Let's GIT it started

Powered by, IEEE CS BMSIT Chapter, IEEE CS Bangalore Chapter and PowerMax University Relations – Dell EMC

Anand Jagadeesh

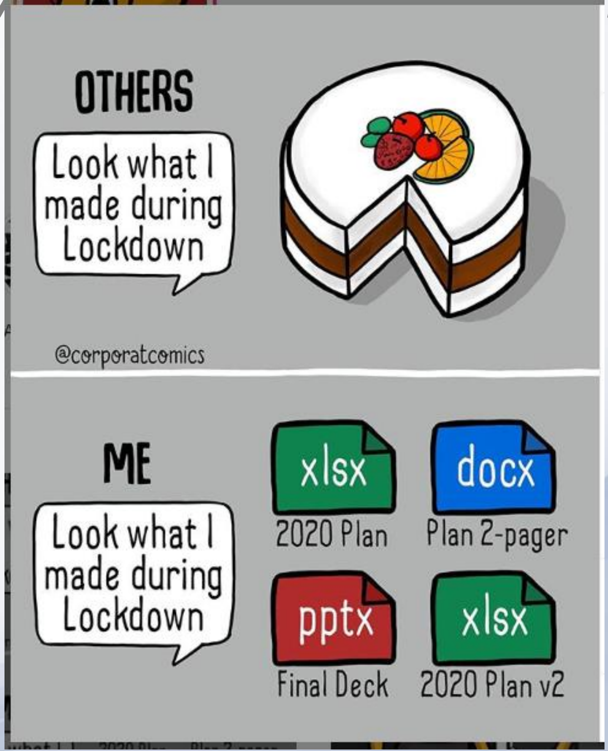
Secretary, IEEE Computer Society Bangalore Chapter

Software Engineer 1 at Dell EMC

GitKraken Ambassador

How’re you?

Lockdown Gives You More Time to Learn! Make Maximum Use of It!



About me

About me

- Anand Jagadeesh / Anand J.
- Software Engineer 1 in PowerMax Remote Replication team of PowerMax Replication and Cloud, Dell EMC (Dell Technologies)
- A Git user – Wouldn’t call myself an expert – Just a learner
- FOSS Enthusiast – Still using Windows – Don’t ask why ;D

The Git Story

Hmm… Wait… Before we Git it started

- How many documents, with the same name newer [illegible] do you have?

- VERSION CONTROL SYSTEMS or VCS

- Types:
  Local Only (FOS or Propr.): RCS – 1982, PVCS – 1985
  Client-Server (FOS or Propr.): CVS – 1986 and 1990(in C), SCM – 1970s
  Distributed (FOS or Propr.): Git – 2005, Plastic SCM – 2006 (Source: WikiPedia)

The Git Story

- Created by the man who created "Linux"
- The name?
    - *unpleasant person*
    - "the stupid content tracker"
    - Pronounceable – not a UNIX command
- 2005/07/11 – 0.99
- 2020/03/22 – 2.26
- Design is inspired by BitKeeper and Monotone
- Distributed
- HTTP/FTP/SSH

Let’s Git

The First Steps

Installing Git:

MacOS: just run `git –version`

Linux Users: `[sudo] apt get install git-all`

Windows: git-scm.com/downloads ---> Download and Install

I would suggest using SSH keys but let's skip for now

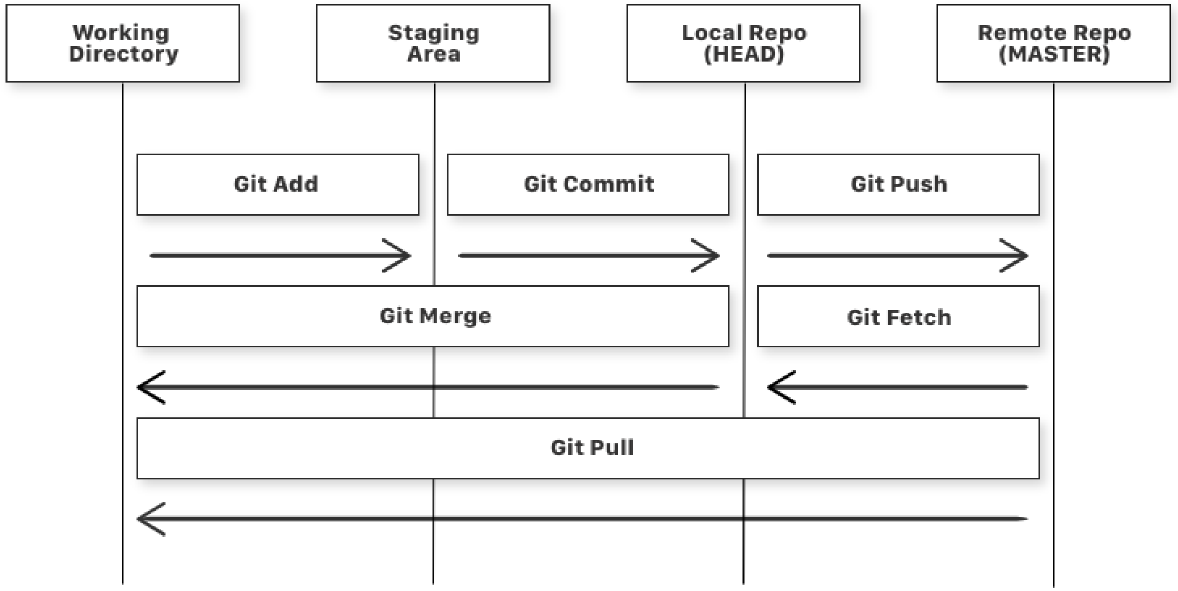
Telling git who you are is important. I suggest running the following commands:

```
git config --global user.name "myname"
git config --global user.email "me@myself.com"
```

`git config --global –list` to see what you just set

Once this is done, you can start using git.

To Git Workflow



Initialize

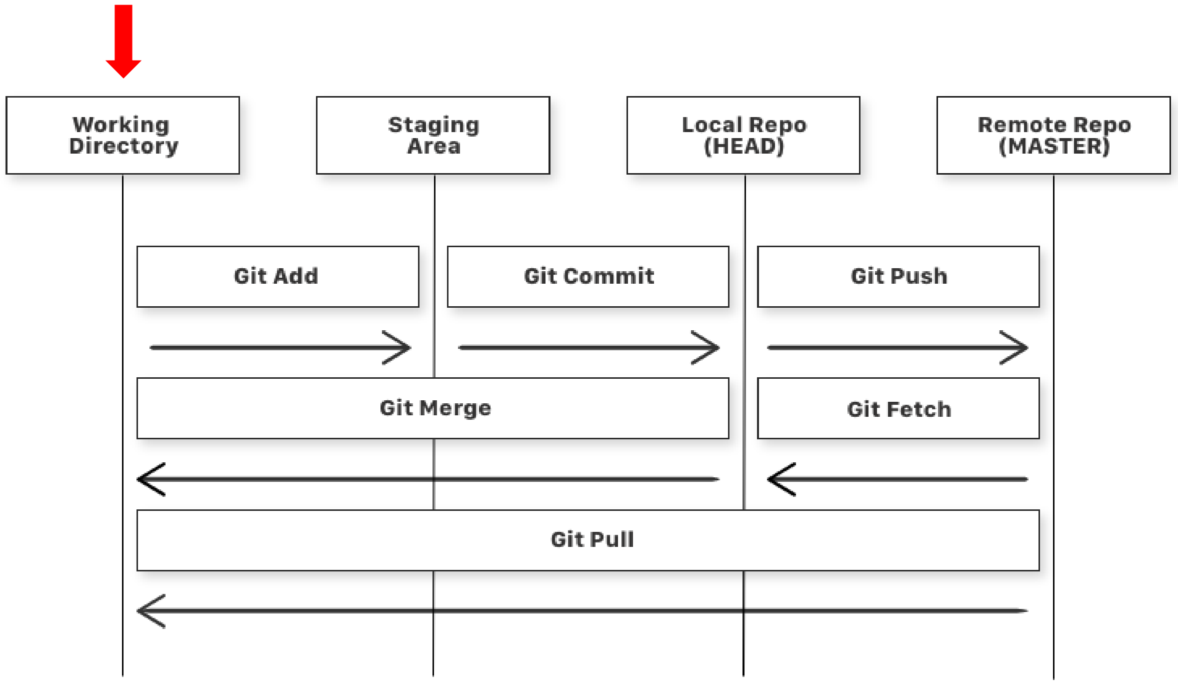
Create a directory/folder

Initializing a repo: `git init`

Add file(s) to the directory/folder

Now files are in the working directory

After Init and adding Files



Stage

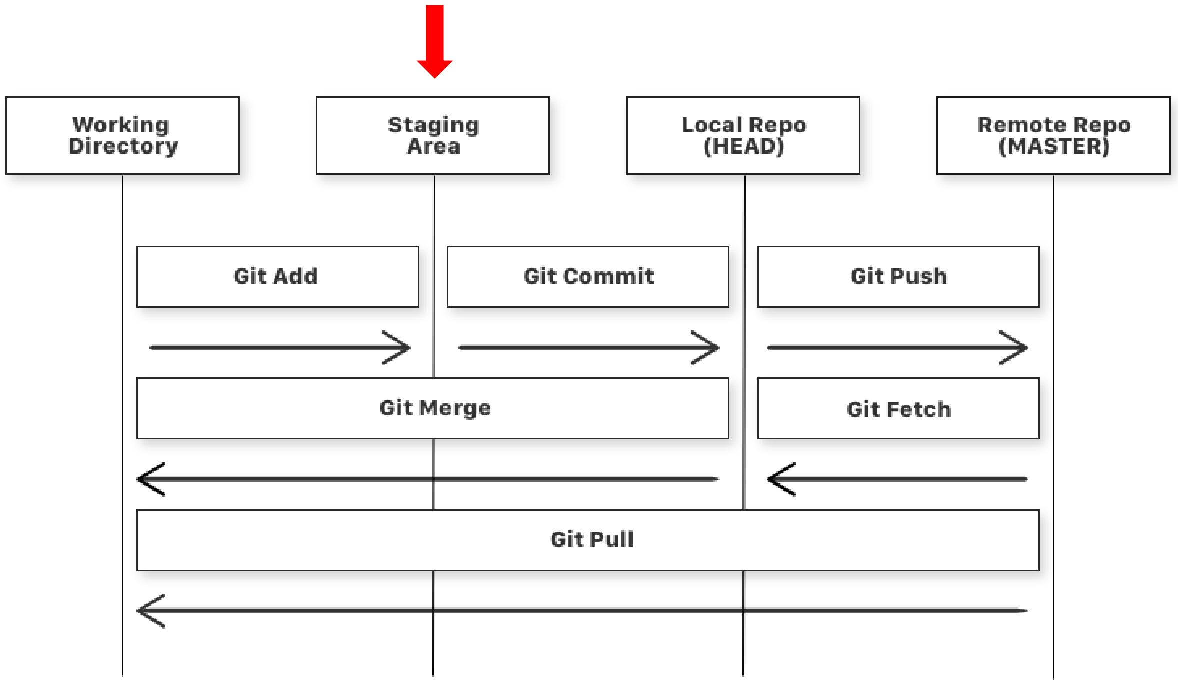
In terminal, enter

```
    git add <filename>
```

Or

```
    git add .
```

After Staging Files



Commit

Now files are *staged*

We just made it ready to be permed – like expression of interest

To make it perm, we need to commit the changes

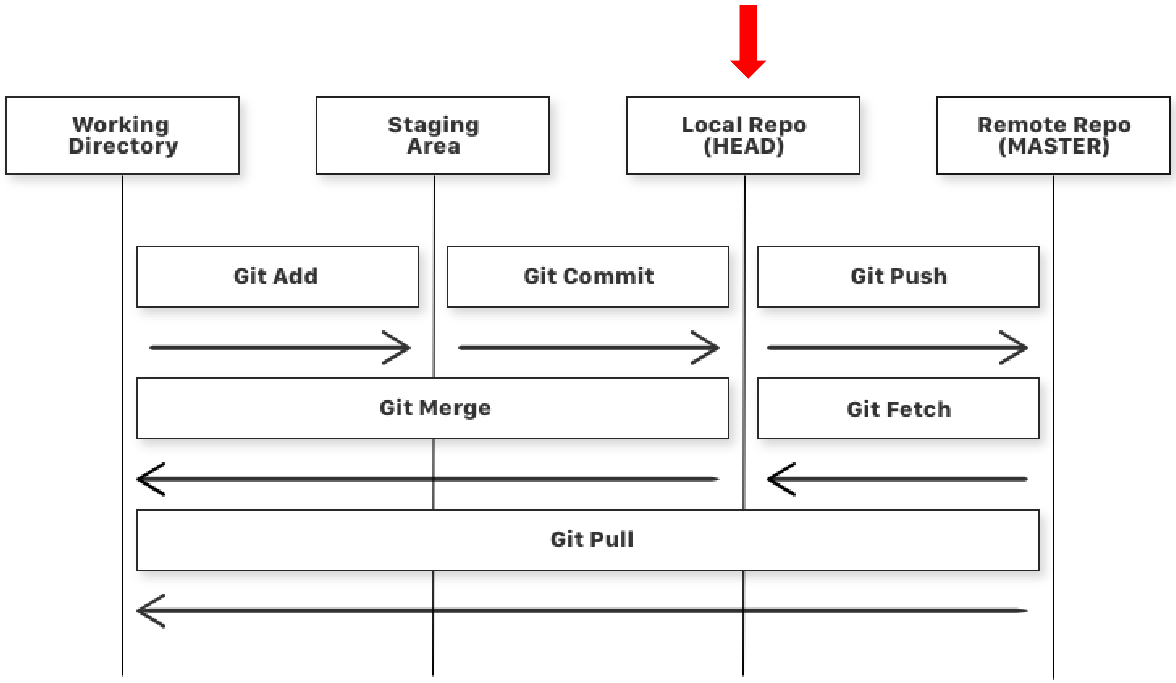
```
    git commit [-a] -m "my message"
```

Want to fix typos or piggy-back changes on last commit?

```
    git commit --amend
```

Now HEAD is your current commit

After Commit



Importance of Remote Repository

Now everything is in the local space – Just that changes are tracked

What if we loose the computer?

Or accidentally delete the folder?

Boom! You will be helically wound around an axis! (For those who didn’t understand this phrase, I’m sorry)

So, the solution is a “Remote Repository”

Remote Repository

Could be on-prem or enterprise or hosted elsewhere, privately

Or can use the services or products like GitHub, GitLab, BitBucket, etc.

Let’s try GitHub in these examples

Creating an Account on GitHub

Create an Account on GitHub at github.com/join

Once created, just create a new repository but, do not initialize it with a readme or anything.

Once repository is created, copy its URL : will show you

That’s your remote repository – Now let’s connect our local to remote

Let’s Connect Repos and Push!

Back to the terminal!

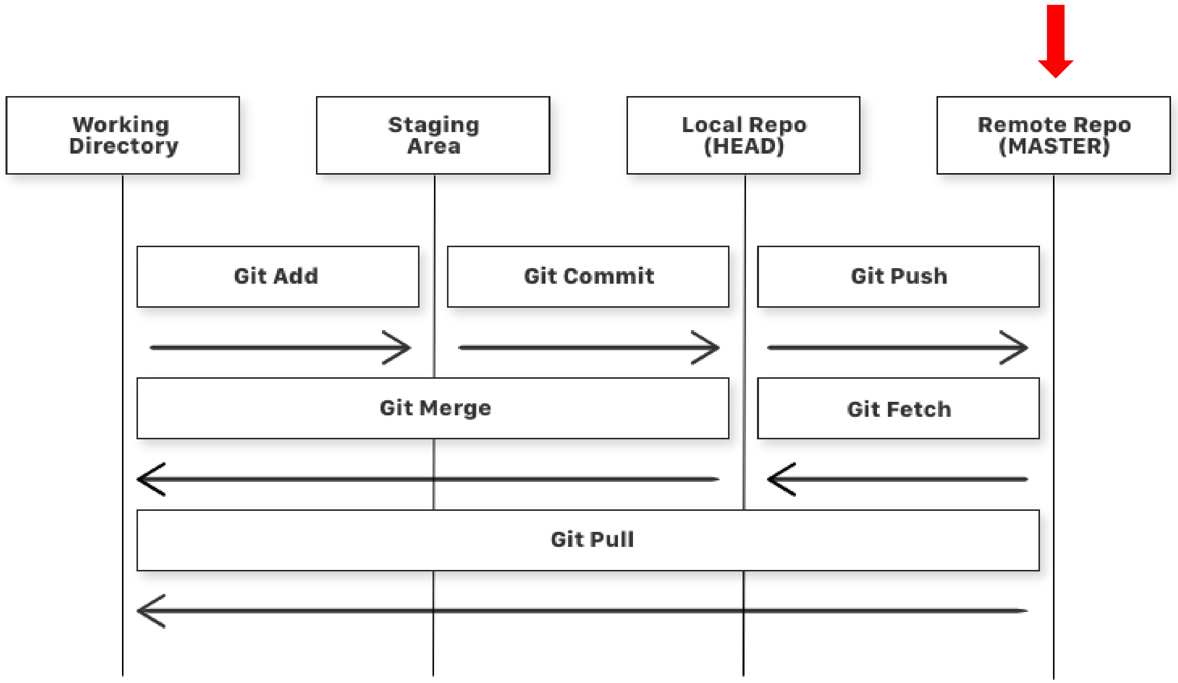
```
git remote add origin <the_copied_url>
```

Verify using: `git remote -v`

Push to remote: `git push origin master`

KaBoOm! Your Changes are in Remote now!

After Push!



Working Directory
Staging Area
Local Repo (HEAD)
Remote Repo (MASTER)
Git Add
Git Commit
Git Push
Git Merge
Git Fetch
Git Pull

Can we do it the other way around?

Yes sure!

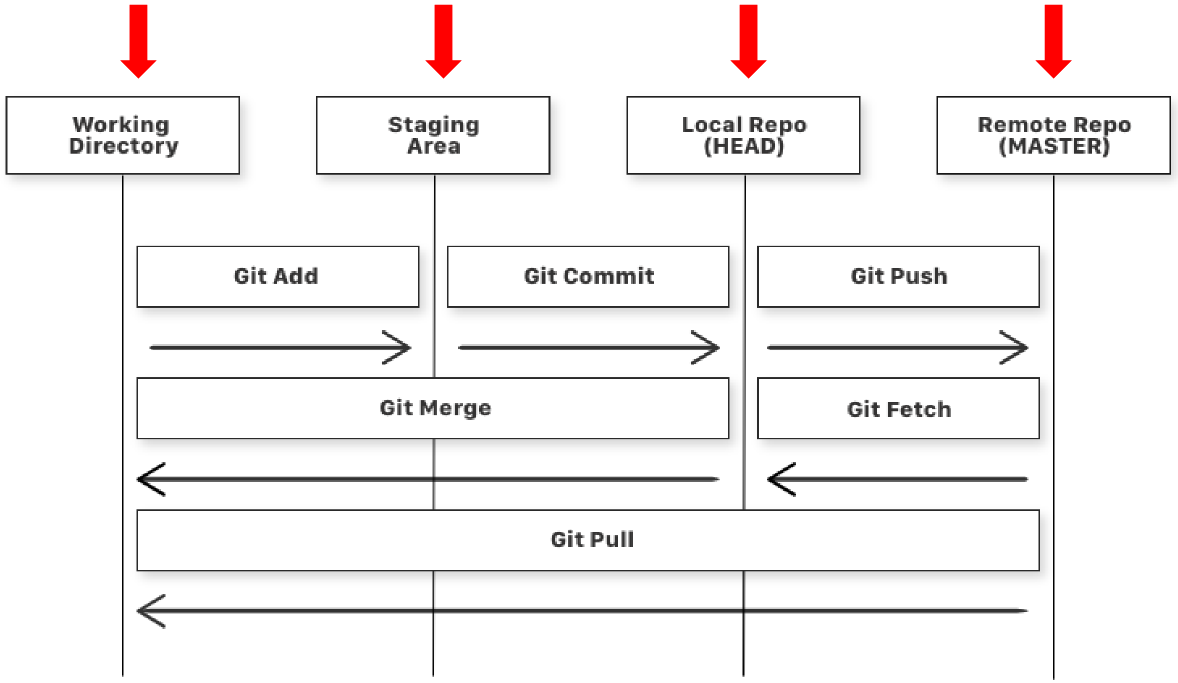
That’s Clone!

Create a repository on GitHub, initialize with a README and copy the link to repo

Now do

```
git clone <link_you_copied> ./<folder_you_created>
```

Now What?



Working Directory
Staging Area
Local Repo (HEAD)
Remote Repo (MASTER)
Git Add
Git Commit
Git Push
Git Merge
Git Fetch
Git Pull

Interesting Right?
But what if multiple people are working on a project?

Oh Man! Your requirements never end! :D

Actually its pretty simple! You need to learn a few spells!

Accio!

Say with me! Accio!

Fetch – Merge – Pull

I wasn't joking about spells! But it wasn't Accio!

Fetch is bringing the changes to local

```
git fetch [--all]
```

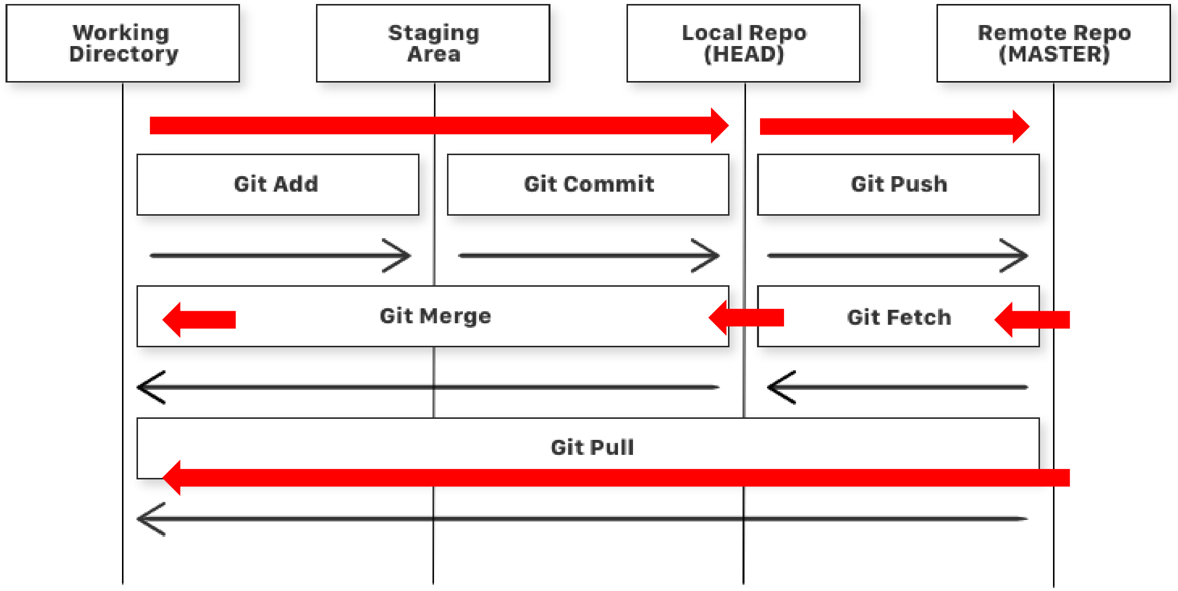
Merge is putting that changes into local

```
git merge
```

Then Pull? ---------> `git pull [origin] <branch_name>`

You have a new Message from Serious Black

After Push!


Working Directory
Staging Area
Local Repo (HEAD)
Remote Repo (MASTER)
Git Add
Git Commit
Git Push
Git Merge
Git Fetch
Git Pull

Git Branches

Different work on the same code can be handled

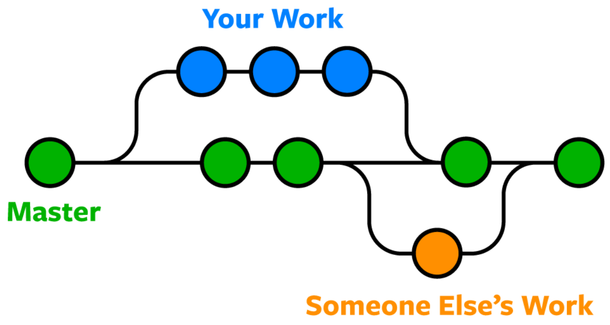
```
git branch [-r|-a]

git checkout [-b] <branch_name>

git push -u origin <branch_name>

git push origin --delete <branch_name>

git branch -D <branch_name>
```



What if I was making changes?

Hm.. Well, you have Stash and Unstash

But I think, stash, unstash, unstage, revert, reset, etc., etc., needs more time to explain.

So, let me jump into a tool instead of confusing you more!

The tool is called GitKraken

https://sl.anandj.xyz/GK ⇨

Intro To GitKraken

A Git Client

INTRODUCING MYSELF

Anand Jagadeesh

@_anandjagadeesh

#gitkrakenambassador

*I am a GitKraken Ambassador, **not a paid employee of GitKraken by Axosoft.***

GitKraken

I certainly know Git very well, and honestly think I’m far faster and more efficient in a Git GUI than I could possibly be in the command line – and I’m certainly not slow in the CLI

Dan Clarke

blogger and co-organizer of .Net Oxford group

The Solution - GUI

A common mantra for devs is “work smarter, not harder.”

There’s no reason GUIs should be the exception to this rule.

The Solution - GUI

A common mantra for devs is "work smarter, not harder."
There's no reason GUIs should be the exception to this rule.

Life is too short to still be using the command line. GUIs have won and are the way of the future!

Even the most die-hard CLI user will be more productive with a web UI in certain situations due to fundamental strengths of GUIs that ultimately are tied to traits inherent in humans

Micah Linnemeier

UX designer at IBM

FREE GITKRAKEN ACCOUNT

https://sl.anandj.xyz/GK

Use my referral URL to be entered to win Swag!

A demo on the GUI Tool

- Let’s see how a local repo is created and pushed to remote
- Also, let’s see how to clone a repo
- Rest of the stuff is for you to explore and learn

What else can you do?

- Create gist(s) on GitHub
- Create a Wiki for your project
- Create a simple personal website or a project website
- Advanced Git commands

Continuing this discussion

- Send me a Feedback at: **https://sl.anandj.xyz/feedback**
- Google Classroom Code: 2oacxmk
  - For Assignments to learn – No Certificates
  - For Chat Forums
  - You need an @gmail.com account to access this
- Mail: anand.j@ieee.org
- LinkedIn: linkedin.anandj.xyz
- codemind.anandj.xyz
- https://sl.anandj.xyz/GK

IEEE COMPUTER SOCIETY Bangalore Chapter

THANK YOU

OBRIGADO gracias どうも ARIGATO hvala DANK U 謝謝 takk MERSI merci obrigado danke schön köszi PALDIES muchas gracias ありがとう TEŞEKKÜR EDERIM MOLTE GRAZIE GO RAIBH MAITH AGAT danke

grazas GRAZZI THANKS qujan TAK благодаря asante благодаря TAK どうも muchas gracias vielen dank

PALDIES DANK U OBRIGADO mesi danke DANK U 감사합니다 köszi благодаря grazie DZIĘKI TACK Gràcies спасибо MULTUMESC 多謝 TEŞEKKÜR EDERIM muchas gracias obrigado NA GODE

IEEE COMPUTER SOCIETY Bangalore Chapter

IEEE