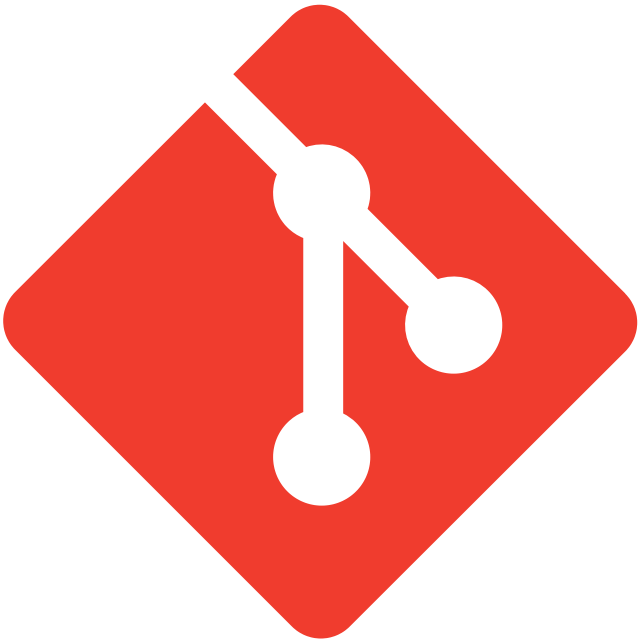

GIT COMMANDS

EVERY DEVELOPER MUST KNOW

&

HOW TO CONTRIBUTE TO A PROJECT

ARUN ARUNISTO

>>> Git config

To configure **git** in machine the developers using the **“git config”** command. This command is used to configure the git in the machine, like below:

By using the `**git config**` you can set your username and email, like below:

For email:

git config --global user.email "<your_email>"

And for username:

git config --global user.name "<your_username>"

And also you can view the configured username/email like below:

git config --get user.name

or

git config --get user.email

And also you can unset the username/email using the **--unset** action

git config --unset user.name

For more information about **config** use the below command:

git config --help

>>> Git add

"git add" is used to add files to the staging area from the working directory. You can use **"git add"** in few ways.

To add all the files you can use

git add .

The above command will add all the files from the working directory to the staging area.
To add a specific file to the staging area you can use

git add <filename>

>>> Git branch

To see the list of branches of your repository you can use the below command:

git branch

The above command will listout all the branch names of your repository.

>>> Git switch

"git switch" used to switching branches that you're working on and also you can create a branch using **"git switch"** command

For changing into an existing branch:

git switch <existing_branch_name>

For creating a new branch you should use:

git switch -c <new_branch_name>

>>> Git init

"git init" command used to initialize a local working directory as a git repository. For example if you're current working directory you want to change it as your git repository you can use the **"git init"** command.

```
git init
```

>>> Git clone

"git clone" command used to clone a repository into your machine. And you can clone repository using **"HTTPS"**, and **"SSH"** link also, like below:

```
git clone <https:link_of_the_repo>
```

>>> Git commit

"git commit" is used to commit a change is to store a version of your project in the git repository, and also you can mention the commit message to track the commits that you have done in your repository.

```
git commit -m "<message>"
```

>>> Git status

"git status" is used to see what you've done so far in your git reposiory, **"git status"**, which allows you to see what files have been added, deleted, or modified in your working directory. If you haven't addded any files it will return **"nothing to commit"**.

git status

>>> Git push

To get the changes you made into the remote repository, you need to push your changes to the remote repository. When you push your changes, you are essentially updating the remote repository with commits that have been made in your local repository.

git push -u origin <your_branch_name>

Here, **"origin"** refers to the **"remote repository"** and **"your_branch_name"** refers to the branch you're working on.

>>> Git show

"git show" allows you to see the changes that you've made on the branch that you're on. It's a great way to view the history of the project.

git show

>>> Git pull

"git pull" is an important command to run to keep your local repository up to date with the remote repository.

git pull

How to Contribute to a Project?

Suppose you're working for your company or on a community project with your team members or friends. How to contribute to that project that's what we are going to see next.

So, if they gave you access to the repo or it's a public repo first you need to **"fork"** the repository. But if they gave access to the **"SSH"** you don't need to do this you can simply clone your project you don't need to **"fork"**. But if you're using the **"HTTPS"** method for cloning the repo first you need to **"fork"** the repository. After forking the repo go to your repositories there you can see the repository that you **"fork"**.

After that copy the **"HTTPS"** clone link and clone the repo into your machine using the **"git clone"** command, for that you need to install **"git"** on your local machine and you need to configure your git account also. If you already installed leave it. So open the command prompt/terminal and type the below code to clone the repo:

```
git clone <repo_link>
```

This command will clone the repo directory into your local machine. Next, you need to change your directory to the cloned repo directory by using the **"cd"** command like below:

```
cd <repo_directory>
```

This will change your directory to the repo directory. And now you can see that on your terminal the default branch name of the repo. Always remember one thing when working with a git repo, it's important to add your own branch don't mess up with the original or the existing one. So before you start your work always remember to add a new branch of your own using the **"git switch"** command like the below:

```
git switch -c <new_branch>
```

Or if you want to add your work on the existing branch switch to the existing branch using the **"git switch"** command like the below:

```
git switch <existing_branch>
```

After switching to your branch you can start your work. After adding/changing the contents/files open your terminal and add the changes that you made on your local machine to the git repo using the **"git add"** command.

If you want to add all the changes that you made use the below command:

```
git add .
```

Otherwise, just add the filename like

```
git add <file_name>
```

After adding you need to commit the changes that you made always remember one thing don't add some random text to your commit message is a bad practice. Try to add some matching words about your work or what you had done on the repo it will be really helpful for the code reviewer.

Commit command be like:

git commit -m "<commit_message>"

After committing push the code from your local repo to the git repo using the below command:

git push -u origin <your_branch_name>

After pushing the changes you need to make a pull request using the **"git pull"** command, or you can directly send a pull request from your repositories dashboard of the Git Hub. Just go to the repo on your profile that you fork. You will see a button that mentions **"Compare & Pull request"**. If you click on that button it will navigate you to the **"Pull requests"** tab of your forked repo. On there you will see another button **"Merge pull request"**, click on that button it will send a pull request to the admin/manager who handles the repository. And they only accept your merge request once the review is done.

If they accept the request and merge your code you will get notified on your git hub profile. You will get a git badge named **"pull shark"** for making your first pull request that gets accepted by the admin/manager who owns the repo.