**Important commands of Git**

**Config**

Setup config:

* **git config –global user.name “your name here”**
* **git config –global user.email “your email id”**

check config:

* **git config –global user.name**

update config:

* **git config –global –edit**

(above code will open vim editor, you can use setup config command too for overwriting user details, esc + : + w to quit vit)

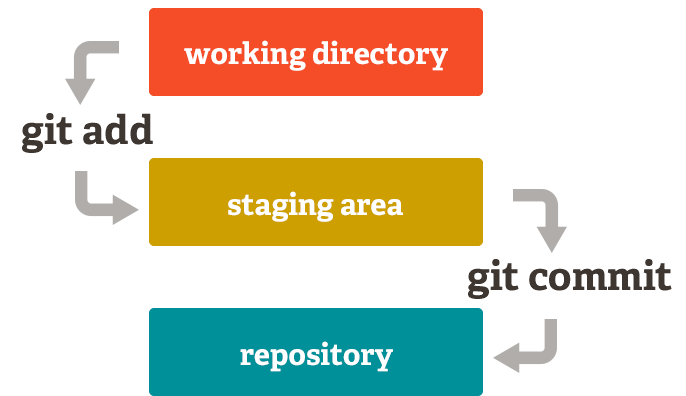
**Important commands**

* **git status** : to know whether it is git repo or not
* **git init** : to initialize as git repo (creates .git hidden file, use ls -a to see)
* **git add file\_name1, file\_name2….** : this will add these specific files

**git add .** : will add all untracked file

* **git commit -m “comments for commit”**

**git add** will make staging area for files/folders and **git commit** will make repository of that files/folders only.



* **git log** : provides details of all commit done by users
* git checkout <commit hash code/branch name> : to go back to any commits
* **git switch** - : to undo the checkout
* **git checkout master** : to go to master branch
* **git switch -c <new-branch-name>** : to create new branch name
* **git branch** : tells about how many branches are there
* **git checkout -b <new-branch-name>** : creates and checkout in one



* **git merge** <branch-name> : to merge branches together

Note: we start with master branch then we make dev branch for developers and then we make feature branch to assign specific feature to specific branch:

Example:

**Master** -> **dev** -> **ashish/addition**

* **adarsh/subration**
* **gaurav/division**

If we make any feature then it wont affect **dev** branch lets say, if **ashish/addition** is done and committed then it will not be preset on **dev** branch. We have to use **git merge ashish/addition** from dev to get addition feature on **dev** branch.

* **touch .gitignore** : this file is ignored by git

**GitHub**

**Push code on Github:**

1. **commit from local system:**
   * git init
   * git add .
   * git commit -m “comments form commit”
2. **Push to github**
   * git remote add origin {path -> url of your repo}
   * git branch -M master
   * git push -u origin master
3. **Push updated code to github**
   * Make changes
   * git add .
   * git commit -m “message for commit”
   * git push

**Note:** git push will throw an error “updated were rejected because the remote contains work that you do not have locally” This happens if you initialized a new github repo with **README** and/or **LICENSE file.**

* + git remote add origin [path to your github repo]
  + git pull origin master : to pull out those changes
  + git push origin master : to push your changes

**Open Source**

* Fork the repo : it will create copy of that repo in your github id
* git clone [url to repo created in your github account]
* make changes
* push changes to repo on your github
* create a pull request from that repo