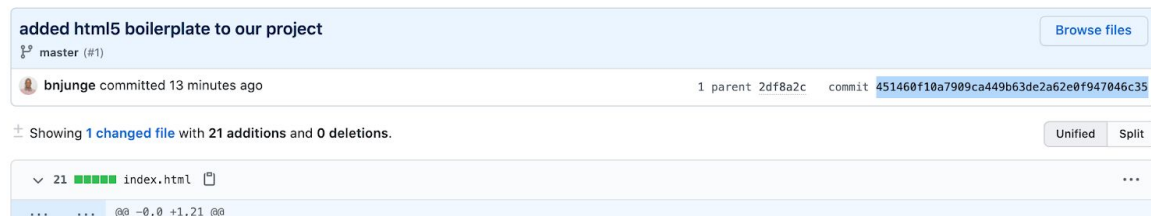
```
Njunge-MacBook-Pro:git njunge$ git log
commit 451460f10a7909ca449b63de2a62e0f947046c35 (HEAD -> firstbranch, origin/firstbranch)
Author: Benson Njunge <survtechke@gmail.com>
Date: Sat Jun 27 01:12:33 2020 +0300

    added html5 boilerplate to our project

commit 2df8a2cebb9659045d364932501438a7e37cda7f (origin/master, master)
Author: Benson Njunge <survtechke@gmail.com>
Date: Fri Jun 26 23:42:27 2020 +0300

    My first commit
Njunge-MacBook-Pro:git njunge$
```

If you wish to undo a specific commit, you can do **git revert hashcode**



The highlighted text

After the revert

Press :wq to save and quit


```
git — -bash — 131x31
commit 451460f10a7909ca449b63de2a62e0f947046c35 (HEAD -> firstbranch, origin/firstbranch)
Author: Benson Njunge <survtechke@gmail.com>
Date: Sat Jun 27 01:12:33 2020 +0300

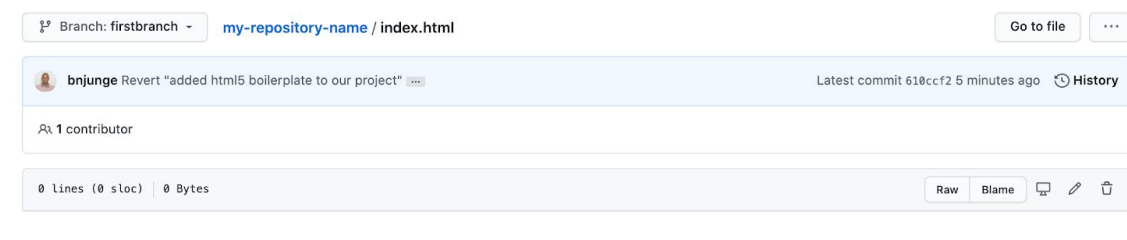
    added html5 boilerplate to our project

commit 2df8a2cebb9659045d364932501438a7e37cda7f (origin/master, master)
Author: Benson Njunge <survtechke@gmail.com>
Date: Fri Jun 26 23:42:27 2020 +0300

    My first commit
[Njunge-MacBook-Pro:git njunge$ git revert 451460f10a7909ca449b63de2a62e0f947046c35
[firstbranch 610ccf2] Revert "added html5 boilerplate to our project"
    1 file changed, 21 deletions(-)
    rewrite index.html (100%)
[Njunge-MacBook-Pro:git njunge$ git revert 451460f10a7909ca449b63de2a62e0f947046c35
On branch firstbranch
Your branch is ahead of 'origin/firstbranch' by 1 commit.
    (use "git push" to publish your local commits)

nothing to commit, working tree clean
[Njunge-MacBook-Pro:git njunge$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (1/1), done.
Writing objects: 100% (3/3), 293 bytes | 293.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/bnjunge/my-repository-name.git
    451460f..610ccf2  firstbranch -> firstbranch
Njunge-MacBook-Pro:git njunge$
```

Lets check our file now



And as you can see, our file is now removed as a commit again.

END

Terms used:

Push - command is used to upload local repository content to a remote repository

Merge - command lets you take the independent lines of development created by **git** branch and integrate them into a single branch

Commit - command is used to save your changes to the local repository.

Origin - is a shorthand name for the **remote** repository that a project was originally cloned from

Master - is a naming convention for a branch. Local repos have only one branch, the default one and is called master branch

Branch - is simply a lightweight movable pointer to one of these commits.

Checkout - is the act of switching between different versions of a target entity

Pull - command is used to **fetch** and download content from a remote repository and immediately update the local repository to match that content.

References:

<https://git-scm.com/docs/>