My Github Guide

PREPARED BY

Animesh Kumar

CS Undergrad

Complete Setup of Github and Git

- mkdir and cd into new-dir
- git init // Converts the directory into repository
- git status //to check status
- Open github website and create a new repository without initializing with README or LICENSE.
- **Setup for password and automatic login:**
    - ssh-keygen -t rsa
        - Run this in cd ~ (home/AnimeshK) directory
        - Go on pressing ENTER for any prompt if comes
    - Go to github avatar-> Settings-> SSH and GPG ->new SSH Key
    - Copy the contents of ~/.ssh/id_rsa.pub file and paste into the *'key'* field of the form just opened on the website.
- In your folder create a README file and write to it, then add it to the staging index, then commit it also.
- Run this

  git remote add origin https://github.com/<username>/<new repo name>.git
- Then

  git remote set-url origin git+ssh://git@github.com/<username>/<new repo name>.git
- Running git remote show origin should show the git@github kind of fetch and push urls.
- git push -u origin <branch1> <branch2> .. // for all branches use --all flag instead of list of <branchi>'s.
    - This will make the listed branches to push to the github repo
- Next time onwards when you are done with all committing and want to reflect the changes on the github repo just run

```
git push
```

- Whenever you create a <new-branch> and want it to use git push, then do below(being at master):

```
git push -u origin <new-branch>
```

- As it turns out, git doesn't automatically push the 'tags' you have put locally to github repo, for that you have to run below command

```
git push --tags
```

Other Commands

- git <cmd name> --help // for help
- git branch // to list branches
- git branch <new branch> // to create <new branch>
- git branch -d <branch> // to delete <branch>
- git checkout <branch> // get into the <branch> as active branch
- git add <file1> <file2> // to stage modified files to staging index
- git commit // to commit finally (opens up the editor for commit message)
- git commit -m "<commit message>" // If the message is short: use this
- git log --oneline --all --graph // display the commit graph
- git log -p // shows the logs in detail in the current branch