


git and github demo session

🕒 Created	@Sep 21, 2020 8:40 PM
👤 Created by	 Abhijeet Pawar
☰ Designation	Project Engineer, CDAC R&D, PUNE.
@ Email	<u>apawar@cdac.in</u>

To create a GitHub account go to link below :



Go to <https://github.com/> and create an account.

Things to do on remote repository [github/gitlab] : -

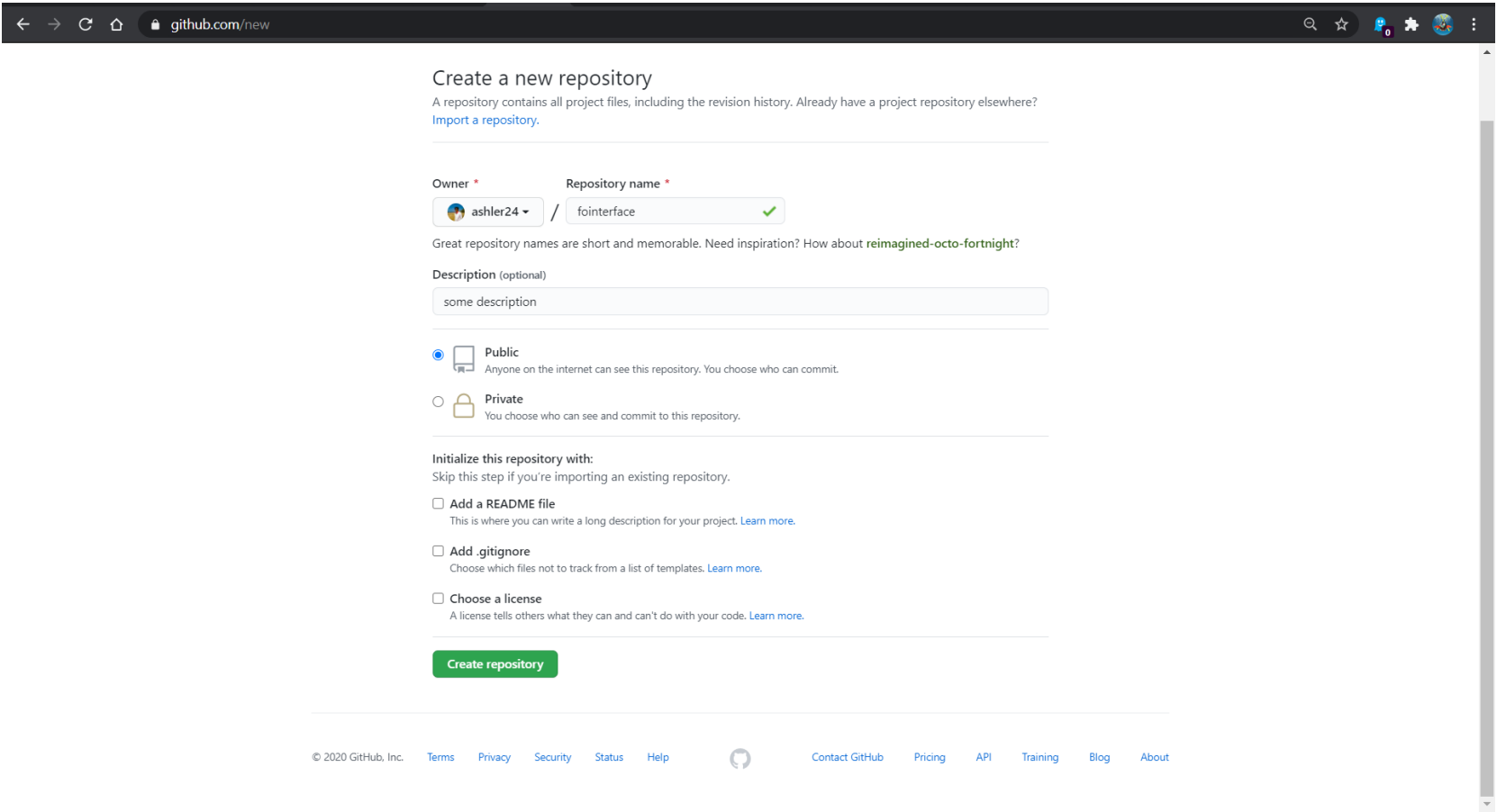


After registering in the GitHub homepage, click on Start a Project to create a new Git repository.



Give name as **fointerface**.


1. Create a new repository on **GitHub**:



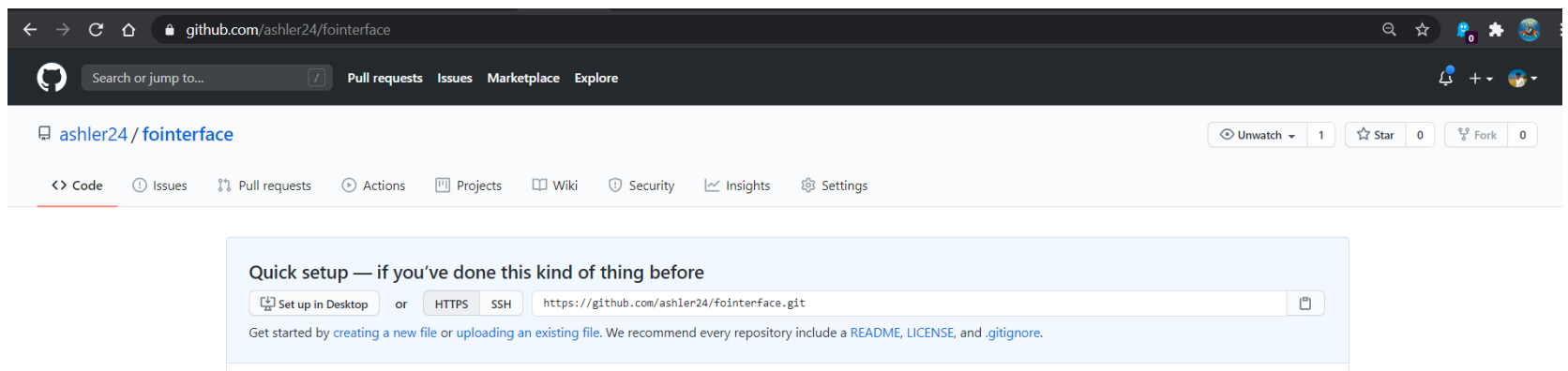
You can make a public repository as well as private. Depending upon your requirements.



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.

2. After creating new repository you see this :



This will be your remote repository url : -

 <https://github.com/ashler24/fointerface.git>

To configure user information in local repository :-

```
$ git config --global user.name "[name]"  
Sets the name you want attached to your commit transactions
```


```
$ git config --global user.email "[email address]"  
Sets the email you want attached to your commit transaction
```

Things to do on your local machine[computer] :- If remote repository on github is not yet created.


1.First make a folder named as **project** your computer :

 Right click → New → folder → project

2.Go in **project** folder and create a folder named as **fointerface** : -

 Right click → New → folder → fointerface

3.Go into **fointerface** folder and open **git bash** :

 Right click → git bash here

4. Initialize a local repository : -

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface
$ git init
Initialized empty Git repository in C:/Users/ashler18/Desktop/project/fointerface/.git/
```



Local/git repository : - A Git repository is the **.git/** folder inside a project. This repository tracks all changes made to files in your project, building a history over time.



If you delete the **.git/** folder, then you delete your project's history.

5. Create a file **index.html** :-

```
<h1>created index.html</h1>
```

6. Now you can see that file is present in our working directory but not yet added in staging directory/index :-

Use command **git status** to see tracked and untracked files :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ 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)
```



Here you can see that index.html is in red color. It means index.html is not yet added in staging index/directory. It is present in working directory.



untracked :- files that are only present in working directory and not in staging index.



tracked :- files that are present in staging index.

7. To add files in staging index :-

Use command **git add .** or **git add <filename>** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git add index.html
```

Now after using **git status** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git status
On branch master
```

```
No commits yet

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



You can see that index.html is in green color which means that file is added in staging index.

8. Now commit the changes that you made :-

Use command **git commit -m "first commit"** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git commit -m "first commit"
[master (root-commit) 61dd385] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 index.html
```



To add and commit at same time :- **git commit -am "first commit"**



A Git commit ID is a SHA-1 hash of every important thing about the commit. I'm not going to list them all, but here's the important ones...

The content, all of it, not just the diff.

Commit date.

Committer's name and email address.

Log message.

The ID of the previous commit(s).

9. To see our commit history :-

Use command **git log --online** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git log --online
61dd385 (HEAD -> master) first commit
```



commit id - 61dd385 commit message - "first commit"

10. To connect local repo to remote repo :-

Use command **git add remote origin <remote repo url>** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git remote add origin https://github.com/ashler24/fointerface.git
```

To check remote repo that is connected use command **git remote -v** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git remote -v
origin  https://github.com/ashler24/fointerface.git (fetch)
origin  https://github.com/ashler24/fointerface.git (push)
```



origin :- In Git, "origin" is a shorthand name for the remote repository that a project was originally cloned from. More precisely, it is used instead of that original repository's URL - and thereby makes referencing much easier.

11. To push our changes from local repo to remote repo :-

Use command **git push -u origin master** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 251 bytes | 125.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/ashler24/fointerface.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

| If remote repository is already exists :-

1. Go to project folder that you made on your local machine/computer :-

Use command **git clone <remote repo url>** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project
$ git clone https://github.com/ashler24/fointerface
Cloning into 'fointerface'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 231 bytes | 1024 bytes/s, done.
```

| Creating branches :

1. To see our local branches :-

Use command **git branch** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git branch
* master
```



branch :- A branch is a pointer to the latest commit in the Git repository.

2. To create a new branch :-

Use command **git branch <branch name>**:-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git branch fointerfacetest
```

To switch to newly created branch use command **git checkout <branch>** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git checkout fointerfacetest
Switched to branch 'fointerfacetest'
```



To create and switch branch in one go - **git checkout -b fointerfacetest**

3. To see all the branches i.e. in local and remote repo :-

Use command **git branch -a** :-

```
$ git branch -a
* fointerfacetest
master
remotes/origin/HEAD -> origin/master
remotes/origin/master
```

Git merge :

1. Now we are in fointerfacetest branch :-

Make some changes in index.html file :-

```
<h2>from fointerfacetest branch</h2>
```

2. Add and commit the changes :

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (fointerfacetest)
$ git commit -am "changes from fointerface branch"
[fointerfacetest 8e00783] changes from fointerface branch
1 file changed, 2 insertions(+), 1 deletion(-)
```



Note :- these changes are only in fointerface test branch as of now. And not yet reflected in master branch. To do that we need to merge these branches.

3. Now go to master branch to merge changes :

Use command **git checkout master** :-

This will switch to master branch.

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (fointerfacetest)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
```

4. To merge branches :-

Use command **git merge <branch name>** :-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git merge fointerfacetest
Updating 61dd385..8e00783
Fast-forward
 index.html | 3 ++-
 1 file changed, 2 insertions(+), 1 deletion(-)
```



Note :- After merging the branch always delete the feature branch. To avoid any problems in future.

5. To delete a feature branch after merging with master branch :-

Use command **git branch -d <branch name>**:-

```
ashler18@ashler18 MINGW64 ~/Desktop/project/fointerface (master)
$ git branch -d fointerfacetest
Deleted branch fointerfacetest (was 8e00783).
```