



Beginner's Guide to Git & Interactive Workshop

Git Gud

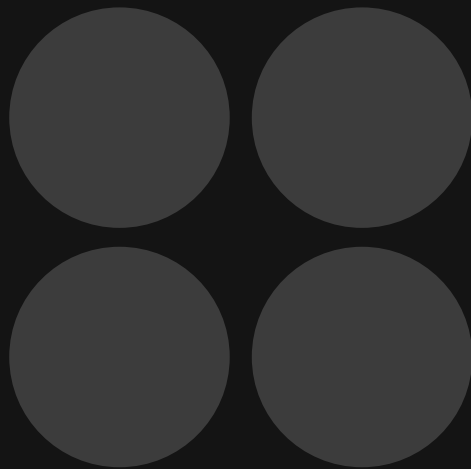




Table of contents

1/

SSH

Setup Git on your machine

2/

Git What?

VCS, Git Features, Git v. GitHub

3/

Git Why?

Why specifically Git?
IDE v. GUI v. CLI

4/

Git How?

Git Basics - Interactive

1/



Assuming you have Git installed:

> Open Git Bash

> Run following commands

```
$ ls -al ~/.ssh
```

> File that ends on .pub? Let me know.

> Otherwise:

```
$ ssh-keygen -t ed25519 -C "you@mail.sth"
```

```
$ cat ~/.ssh/id_ed25519.pub
```

> GitHub -> Settings -> SSH keys -> New SSH key

SSH

2/



VCS, Git Features,
Git v. GitHub

Git What?



- > Version Control System
- > Saves changes to files in repository
- > Clear development history
- > Easier to locate bugs, roll back
- > Alternatives to VCS?

VCS





- > Most Popular VCS
- > Thought up in 2005 by Linus Torvalds
- > Alternatives to Git?
 - > Mercurial (Facebook, Mozilla)
 - > SVN (LinkedIn)
- > Centralized v. Distributed
- > Repositories; Local and Remote

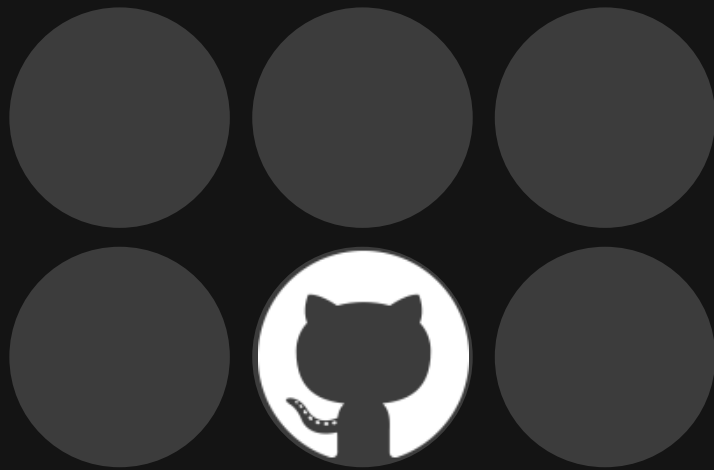
Git





- > GitHub; Git as a service
- > Alternatives?
 - > Bitbucket
 - > GitLab
 - > SourceForge
 - > Host your own
- > GitHub most popular
- > But.. Microsoft

Git v. GitHub



3/



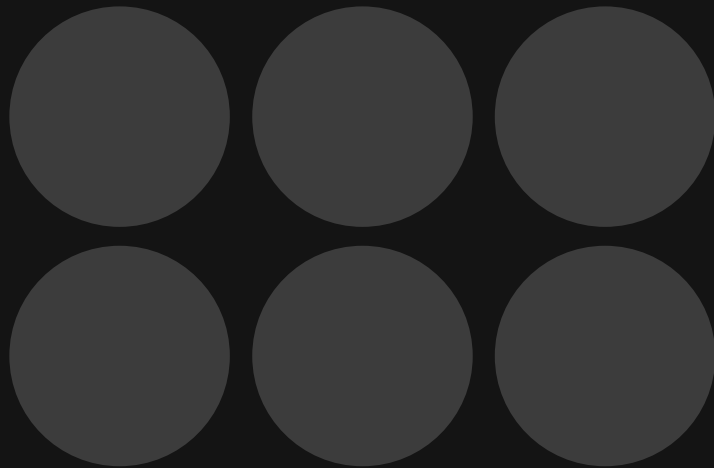
Why specifically Git?
IDE v. GUI v. CLI

Git Why?



- > Industry Standard
- > Git; Free, Open Source, Fast, Scalable
 - > Alternatives.. not so much
- > IDE v. GUI v. CLI

Yeah , why?



4/

- Git Basics

- Interactive portion

Git How?



- > Few ways..
 - > Easiest is starting on GitHub
 - > README, .gitignore, license
- > Local new repo
 - > git init
 - > Sync with remote (optional)

Making a Repository





Cloning

> Linked copy from remote to local

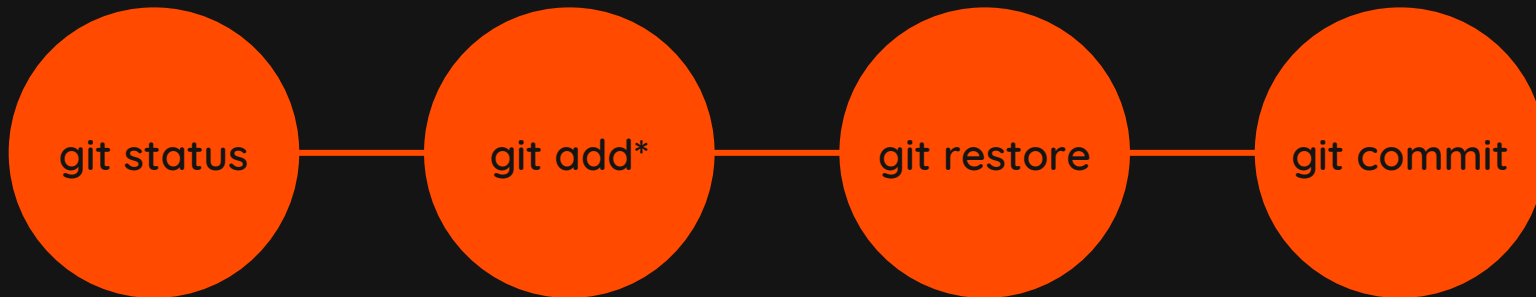


Forking

> Independent copy from remote to remote



Clone v. Fork



Status

Shows unstaged and untracked files

Add

Adds an unstaged and/or untracked file to staging area

Restore

Removes files from the staging area

Commit

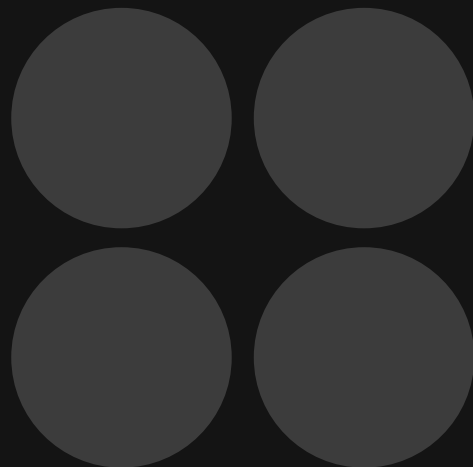
'Saves' the staging area to local repository



Basic Workflow

Git Version 2.x

Command	New Files	Modified Files	Deleted Files	Description
<code>git add -A</code>	✓	✓	✓	Stage all (new, modified, deleted) files
<code>git add .</code>	✓	✓	✓	Stage all (new, modified, deleted) files in current folder
<code>git add --ignore-removal .</code>	✓	✓	✗	Stage new and modified files only
<code>git add -u</code>	✗	✓	✓	Stage modified and deleted files only

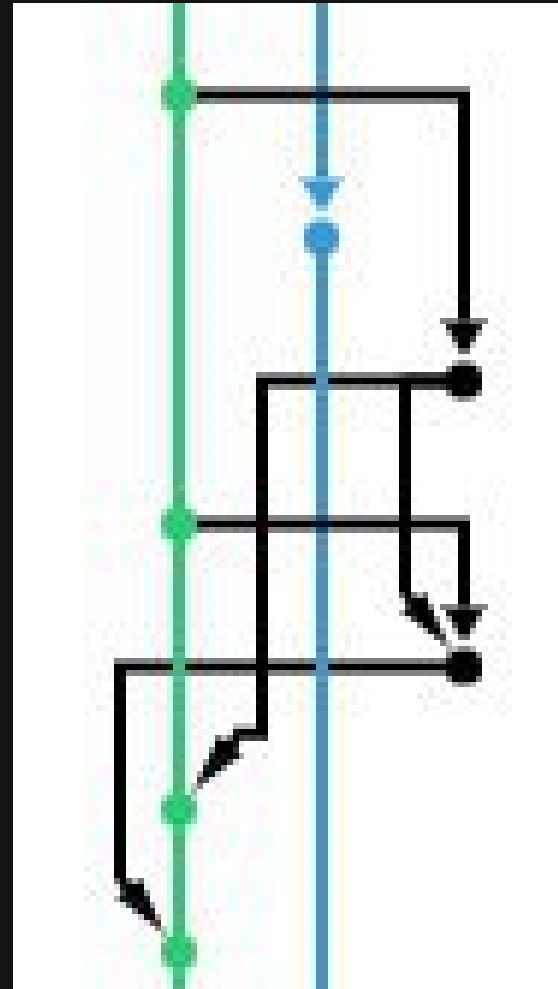


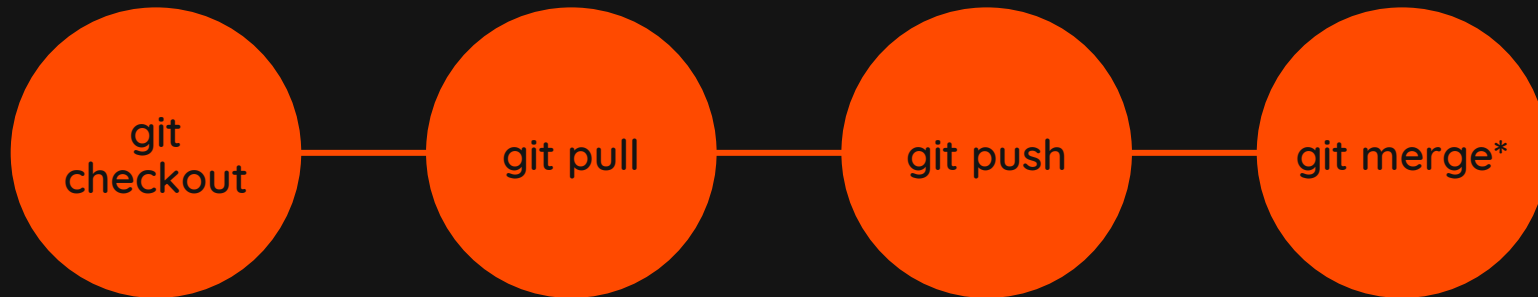
•
Git add



- > Branch what?
- > Branch why?
- > Branching strategies
 - > Features
 - > Personal
- > Branch how?

Branching





Checkout

Moves to a given branch, or creates it if it doesn't exist

Pull

Adds changes from remote into your local repo

Push

Pushes changes to local repo to remote repo

Merge

Combines two branches



Branch 'Workflow'

Make sure
dest is up
to date

Merge dest
into your
feature
branch

Push
changes

Merge
feature
branch into
dest

1/

Checkout to
destination repo
and pull

2/

Checkout to feature,
merge dest into
feature

3/

Fix merge conflicts,
repeat step 2 if
conflicts occurred

4/

Checkout to dest,
merge feature into
dest*



Branch 'Workflow'



> GitHub feature, not Git

> Merging into dev/main

> Code reviews

Pull requests



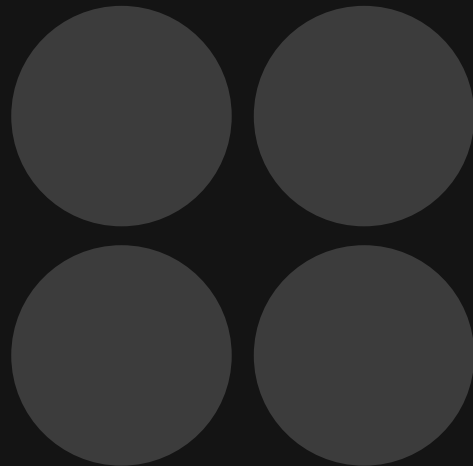


Next time; more advanced Git features!

Find this presentation at:
github.com/Druyv/GitGud



That 's all!





Do you have any questions?

youremail@freepik.com

+91 620 421 838

yourcompany.com

Please keep this slide for attribution

CREDITS: This presentation template was created by Slidesgo,
including icons by Flaticon, and infographics & images by
Freepik



Thanks !

