

- |   |  |
|---|--|
| 1 | Run this notebook as slides by clicking in the "Enter/Exit RISE Slideshow" button in the toolbar |
|---|--|

## Notebook 0

# Git commands for HydroShare Jupyter App (or local Anaconda)

```
In [1]: 1 from datetime import date
2 today = date.today()
3 d2 = today.strftime("%B %d, %Y")
4 print("Updated by Alfonso Torres-Rua, ", d2)
```

Updated by Alfonso Torres-Rua, April 20, 2020

A live document of my experience with git and Hydroshare.

Derived from <https://www.linux.com/learn/beginning-git-and-github-linux-users>  
(<https://www.linux.com/learn/beginning-git-and-github-linux-users>)

## Main point/lesson:

It is important to be organized when developing code. The main tasks are:

1. Creating a GitHub account (**for this class**)
2. Creating a git repository
3. fork a repository to your GitHub account (**for this class**)
4. Cloning Github repository to Hydroshare (or a local Anaconda installation) (**for this class**)
5. Adding files to a local (cloned) git repository
6. Removing files from a local (cloned) git repository
7. Committing and Pushing files to cloud git repository (Github)
8. Pulling files from cloud git repository (Github) (**for this class**)
9. Forcing pulling files from cloud git repository (Github) and rejecting local files changes (**for this class**)

So, for a research project or homeworks of a class, you can start a repository (repo) in GitHub and then clone it to Hydroshare (or an local Anaconda installation)

## 1. Creating a GitHub account:

Go to <https://github.com/> (<https://github.com/>) and create an account. It is free.

## 2. Create a git

## repository (not necessary for this class):

Click on the **NEW** (green button) in your GitHub account.

In [2]:

```
1 %%html
2 <img src='0 Git commands/pict
```

 torresruea ▾

Repositories

 New

Find a repository...

Then:

1. Give it a name (I am calling it playground2)
2. Choose private (only you have the access) or public.
3. Check the "Initialize this repository with a README (the README is where an explanation about the repository is placed)
4. In the Add.gitignore select the programming language (I chose Python)
5. In the add.license select None or an open or close license.
6. Click "Create Repository"

In [3]:

```
1 %%html
2 <img src='0 Git commands/pict
```

### 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 \*  
 / playground2 ✓

Great repository names are short and memorable. Need inspiration? How about [miniature-palm-tree?](#)

Description (optional)

this is an empty repository

☒ **Public**  
Anyone 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: **Python** ▾

Add a license: **None** ▾ ⓘ

Create repository

In the next screen, you can still modify the README.md using the pen icon

```
In [4]: 1 %%html
        2 <img src='0 Git commands/picture2.png', width=700, height=200>
```

The screenshot shows a GitHub repository page for 'torresruea / playground'. At the top, there are buttons for 'Unwatch', 'Star' (0), and 'Fork' (0). Below this is a navigation bar with links for 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Insights', and 'Settings'. The main content area shows the repository description: 'testing git linux commands for cloud interaction. this git does not contain code nor data'. Below this is a section with repository statistics: '1 commit', '1 branch', '0 releases', and '1 contributor'. There are also buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find File', and 'Clone or download'. The commit history shows an 'Initial commit' by 'torresruea' 4 minutes ago. The file list shows 'README.md' as the only file. The README content displays the title 'playground' and the same description text.

### 3. Fork a Github repository to your account:

If we find an online GitHub repository of interest, we may want to have a copy of it (**fork**) on our GitHub account.

Why FORK and not the original link? Many reasons:

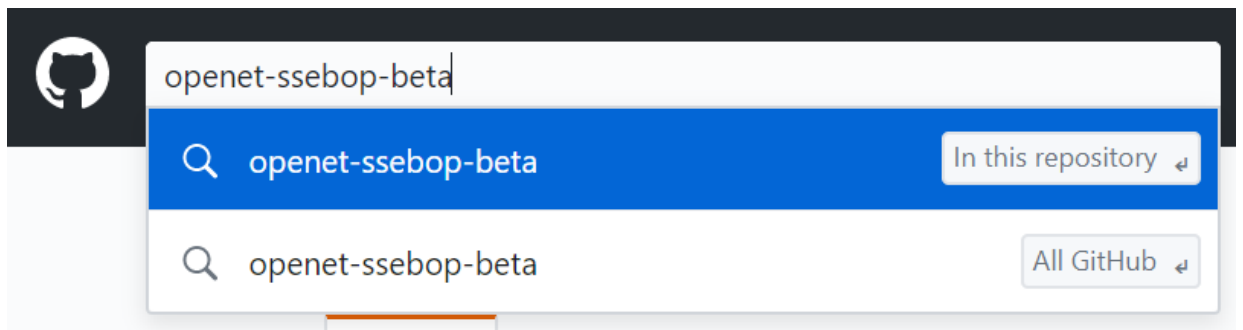
1. The original code may disappear. Author may decide that. See this link  
<https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of->

[code/ \(https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code/\)](https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code/)

2. The original code may change (drastically) and won't work with your developed linked code or modifications.
3. Documenting your work development. I.e. you may be developing a model and having all the components for others to run is necessary (work replicability).
4. You may think of another reason too..

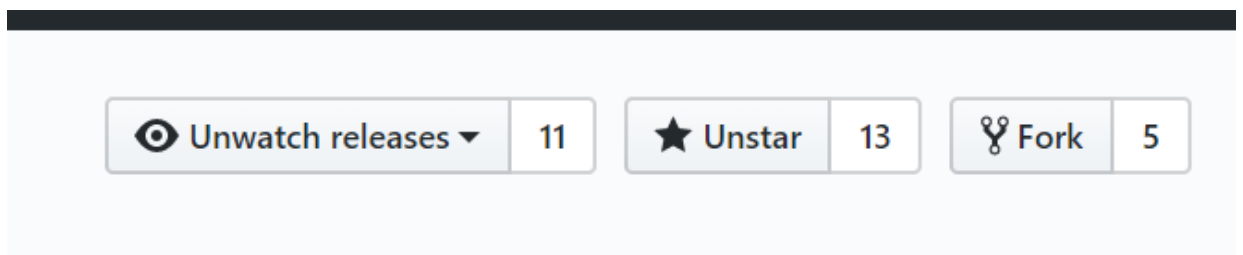
First, we have to find a repository to "fork", to copy it to your Github account. For example, let's fork the repository **openet-ssebop-beta**. You can look for it in the search box and click on "All Github":

```
In [5]: 1 %%html
        2 <img src='0 Git commands/picture3.png', width=700, height=200>
```



Click on Fork:

```
In [6]: 1 %%html
        2 <img src='0 Git commands/picture4.png', width=700, height=200>
```



## 4. Cloning Github repository to Hydroshare (or a local Anaconda installation)

You can clone your own or forked repositories to Hydroshare. Clone means create a copy somewhere else from Github (local computer or the cloud) while keeping a link to Github for synchronization and changes tracking.

To clone we need to create a location where to clone files. I recommend creating a folder.

Go back to HydroShare, and:

1. Open the Jupyter App.

- 
2. Go to Jupyter notebooks view (click on File then Open)
3. Create a git repo folder (I call mine git-repos) using these terminal commands if there is none previously and navigate to it.

```
1 #open terminal from the Jupyter notebooks view
2 # navigate to notebooks and type the two commands below:
3
4 mkdir git-repos
5
6 cd git-repos
```

- 
- 
- 
4. In the same terminal, clone the repository by clicking in the Github green button (which shows a web address like this <https://github.com/torresrui/playground.git> (<https://github.com/torresrui/playground.git>) ) and then typing in the terminal the following command:

```
1 git clone https://github.com/torresrui/playground.git
```

- 
- 
- 
- 
5. pass the command below in the terminal to see the files in the local version of the repository

```
1 ls playground/
```

## 5. Adding files to a local (cloned) git repository

When new files were added to the cloned repo in Jupyter, we need to tell git to track them. Every time you finish working on notebooks, go to a terminal, navigate to the local repo folder and:

```
1 git add *
```

then run commands in #7

## 6. Removing files from a local (cloned) git repository

When files are deleted in the cloned repo, we need to tell git to un-track them. So navigate through a terminal to the local repo folder and:

```
1 git add -u
```

then run commands in #7

## 7. Committing and Pushing files to cloud git repository (Github)

If you are like me who does not know how many files were changed, from a terminal, navigate to the local repo folder and follow these two foolproof commands:

```
1 git commit -a
```

then

```
1 git push origin master
```

## 8. Pulling files from cloud git repository (Github)

If new version of files (or new files were added) are available in the online git repo in GitHub, or when you are working with collaborators in a Github repository (and they did changes to the online repo), from a terminal, navigate to your local repo and type:

```
1 git pull origin master
```

## 9. Forcing pulling files from cloud git repository (Github) and rejecting local files changes (happened in class)

If new version of files (or new files were added) are available from GitHub, AND THE COMMAND FOR #8 DID NOT WORK (ERROR MESSAGE), from a terminal, navigate to the local git repo and type:

```
1 git fetch --all
2 git reset --hard origin/master
3 git pull origin master
```

# THAT'S IT! THE END