iamneo

GitHub

What is Git

- Created by Linus Torvalds, April 2005
- Replacement for BitKeeper to manage Linux Kernel changes
- A command line utility
- Uses checksums to ensure data integrity
- Distributed Version Control Systems(DVCS)
- Cross-Platform (including Windows)
- Free Open Source Platform
- You can imagine git as something that sits on top of your file system and manipulates files

Version Control Systems

- Version control is a system that records changes to a file or set of files

- The following are the types of version control systems:
- 1. Local Version Control System
- 2. Centralized Version Control System
- 3. Distributed Version Control System

What is “Distributed Control System”

- Version Control System is a system that records changes to a file or set of files over time so that you can recall specific versions later

- Distributed means that there is no main server and all of the full history of the project is available once you cloned the project

Git distributed version control

- No need to connect to central server
- Can work without internet connection
- No single failure point
- Developers can work independently and merge their work later
- Every copy of a Git repository can serve either as the server or as a client
- Git tracks changes, not versions
- Bunch of little change sets floating around

Is Git for me?

- People primarily working with source code
- Anyone wanting to track edits (especially changes to text files)
- Review history of changes
- Anyone wanting to share, merge changes
- Anyone not afraid of command line tools
- You can imagine git as something that sits on top of your file system and manipulates files
- This “something” is a tree structure where each commit creates a new node in that tree
- Nearly all git commands actually serve to navigate on this tree and to manipulate it accordingly

Popular language use in Git

- HTML
- CSS
- Javascript
- Python
- ASP
- Scala
- Shell Scripts
- PHP
- Ruby
- Perl
- Java
- C

What is Repository

- “repo” = repository

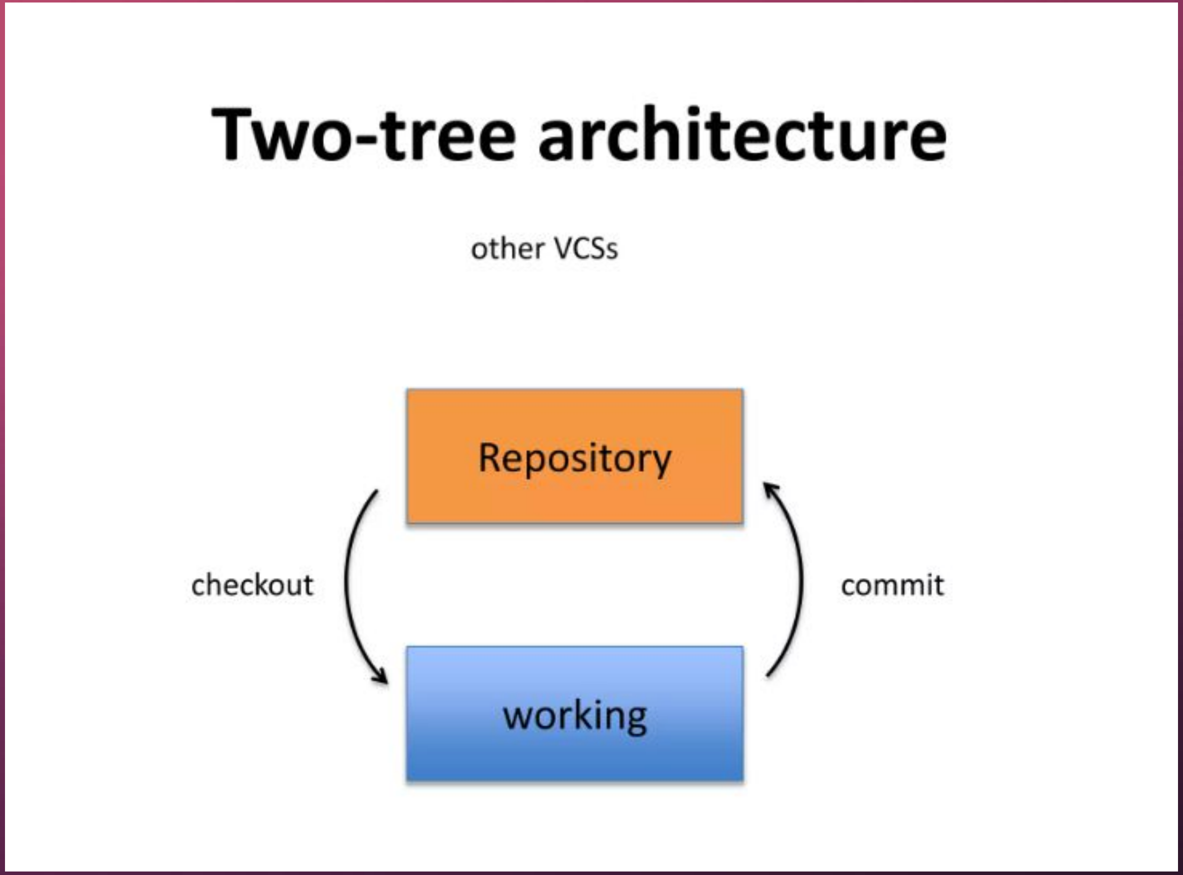
- Usually used to organize a single project

- The purpose of git is to manage a project, or a set of files, as they change over time. Git stores this information in a data structure called a repository

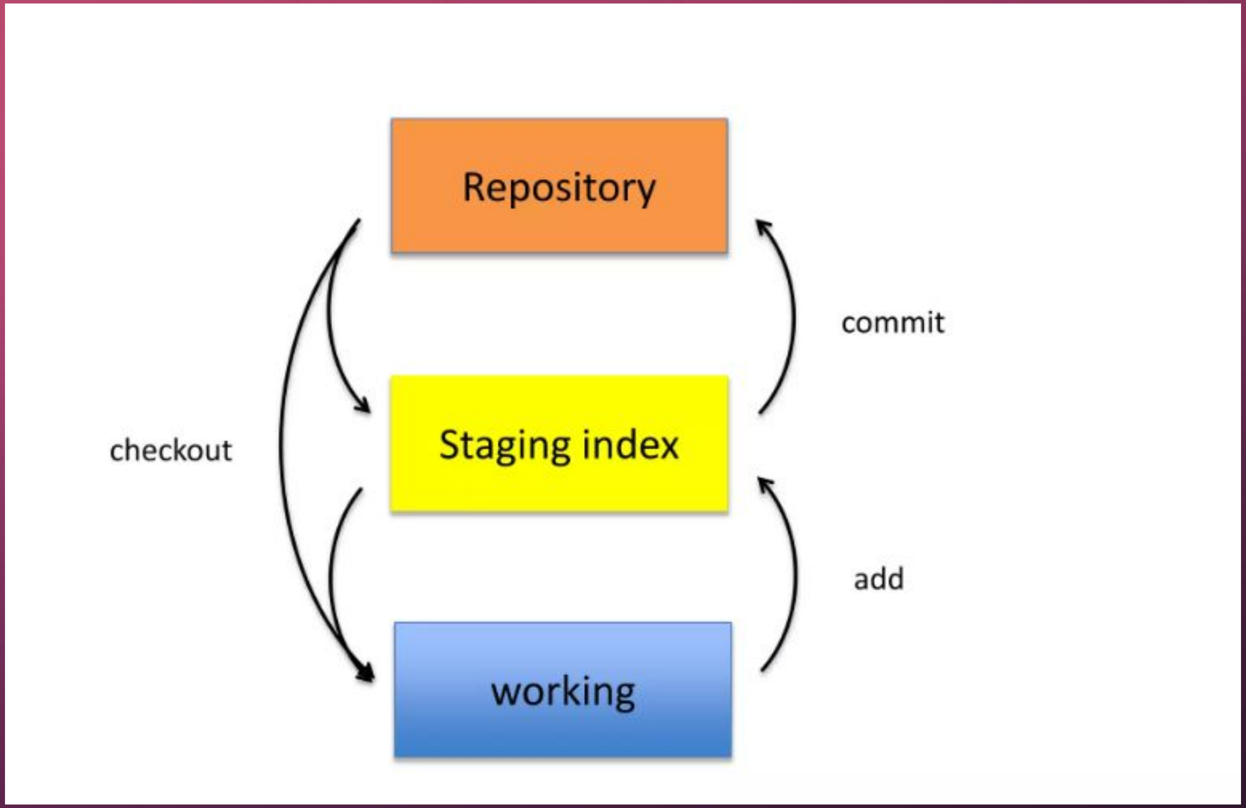
- A git repository contains, mainly a set of commits

- Repos can contain folders and files, images, videos, spreadsheets and data sets – anything your project needs

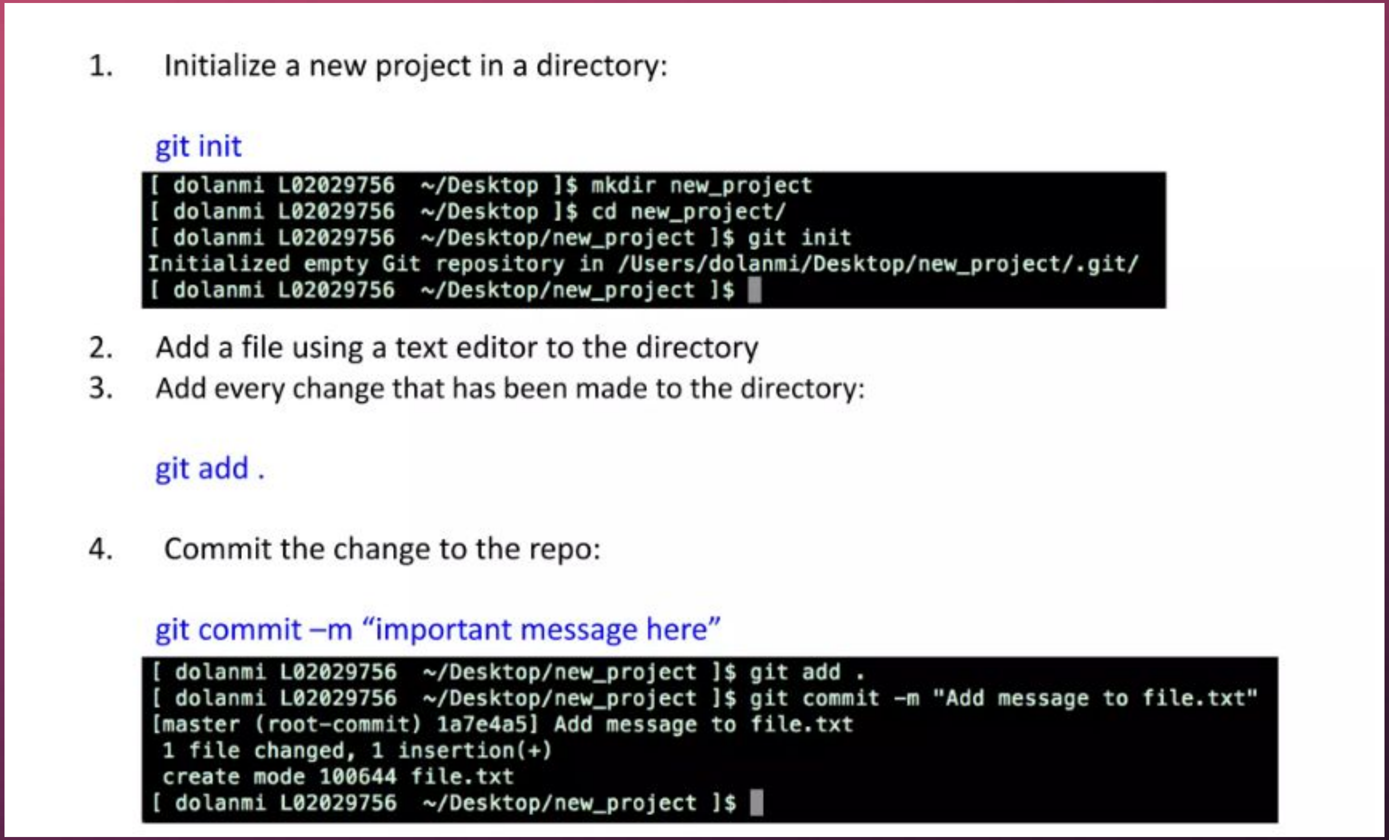
Two Tree Architecture – Other VCS's



Git uses a three-tree Architecture



A simple Git workflow



1. Initialize a new project in a directory:

   git init

```
[ dolanmi L02029756  ~/Desktop ]$ mkdir new_project
[ dolanmi L02029756  ~/Desktop ]$ cd new_project/
[ dolanmi L02029756  ~/Desktop/new_project ]$ git init
Initialized empty Git repository in /Users/dolanmi/Desktop/new_project/.git/
[ dolanmi L02029756  ~/Desktop/new_project ]$ █
```

2. Add a file using a text editor to the directory
3. Add every change that has been made to the directory:

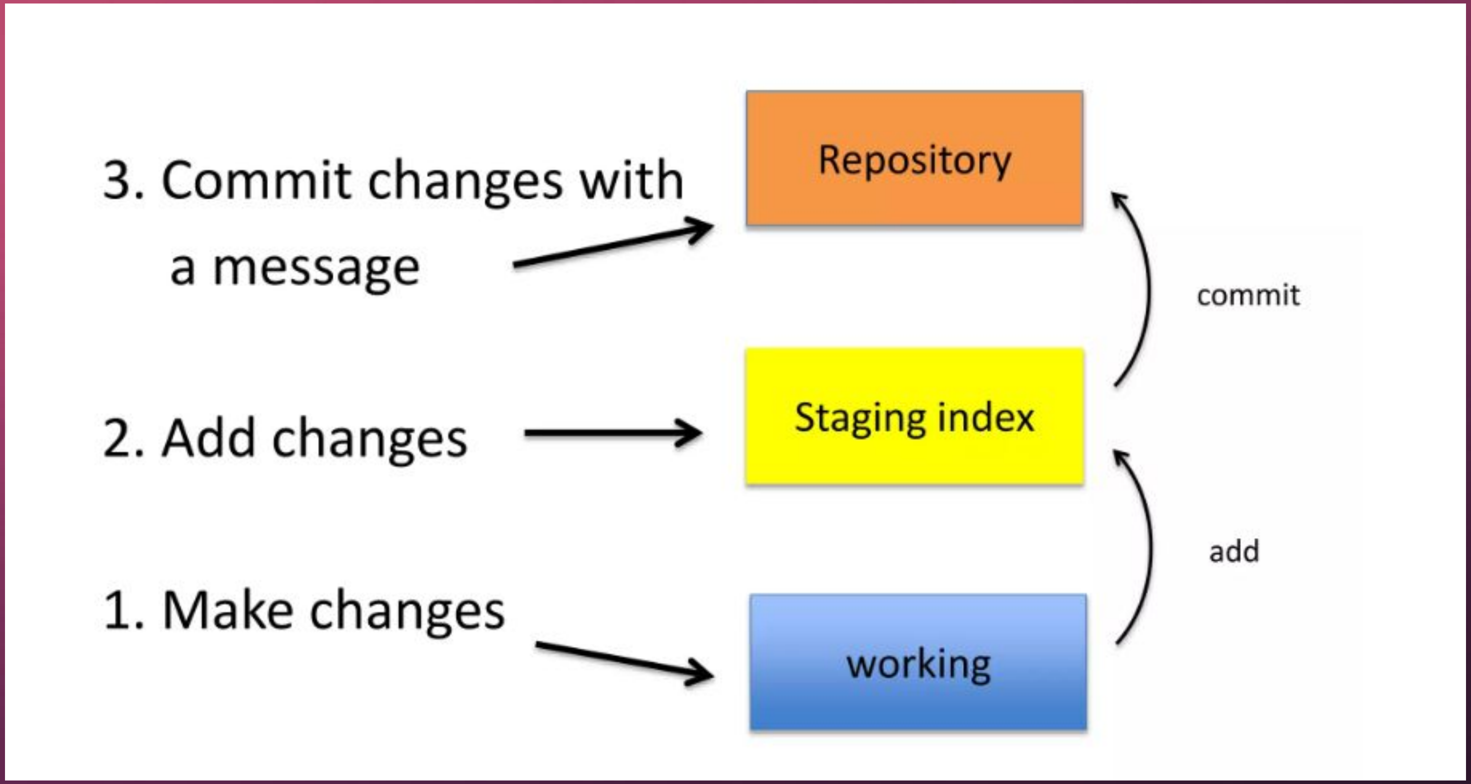
   git add .

4. Commit the change to the repo:

   git commit –m “important message here”

```
[ dolanmi L02029756  ~/Desktop/new_project ]$ git add .
[ dolanmi L02029756  ~/Desktop/new_project ]$ git commit -m "Add message to file.txt"
[master (root-commit) 1a7e4a5] Add message to file.txt
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
[ dolanmi L02029756  ~/Desktop/new_project ]$ █
```

After initializing a new git repo…



A note about commit messages

- Tell what it does (present tense)

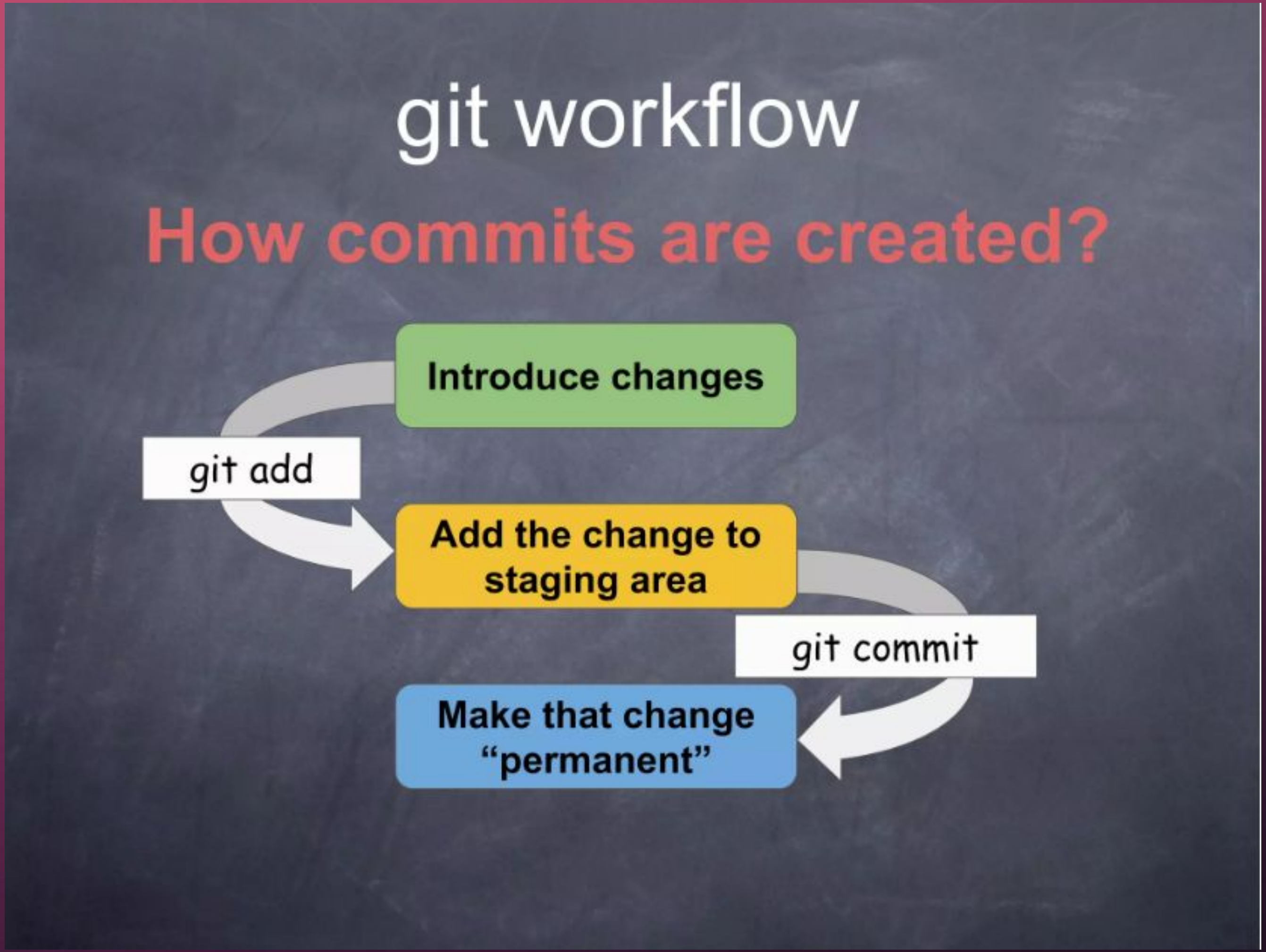
- Single line summary followed by blank space followed by more complete description

- Keep lines to <=72 characters

- Ticket or bug number helps

- A commit object mainly contains three things
- 1. A set of changes the commit introduces
- 2. Commit messages describing the changes
- 3. A hash, a 40-character string that uniquely identifies the commit object

Git workflow

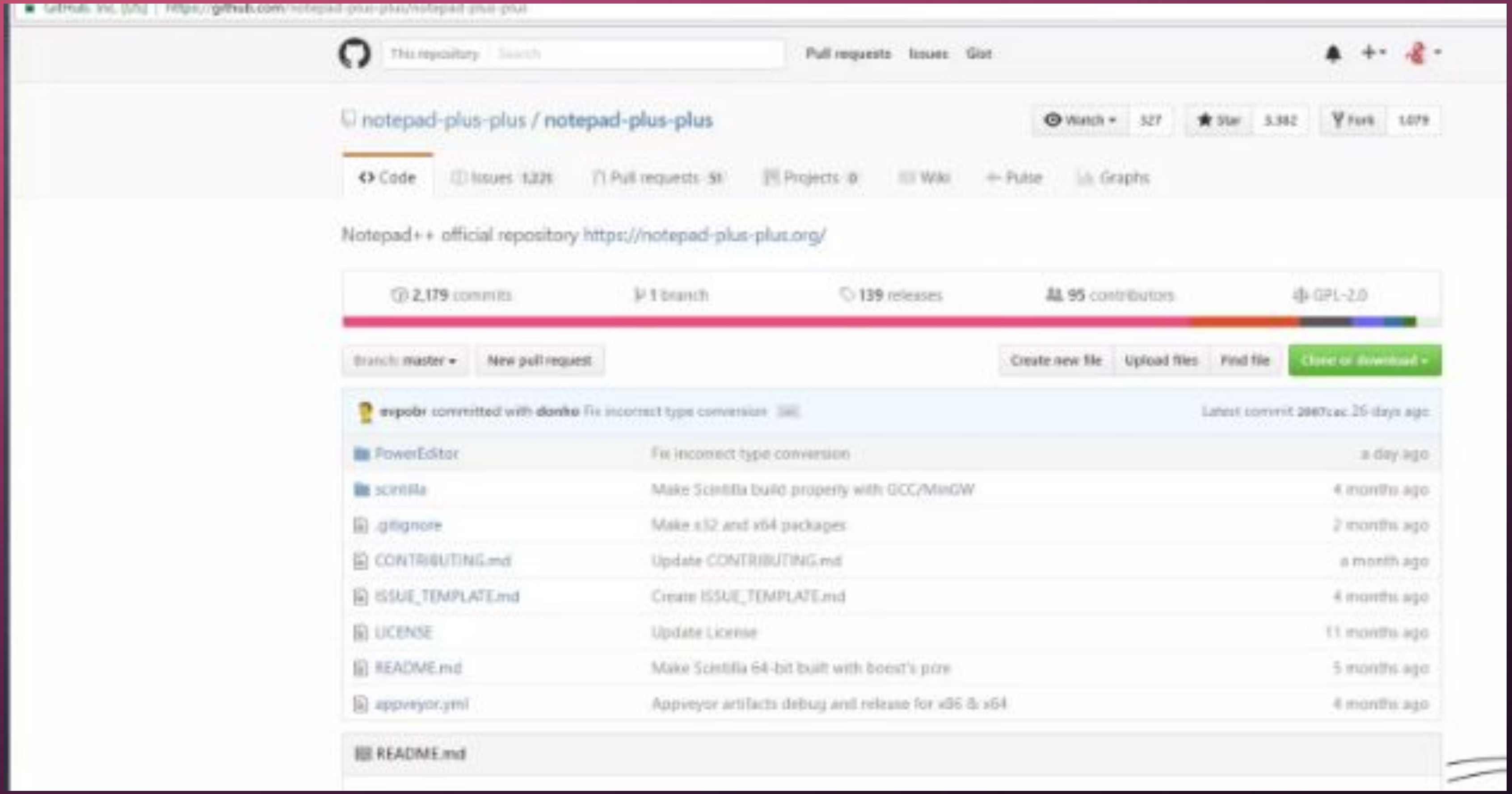


The three steps of git

- Introduce a change : introduce a change to a file that is being tracked by git
- Add the actual change to staging area : Add the change you actually want using “git add”
- Commit: Commit the change that has been added using git commit.

Github

- Git hub is a web based Git repository hosting services



notepad-plus-plus / notepad-plus-plus

Notepad++ official repository https://notepad-plus-plus.org/

2,179 commits | 1 branch | 139 releases | 95 contributors | GPL-2.0

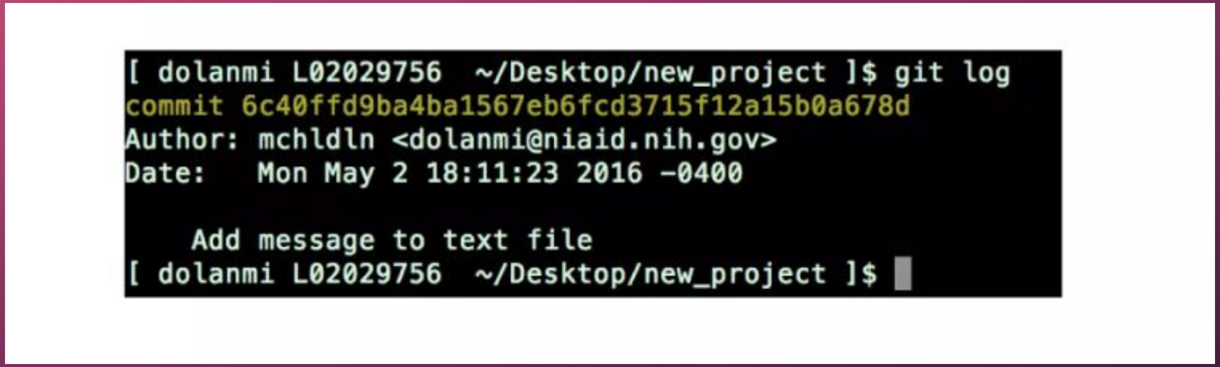
evpobr committed with donho Fix incorrect type conversion — Latest commit 2007cac 26 days ago

PowerEditor	Fix incorrect type conversion	a day ago
scintilla	Make Scintilla build properly with GCC/MinGW	4 months ago
.gitignore	Make x32 and x64 packages	2 months ago
CONTRIBUTING.md	Update CONTRIBUTING.md	a month ago
ISSUE_TEMPLATE.md	Create ISSUE_TEMPLATE.md	4 months ago
LICENSE	Update License	11 months ago
README.md	Make Scintilla 64-bit built with boost's pcre	5 months ago
appveyor.yml	Appveyor artifacts debug and release for x86 & x64	4 months ago

README.md

How do I see what was done

- Git log



```
[ dolanmi L02029756  ~/Desktop/new_project ]$ git log
commit 6c40ffd9ba4ba1567eb6fcd3715f12a15b0a678d
Author: mchldln <dolanmi@niaid.nih.gov>
Date:   Mon May 2 18:11:23 2016 -0400

    Add message to text file
[ dolanmi L02029756  ~/Desktop/new_project ]$
```

The HEAD Pointer

- Points to a specific commit in repo
- As new commits are made, the pointer changes
- HEAD always points to the “tip” of the currently checked-out branch in the repo
- Not to the working directory or staging index
- Last stage of repo (what was checked out initially)
- HEAD points to parent of next commit(where writing the next commit takes place)

- Git status - allows one to see where files are in the three tree schema
- Git diff – compares changes to files between repo and working directory
- Git rm filename.txt – moves deleted file change to staging area
- Git mv filename1.txt filename2.txt – Moving or renaming files

Frequently used commands

- Git init
- Git status
- Git log
- Git add
- Git commit
- Git diff
- Git rm
- Git mv

Git init

- Creates a new git repository
- Can be used to convert an existing, under versioned project to a git repository or initialize a new empty repository

Git Clone

- Copies an exiting git repository

Git Log

- Shows the commit logs

Git Add

- Adds Changes

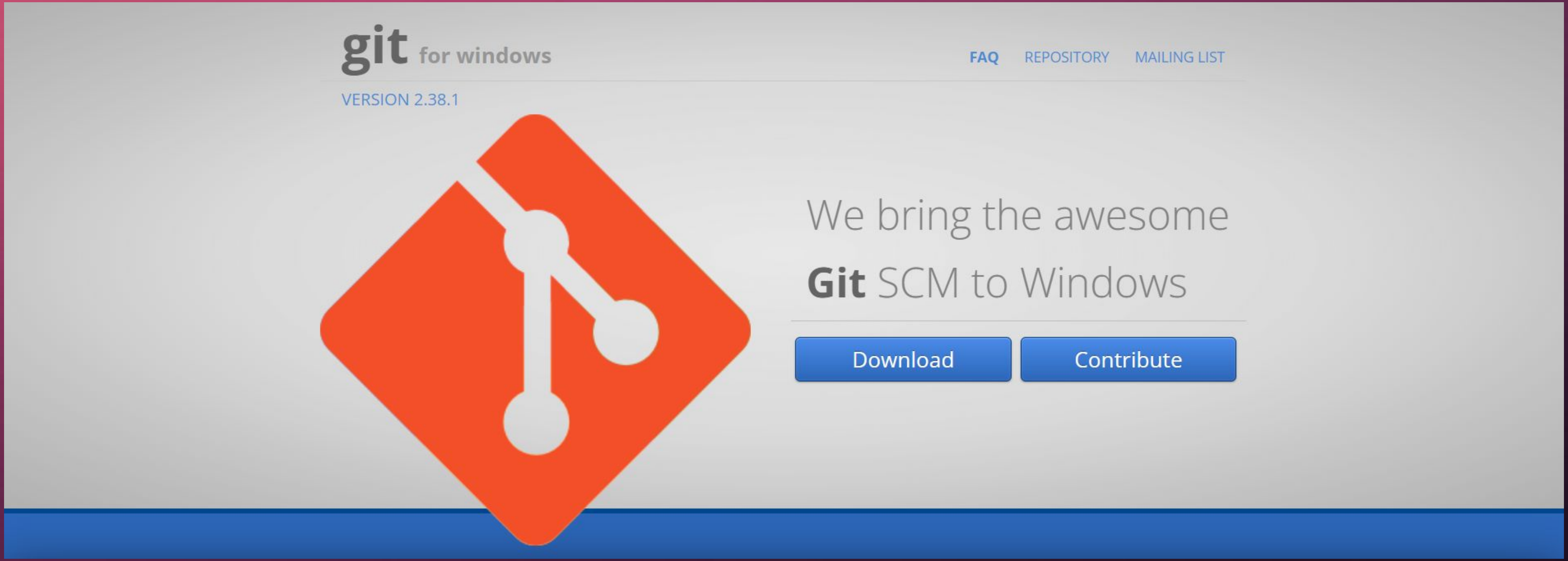
Git diff

- Displays the change that was introduced

- Useful flag:
- --cached:
- Displays the change that was added using “git add”

Install Github

- https://gitforwindows.org/
- Click on to Download button



We bring the awesome Git SCM to Windows

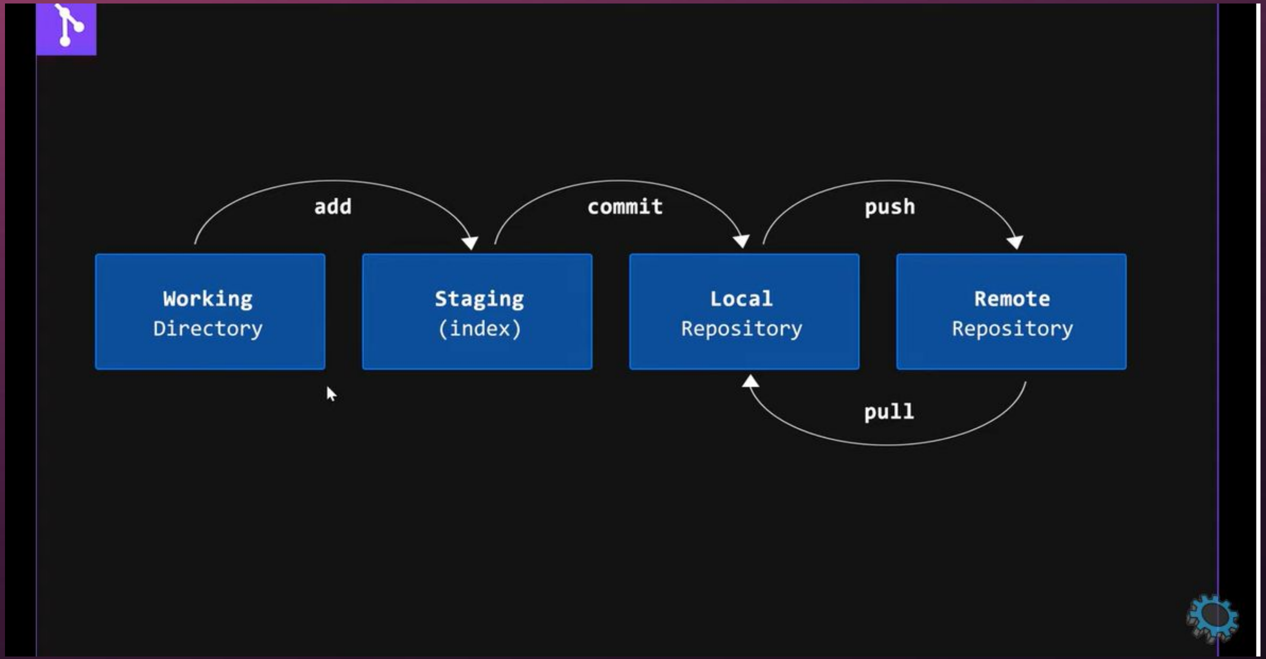
Github Account

- https://github.com/
- Create your account

iamneo

To Create Git in Local Repository

- Step 1: $ cd c:\localhost
- Step 2 : $ mkdir my-new-repository
- Step 3: $ cd my-new-repository
- Step 4: $ git init
- Step 5: $ touch my-new-filename.text (Creating a new file in the repository)
- Step 6 : $ git status



- Step 6: $ git add my-new-filename.txt
- Step 7: $ git status
- Step 8: $ git commit –m ”create a new file my-new-filename.txt” (commit the changes to the local repository)
- Step 9: $ git log(to see the file added)
- Step 10: $ git checkout –b “my-new-branch” (create a new branch. Branch is adding feature without affecting the main project
- Step 11: $ git branch(to see which branch we are in)
- Step 12: $ touch myfile-branched.txt (create a new file in the new branch)
- Step 13: $ git add myfile-brancehed.txt(add the file to the new branch)
- Step 14: $ git commit –m “Create a new file mufile-branched.txt”
- Step 15: $ git status

- Step 16 : $ git log (changes shows the log history)
- Merge the new branched file with the master file
- Step 17: $ git checkout
- Step 18: $ git branch
- Step 19: $ git merge my-new-branch(merged the changes to the main file)

GitHub Account

- Step 1: Click on new Repository
- Step 2: Give Name for the Repository
- Step 3: Copy the push command from the github repository to the local host
- Step 4: git push –u origin master
- Step 5: Check the repository for the files