

Git & Github Tutorial

J.M. Schneider, Ph.D.

ReproNim Fellow

What is Github?

GitHub is a code hosting platform for **version control** and **collaboration**.

It lets you and others work together on projects from anywhere.

Main Premise

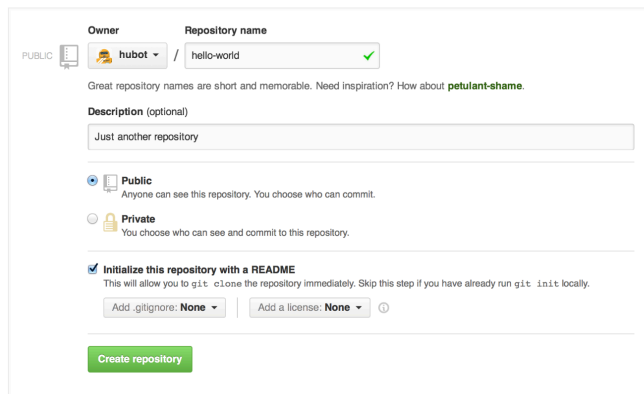
1. Make Changes



REPOSITORIES

CREATE YOUR OWN

1. In the upper right corner, next to your avatar or identicon, click + and then select **New repository**.
2. Name your repository `hello-world`.
3. Write a short description.
4. Select **Initialize this repository with a README**.



The screenshot shows the GitHub 'Create new repository' form. At the top, there are fields for 'Owner' (set to 'hubot') and 'Repository name' (set to 'hello-world' with a green checkmark). Below this is a note about repository names and a link to 'petulant-shame'. There is a 'Description (optional)' text area with the placeholder 'Just another repository'. Under the 'Visibility' section, 'Public' is selected with a radio button, and 'Private' is unselected. At the bottom, the checkbox 'Initialize this repository with a README' is checked. Below this checkbox are two dropdown menus: 'Add .gitignore: None' and 'Add a license: None'. A green 'Create repository' button is at the bottom.

WORK OFF SOMEONE ELSE'S

- **Forking a repository:**
 - act as a sort of bridge between the original repository and your personal copy.
- **Clone your fork:**
 - Clones a copy of the fork to your computer
 - This will put the repository onto your local computer to make changes to
`git clone <HTML link>`



Main Premise

1. Make Changes



2. Stage Changes



Staging the Change

- The purpose is to move your changes from your local computer to your git 'stage' where they can be committed
- `git add <filename>`
- `git add -A`
- `git ignore <filename>`
- `-rm`

Main Premise

1. Make Changes



2. Stage



3. Commit!



Committing

- The purpose of commit is like taking a “snapshot” of the repository as it stands now with the commit command
- It is the process which records changes in the repository and is done locally.
 - `git commit -m <"write what you did here.">`
 - `git commit`
 - If you don't include `-m`, `nano`, or whatever editor you use, will popup and let you edit a message for your commit

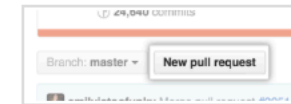
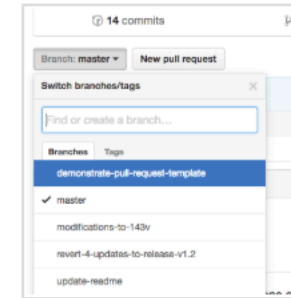
Pushing & Pulling



- **Pushing:** sends the recent commit history from your local repository up to GitHub.
 - If you're the only one working on a repository, pushing is fairly simple.
 - `git push`
 - If there are others accessing the repository, you may need to pull before you can push.
 - See **Troubleshooting** slide for help

Creating the pull request

- 1 On GitHub, navigate to the main page of the repository.
- 2 In the "Branch" menu, choose the branch that contains your commits.
- 3 To the right of the Branch menu, click **New pull request**.
- 4 Use the *base* branch dropdown menu to select the branch you'd like to merge your changes into, then use the *compare* branch drop-down menu to choose the topic branch you made your changes in.
- 5 Type a title and description for your pull request.
- 6 Click **Create pull request**.



es
d



Some Troubleshooting...

- Am I in the right folder?
 - `git status`
- Other people may be working on a similar file
 - `git remote pull upstream master`
 - This ensures that everything in your fork is up to date
 - `git remote add origin <main repo>`
 - This will only work if you have administrative power in the main repository though
 - Merge conflicts
 - Open file in nano (or whatever text editor), address changes, then begin again at add
 - `git status` can inform you of where the problem lies