**Remote Hosting**

If you are working alone, there is no need to remotely host your repos, but if you are working on a project with others remote hosting is essential. There are a few companies who offer remote git hosting. The most popular and well known is GitHub.com, and that’s the one we’ll cover.

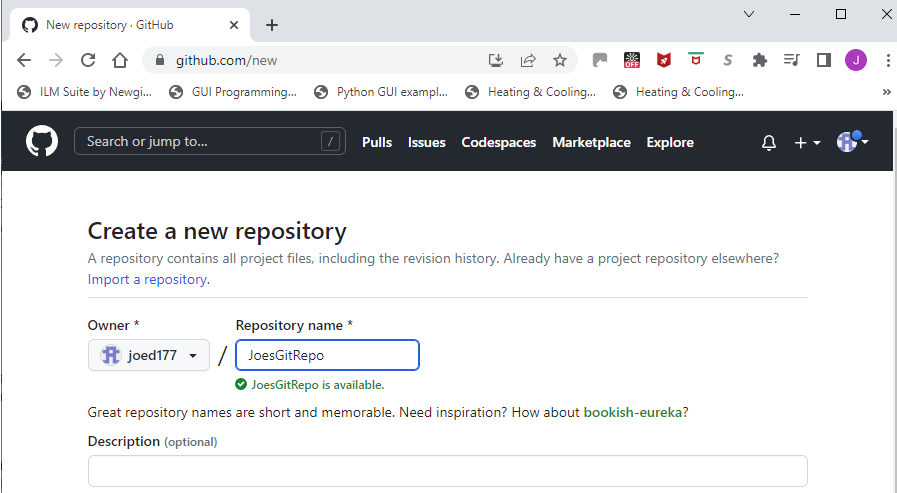
If the number of people in the team is small, typically five or less, and the number of minutes of processing is small, GitHub is free. GitHub provides unlimited public or private repos. If you intend to do more than what’s allowed, you can purchase more capacity. GitHub provides a mobile client and the Git Desktop. The cost of additional features is modest.

Repositories on GitHub can be public or private. Private repos can not be seen by others, unless you provide a link. Public repositories are visible to others, but others can’t change the code, though they can make a copy (clone) of the code for their own use.

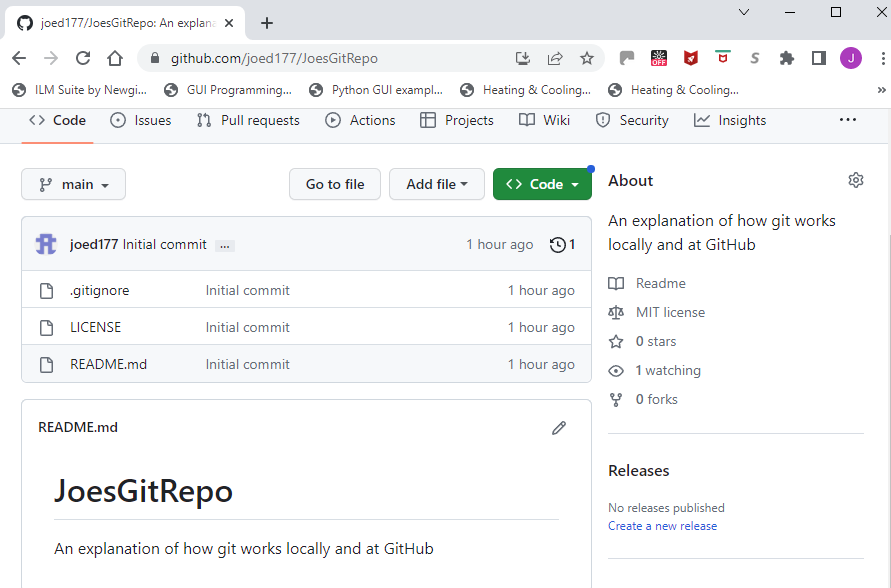
GitHub offers 11,000 actions such as building and deploying your code in addition to source control. Some actions are free, others are fee based.

Obtain a free account and log into the account to follow along.

We are logged into GitHub.com and we are creating a repository called JoesGitRepo.



We’re choosing to make this repo public. We’ll add a brief description of the repo, then we’ll add a readme file, we’ll select the MIT license, and a .gitignore file using the VisualStudio filter. Visual Studio creates lots of files that change every time it is run such as a .exe and .sio files and these will be different on different machines, so there’s no need to track changes to these files. Then we’ll click the Create repository button in the lower right hand section of the screen. With that the repo is created.



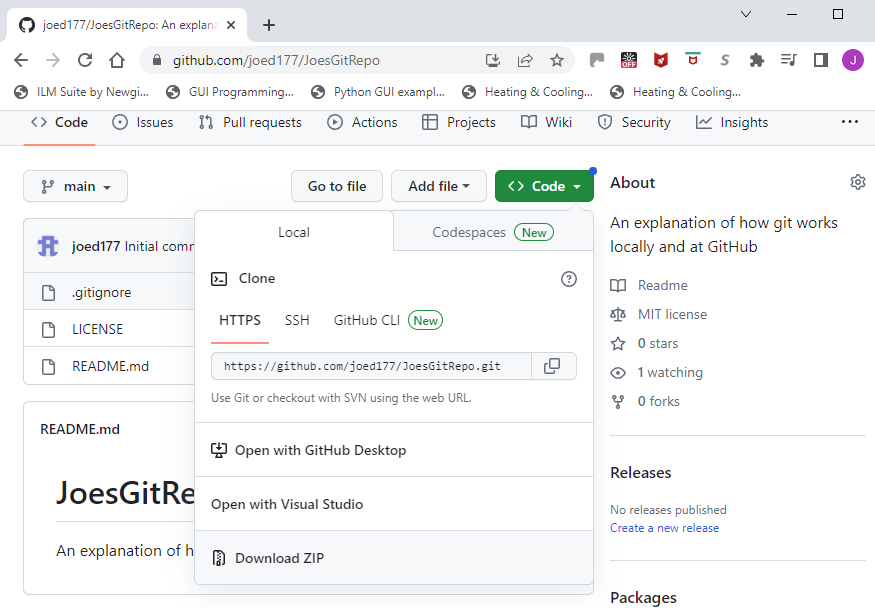
Notice that three files have been created and committed by GitHub. Also notice that we are on the main branch. At this point, there is only one branch. Eventually we’ll want to pull down a copy of the JoesGitRepo repository. When we do this for the first time, the folder containing the local repository should not have a .gitignore, or a README.md.

Once the remote repo has been created, we will want to bring it down to our local machine. To do this, we’ll want to clone the remote repo. If we click the Code button outlined in green above, you will have the option to clone the repo. The first time you do this you will be asked to log in. Once you do this, your credentials will be saved and you will not have to log in again for this repo.

Select HTTPS, outlined in green, then click the icon outlined in red which makes a copy of the URL in the image below. Then on your local machine enter:

Git clone <https://github.com/joed177/JoesGitRepo.git>

A folder named JoesGitRepo will be created on the local machine, and it will include the files that came down from GitHub.



Once you have cloned the repository on your local drive, you can add files. The main branch in the local repository will be called main. The process goes like:

Git branch MyNewBranch # create new branch off main

Git checkout MyNewBranch # puts you on the new branch

Add your files to the repository folder. After you have done this,

Git status # shows the new files not being tracked

Git add filename # OR

Git add . # followed by

Git commit # add files to tracking on MyNewBranch

Git merge main # merges the main branch on to MyNewBranch

Git checkout main # moves control to the main branch

Git merge MyNewBranch # Merges MyNewBranch to main. The two branches should now be #identical

The last two steps move the changed repo to GitHub

Git pull # pulls the repo from GitHub to check for inconsistencies. If there are

# any, they must be resolved before pushing back to Github.

Git push # moves the local repository to GitHub. GitHub will ask you to

#Authenticate

The process is complete.

You have:

* created a blank repository on GitHub,
* cloned (copied) the blank repository to your local machine,
* created a new branch to which you added new files or edited existing files
* added these changes to the local branch
* committed these changes to the local branch
* merged the main branch to the new branch
* switched to the main branch and merged the new branch into the main branch
* pulled the remote repository to check for inconsistencies
* pushed the main branch back to GitHub