Git: An Introduction

Prashant K. Sharma
Diptesh Kanojia

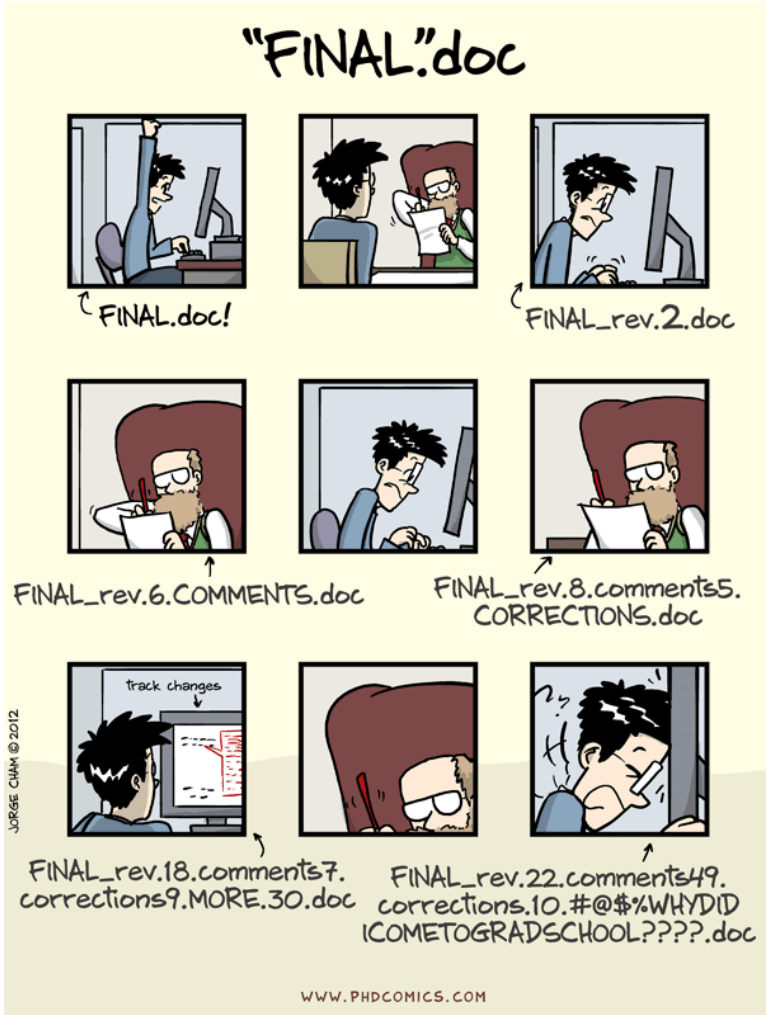
e-Yantra Team
ERTS Lab, IIT Bombay

IIT Bombay
April 17, 2021

Agenda for Discussion

1. Git: What and Why ?
   - Why Version Control ?
   - Git Overview
2. Git: How?
   - First Steps
   - Creating a repository
   - Adding files
   - Committing changes
   - Staging modifications
   - Move or Delete files
   - Git Life Cycle

Why Version Control ?



Introduction to Git

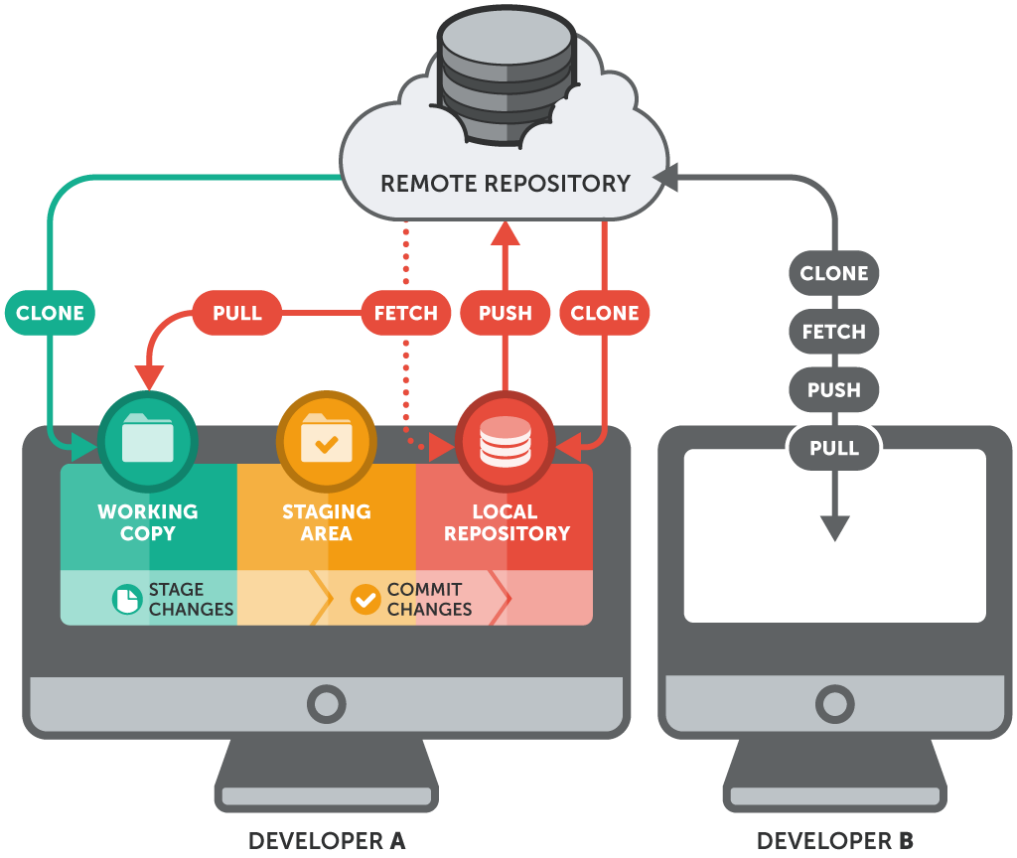


Figure 1: Git Overview

Git Configuration

To start using Git on your system for the first time, you will need to configure it. There are many things to configure, but to start we can just configure it with our name and email address.

Git Configuration

To start using Git on your system for the first time, you will need to configure it. There are many things to configure, but to start we can just configure it with our name and email address.

Bash

```
$ git config --global user.name "Vlad Dracula"
$ git config --global user.email "vlad@tran.sylvan.ia"
```

Figure 2: Git configuration

Creating a repository

A repository is nothing but a folder with git configuration, where you store all your work for a project for version control.

Creating a repository

A repository is nothing but a folder with git configuration, where you store all your work for a project for version control.

Create a directory with the name `super` and initialize it as a git repository.

Creating a repository

A repository is nothing but a folder with git configuration, where you store all your work for a project for version control.

Create a directory with the name `super` and initialize it as a git repository.

Run these commands on Bash

```
$ mkdir super

$ cd super
```

Creating a repository

A repository is nothing but a folder with git configuration, where you store all your work for a project for version control.

Create a directory with the name `super` and initialize it as a git repository.

Creating a repository

A repository is nothing but a folder with git configuration, where you store all your work for a project for version control.

Create a directory with the name `super` and initialize it as a git repository.

Run these commands on Bash

```
$ git init

$ ls -a
```

Adding files

We now have a Git repository. Now, we can start adding files and making changes so that all of it can be tracked using Git.

Adding files

We now have a Git repository. Now, we can start adding files and making changes so that all of it can be tracked using Git.

How do we add the file `players.csv` to our repository, so that git can start tracking our new file ?

Adding files

We now have a Git repository. Now, we can start adding files and making changes so that all of it can be tracked using Git.

How do we add the file `players.csv` to our repository, so that git can start tracking our new file ?

Use of **git add** command

```
$ git add players.csv
```

Adding files

We now have a Git repository. Now, we can start adding files and making changes so that all of it can be tracked using Git.

How do we add the file `players.csv` to our repository, so that git can start tracking our new file ?

Use of **git add** command

```
$ git add players.csv
```

How do we check the status of our repository ?

Use of **git status** command

```
$ git status
```

Committing changes

Once we have added the new file to our repository, we need to commit the changes so that Git can take a snapshot of the current state of the repository.

Committing changes

Once we have added the new file to our repository, we need to commit the changes so that Git can take a snapshot of the current state of the repository.

How do we commit the changes to our repository?

Committing changes

Once we have added the new file to our repository, we need to commit the changes so that Git can take a snapshot of the current state of the repository.

How do we commit the changes to our repository?

Use of **git commit** command

```
$ git commit -m "added players.csv"
```

Committing changes

Once we have added the new file to our repository, we need to commit the changes so that Git can take a snapshot of the current state of the repository.

How do we commit the changes to our repository?

Use of **git commit** command

```
$ git commit -m "added players.csv"
```

How do we check the status of our repository ?

Use of **git status** command

```
$ git status
```

Staging modified files

We can keep making changes to our repository by adding new files or by making modification to the existing files.

Staging modified files

We can keep making changes to our repository by adding new files or by making modification to the existing files.

Following commands will be used to track, check status and commit the changes that we are making to the repository.

Staging modified files

We can keep making changes to our repository by adding new files or by making modification to the existing files.

Following commands will be used to track, check status and commit the changes that we are making to the repository.

Key git commands so far

```
$ git add <file>

$ git status

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

Move or Delete files

To delete or move a file in your repository use **git rm** or **git mv** commands respectively

Move or Delete files

To delete or move a file in your repository use **git rm** or **git mv** commands respectively

What this does is, make the action of deletion or moving applied on Linux file system and stage the task (delete or move) for our next commit.

Move or Delete files

To delete or move a file in your repository use **git rm** or **git mv** commands respectively

What this does is, make the action of deletion or moving applied on Linux file system and stage the task (delete or move) for our next commit.

NOTE Make sure to commit after deletion to make changes to your repository.

Git Life Cycle

Each file in our repository can be in one of the four stages, listed below

Git Life Cycle

Each file in our repository can be in one of the four stages, listed below

- **Untracked** : It's not listed in the last commit
- **Unmodified**: It hasn't changed since the last commit
- **Modified** : It has changed since the last commit
- **Staged** : The changes will be recorded in the next commit made.

Git Life Cycle

We use different methods in git to move each file from one stage to another stage

Git Life Cycle

We use different methods in git to move each file from one stage to another stage

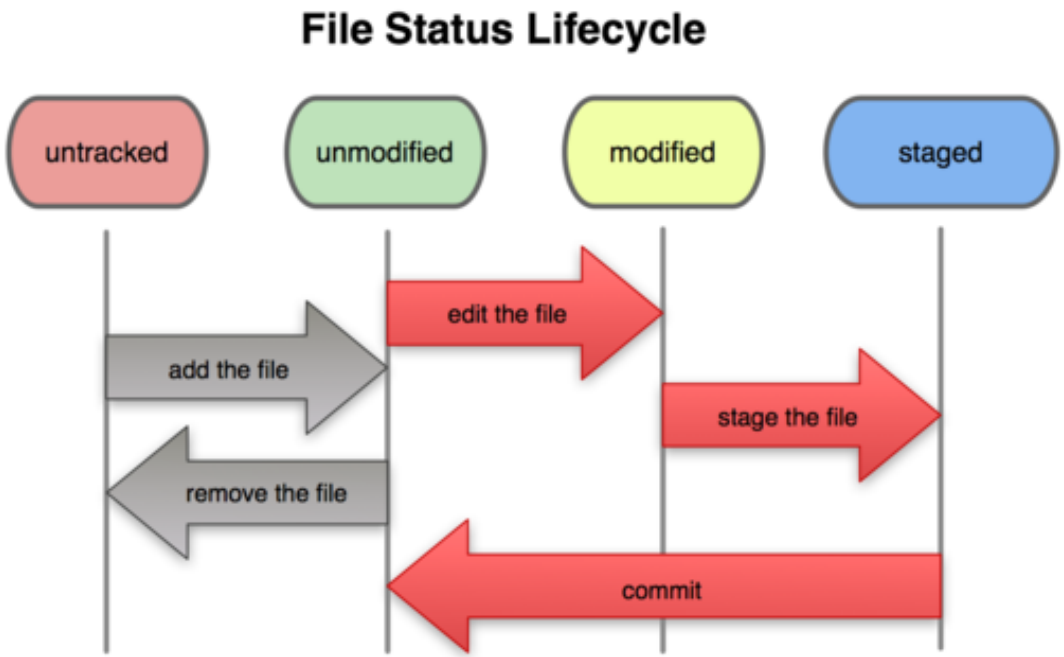


Figure 3: Credit: The Pro Git Book

References

- Git Cheat Sheet

Thank You!

Author: Prashant K. Sharma
Contributor: Diptesh Kanojia

Post your queries at: resources@e-yantra.org