Basics of Github And Open Source Community

https://www.linkedin.com/in/nupurthakur27/

https://github.com/nupur-thakur

Nupur Thakur (B.Tech Computer Science 4th Year)
Open Source Contributor, Layer5 (https://layer5.io/)

Origin of Git

- Meaning of the word “GIT” from Cambridge Dictionary:

  Git : a person, especially a man, who is stupid or unpleasant

- Linus needed a new source code revision manager for Linux,and none of the available options in 2005 where good enough,so he wrote his own in. Kernel 2.6.12 was the first release managed by Git and version 1.0 of Git was released in December 2005.

Difference between Git and GitHub

Git is a distributed version control tool that can manage a development project's source code history, while GitHub is a cloud based platform built around the Git tool.
Git is a tool a developer installs locally on their computer, while GitHub is an online service that stores code pushed to it from computers running the Git tool.

Start a New Repository

Git init

Git add sample.txt

Git commit -m "First Message"

#Some Changes

Git commit -m "Second Message"

Typical Workflow

Creating a New Repository

```
git init
git add sample.txt
git commit -m “Initial Commit”
#some changes
git commit -m “Second Message”
```

Typical Workflow

```
git pull # edit files
git commit -am “Implemented X”
git push
git log
git show
```

How to Start Contributing in Open Source Communities.

Swag: T-shirt, stickers
Requirements: 4 pull requests in any repository
How to sign up: Hacktoberfest Website
Notes: All PRs count unless the repo has been marked spam, or the PR is marked as spam or invalid.

Introduction to hacktoberfest

Git Cheat Sheet

Remember!
`git <COMMAND> --help`

Global configuration is stored in ~/.gitconfig.
`git config --help`

master is the default development branch.
origin is the default upstream repository.

Create

From existing data
```
cd ~/my_project_directory
git init
git add .
```

From existing repository
```
git clone ~/existing_repo ~/new/repo
git clone git://host.org/project.git
git clone ssh://user@host.org/project.git
```

Show

Files changed in working directory
`git status`

Changes made to tracked files
`git diff`

What changed between ID1 and ID2
`git diff <ID1> <ID2>`

History of changes
`git log`

History of changes for file with diffs
`git log -p <FILE> <DIRECTORY>`

Who changed what and when in a file
`git blame <FILE>`

A commit identified by ID
`git show <ID>`

A specific file from a specific ID
`git show <ID>:<FILE>`

All local branches
`git branch`
star (*) marks the current branch

Revert

Return to the last commited state
`git reset --hard`
This cannot be undone!

Revert the last commit
`git revert HEAD`
Creates a new commit

Revert specific commit
`git revert <ID>`
Creates a new commit

Fix the last commit
`git commit -a --amend`
(after editing the broken files)

Checkout the ID version of a file
`git checkout <ID> <FILE>`

Update

Fetch latest changes from origin
`git fetch`
(this does not merge them)

Pull latest changes from origin
`git pull`
(does a fetch followed by a merge)

Apply a patch that someone sent you
```
git am -3 patch.mbox
```
In case of conflict, resolve the conflict and
`git am --resolved`

Publish

Commit all your local changes
`git commit -a`

Prepare a patch for other developers
`git format-patch origin`

Push changes to origin
`git push`

Make a version or milestone
`git tag v1.0`

Branch

Switch to a branch
`git checkout <BRANCH>`

Merge BRANCH1 into BRANCH2
```
git checkout <BRANCH2>
git merge <BRANCH1>
```

Create branch BRANCH based on HEAD
`git branch <BRANCH>`

Create branch BRANCH based on OTHER and switch to it
`git checkout -b <BRANCH> <OTHER>`

Delete branch BRANCH
`git branche -d <BRANCH>`

Resolve merge conflicts

View merge conflicts
`git diff`

View merge conflicts against base file
`git diff --base <FILE>`

View merge conflicts against your changes
`git diff --ours <FILE>`

View merge conflicts against other changes
`git diff --theirs <FILE>`

Discard a conflicting patch
```
git reset --hard
git rebase --skip
```

After resolving conflicts, merge with
```
git add <CONFLICTING_FILE>
git rebase --continue
```

Workflow