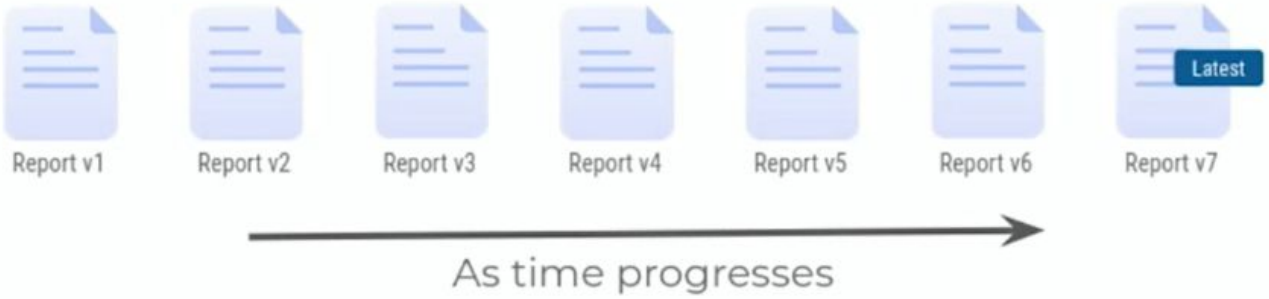
Git

Managing Versions of a Document

We keep making many changes in while building a software or even writing anything.

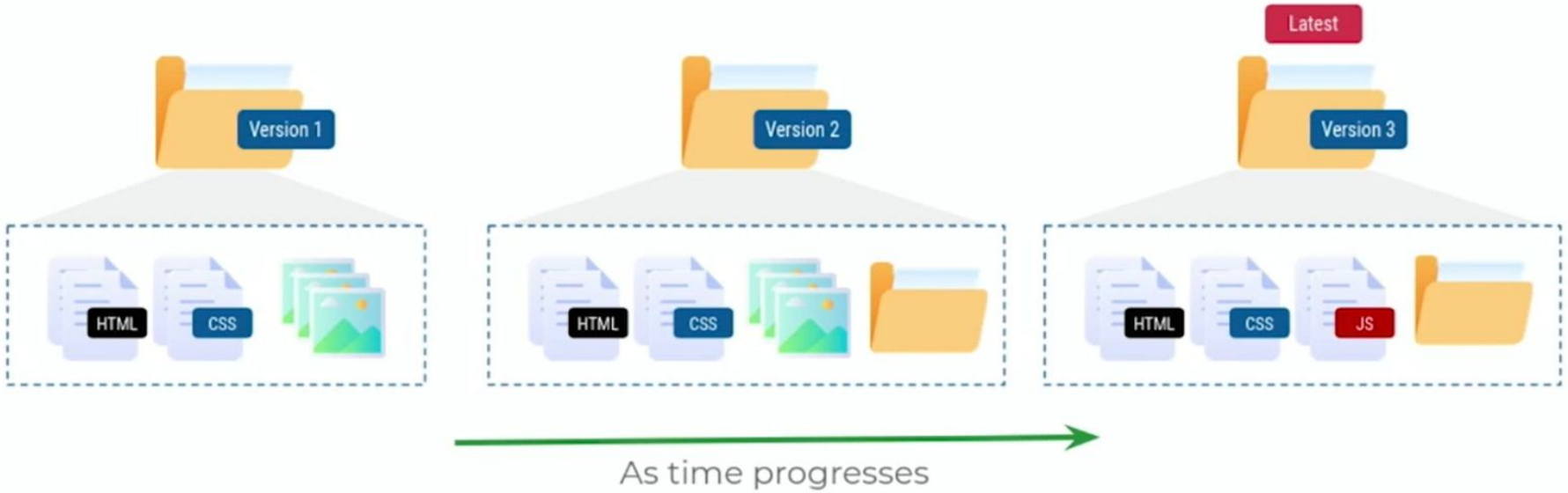


It is a ~~very~~ long established ~~point~~ fact that a ~~person~~ reader will be distracted by the readable content of a page when looking at its layout. ~~The reason why we use a placeholder text is~~ The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters, as opposed to using 'Content here, content here',

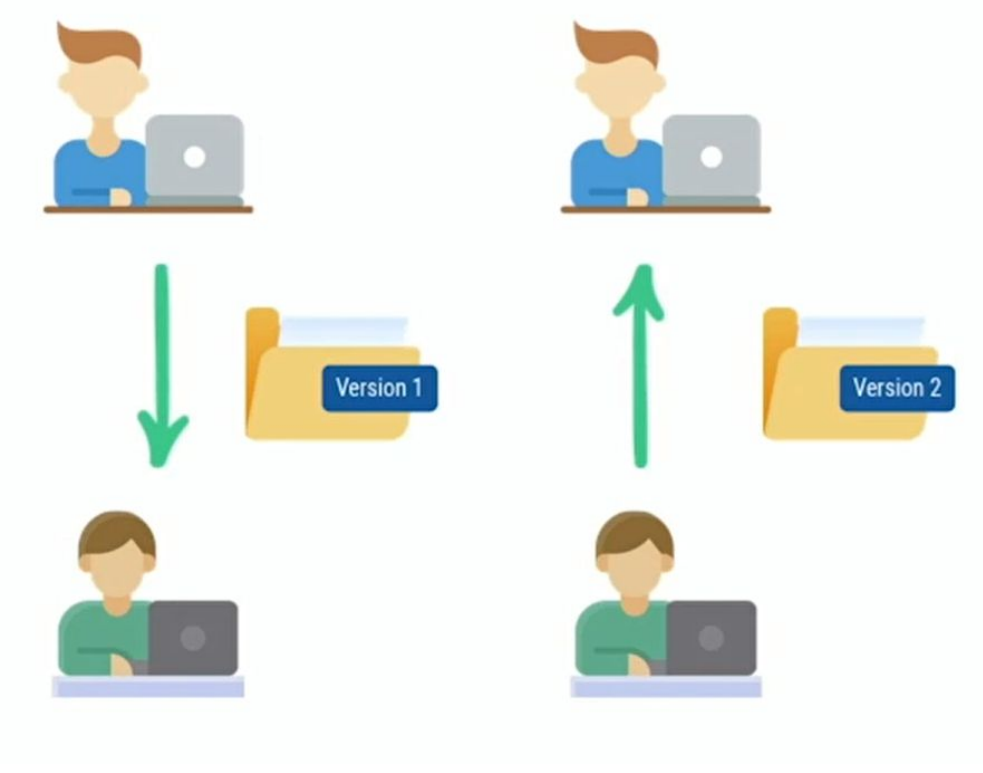
Advantages of Versioning

- We can quickly revert back to any of the older Versions, or pick up changes from older version.
- We can track how the files are modified over time.

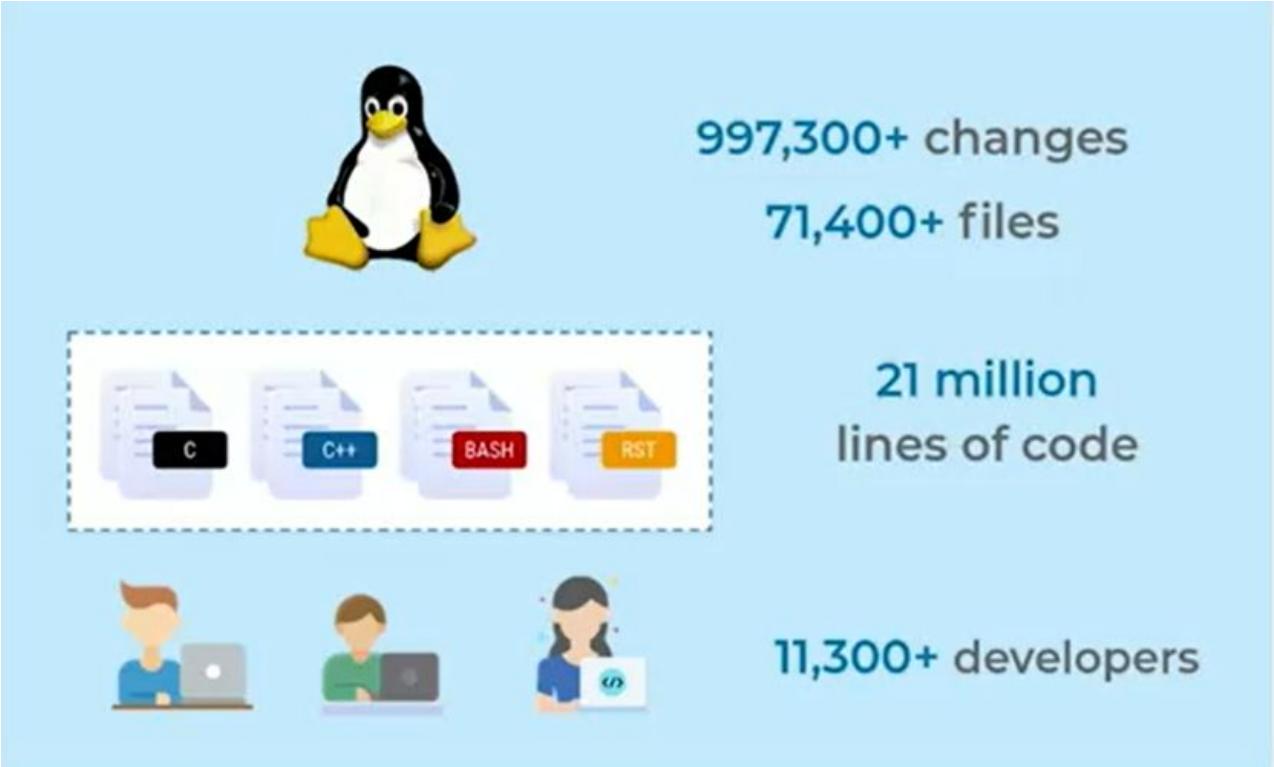
Managing Versions of a project



Managing Versions of a project



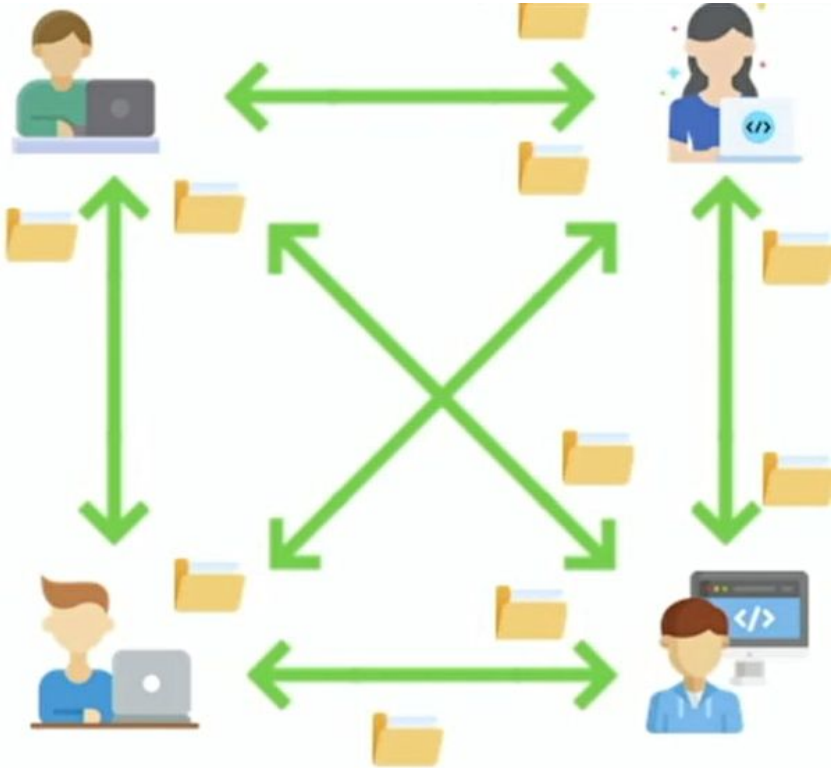
Linux Kernel Project



While working on a software Project, we keep making changes to existing files to add new features and fix existing bugs.

Collaboration

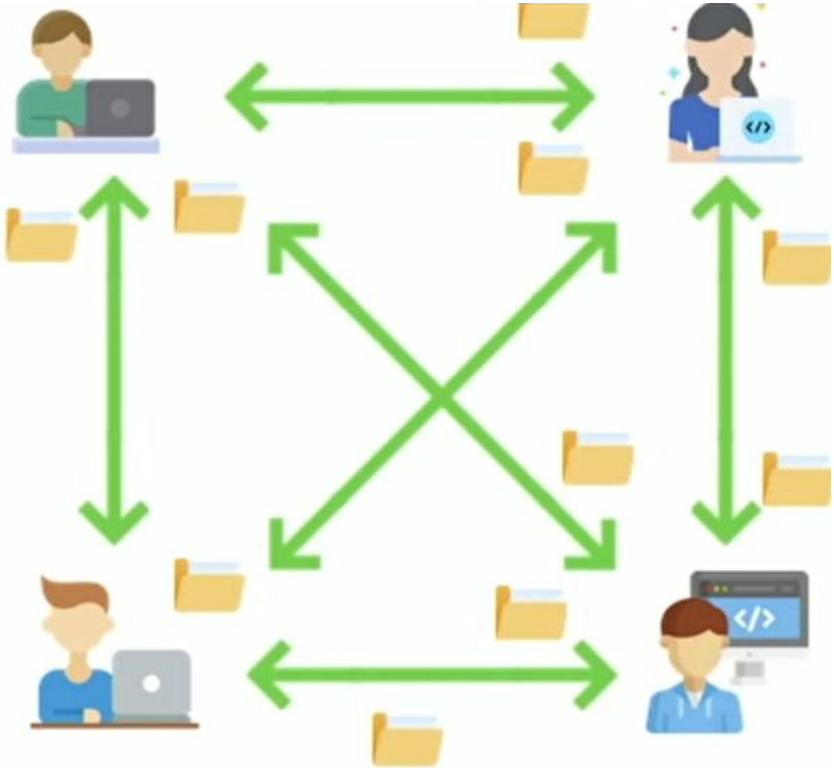
- While working on a huge Project with many files which being developed by multiple people it is hard to manually keep track of the changes.
- Any small change may completely crash the project.



Collaboration & Versioning

When working with multiple people it is useful to know

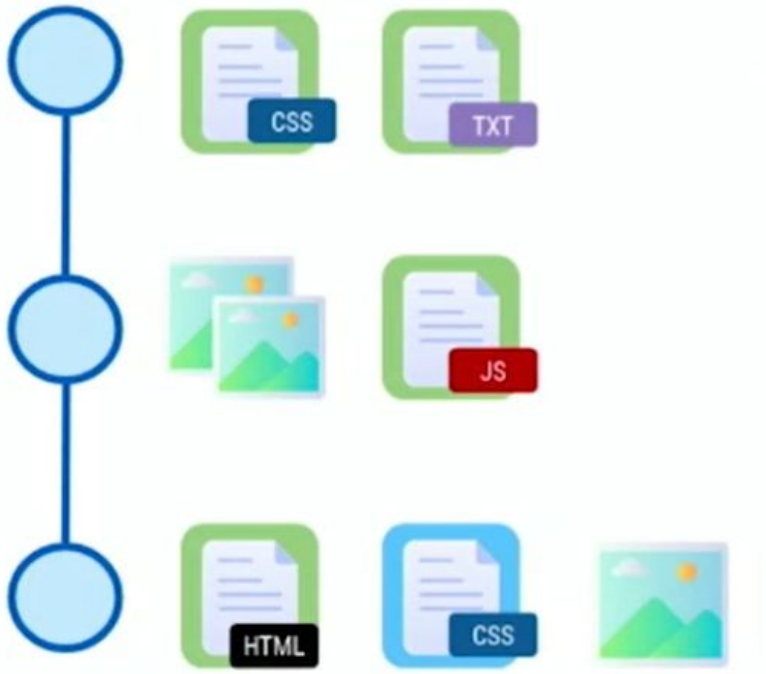
- Who made changes to given file?
- When are these changes made?



Source Code Management
Version Control System

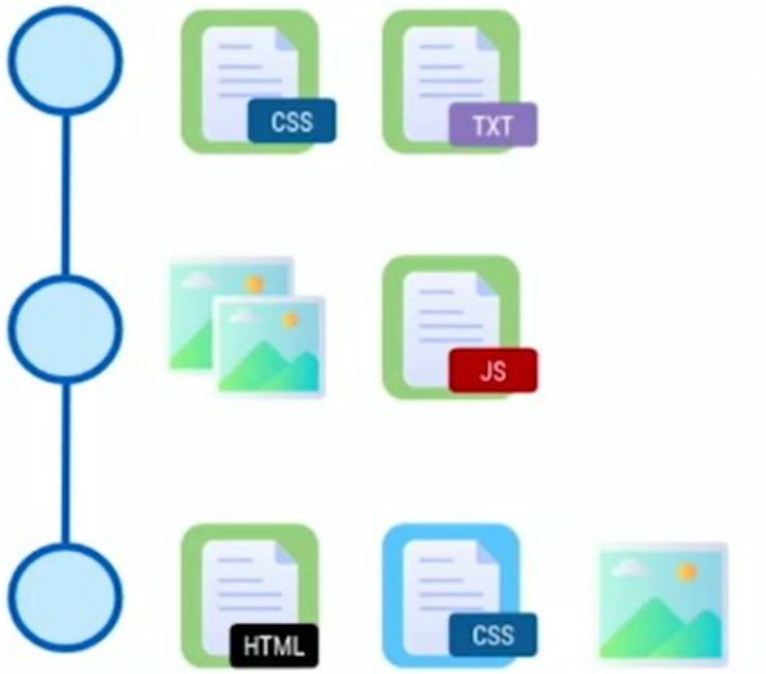
What is VCS?

A version control system records all changes made to a file or set of files, so a specific version may be called later if needed.



Source Code Management
Version Control System

- ★ Each change to Project can be considered a new Version of the Project.
- ★ Version control system simplifies tracking changes to the project and allows us to switch back to any previous version.



Source Code Management

Benefits of Version Control System

★ Keep track of all modifications made in code.

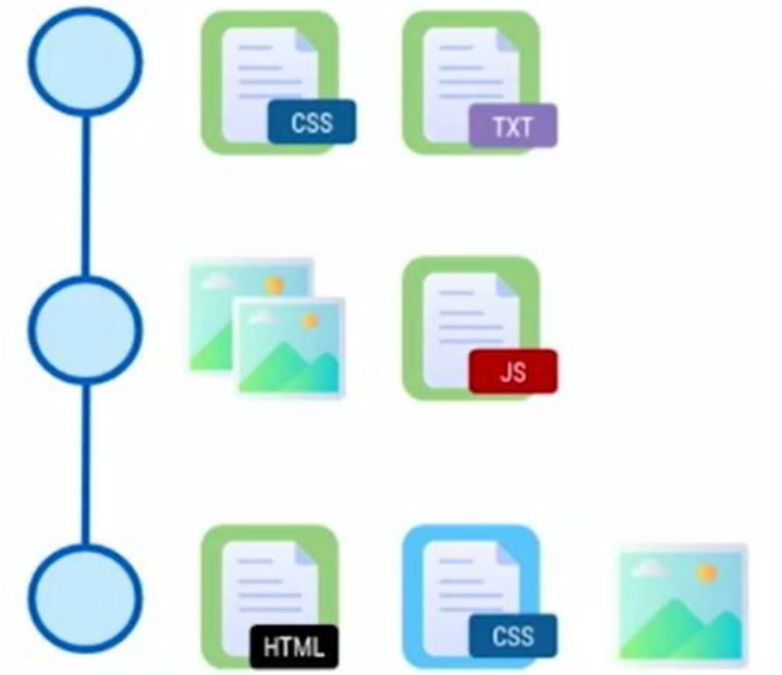
★ Comparing earlier versions of code.

★ Managing and protecting Source Code.

Version Control System

There are several tools that helps us manage versions of the source code of software.

- Git
- Subversion
- Mercurial



Git

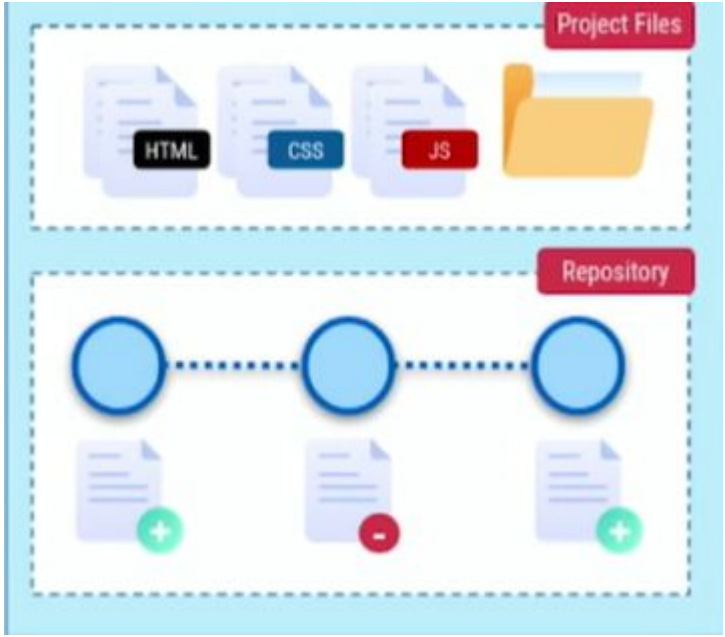
Git is a free, open-source and most widely used distributed version control system.

A software used for tracking changes in any set of files.

Git

Repository

A Git Repository (Repo) is like a database which maintains different versions of the project files.



Git

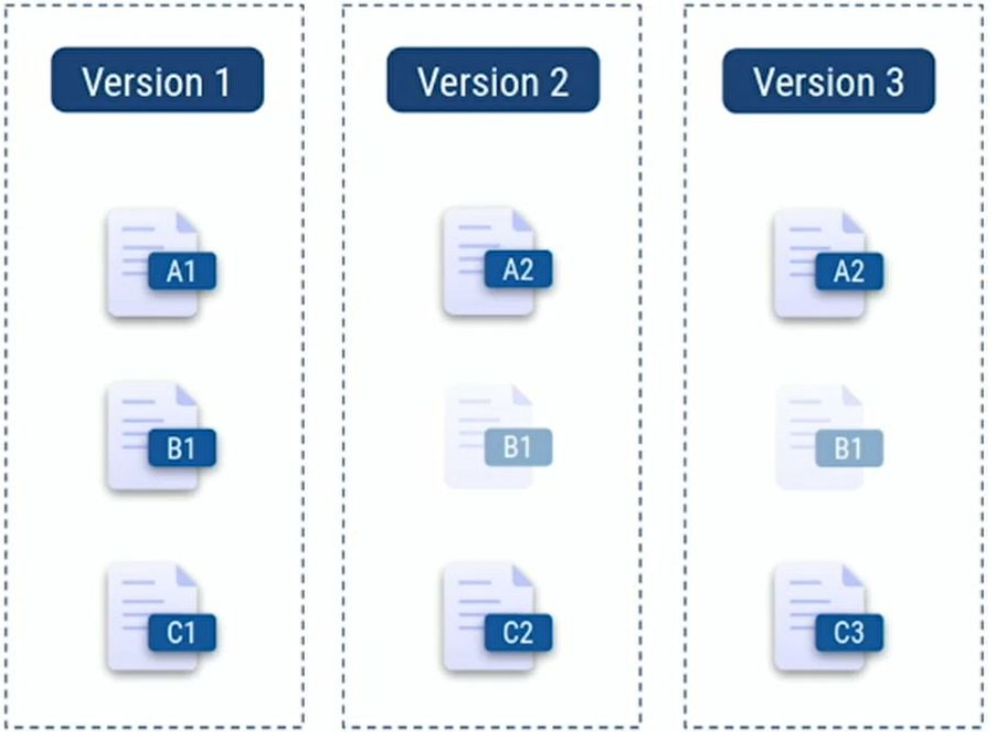
Snapshots

- Git allows us to take snapshots of our Project files to create versions of the Project.
- These versions or snapshots are referred to as Commits in Git terminology.

Git

Snapshots

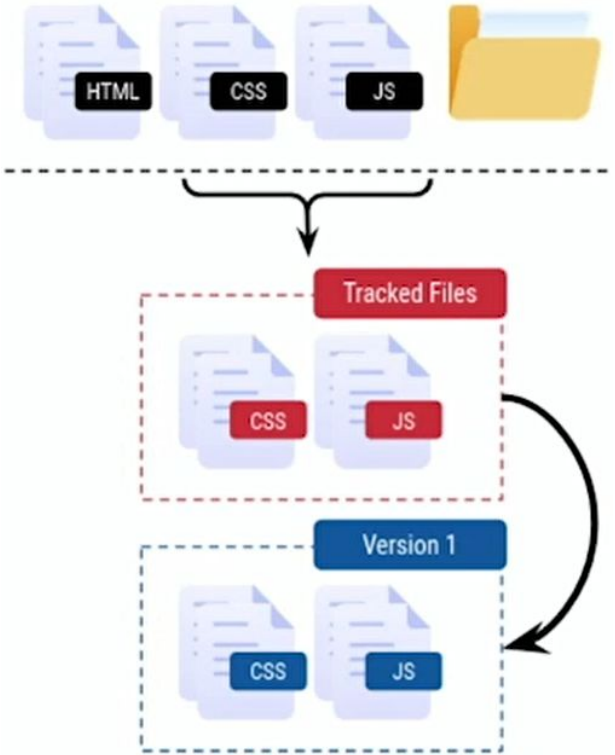
Example



Git

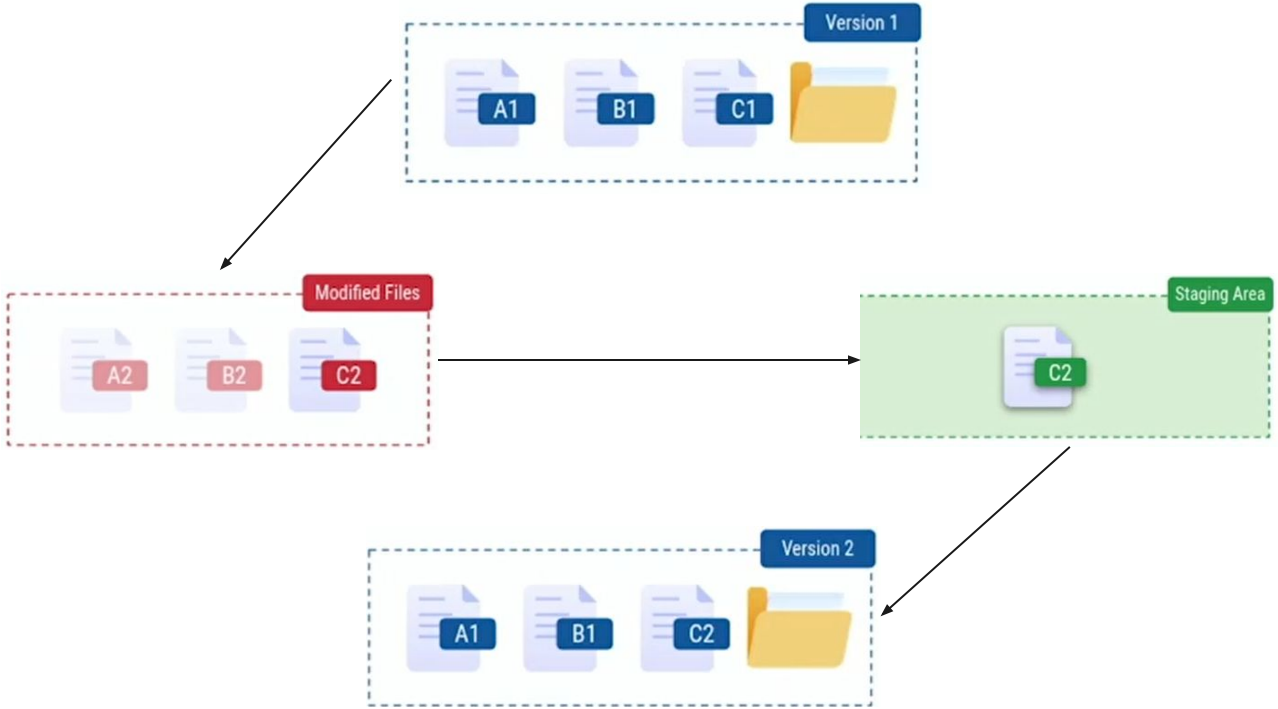
Tracking Files

- By default, Git doesn't track changes to a file and doesn't maintain versions automatically.
- We have to explicitly specify git to track file changes and save versions.



Git

Staging Area



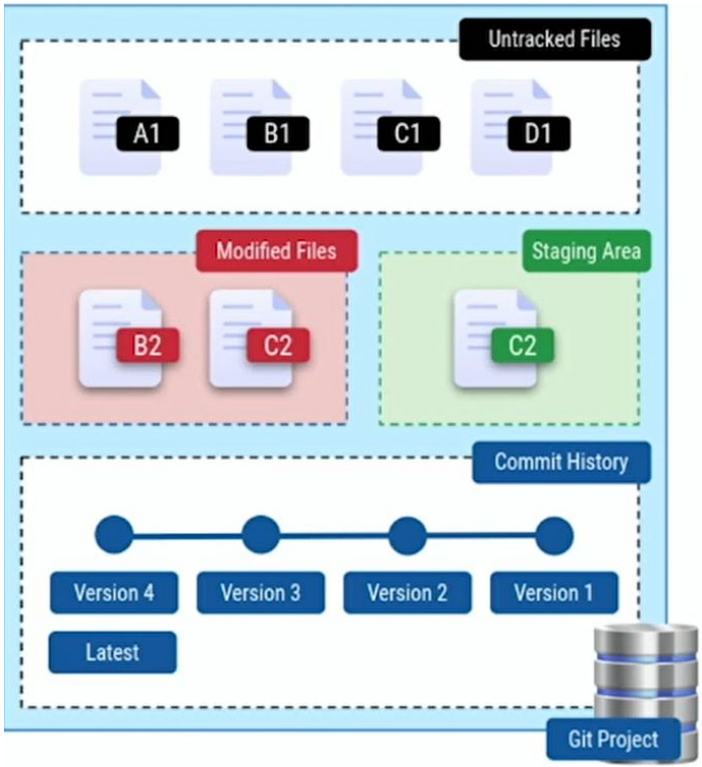
Git

Git's view of a Repository

Untracked Files : The set of files whose changes are not tracked by Git

Tracked Files : The set of Files which are watched by Git for any changes.

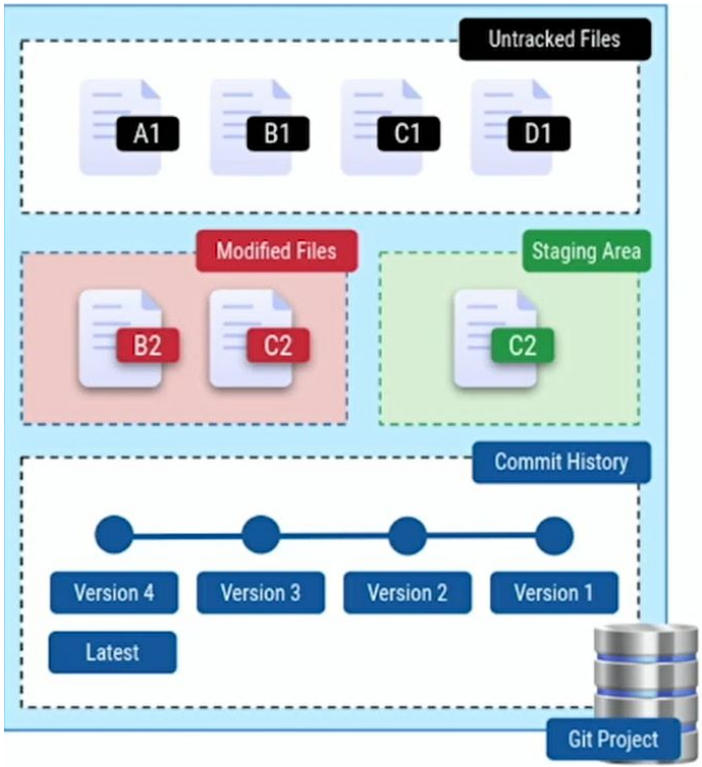
- Modified Files
- Staged Files
- Commited Files



Git

Git's view of a Repository

- **Modified Files :** These are the files which are modified after the latest snapshot.
- **Staged Files :** The set of files which are about to be committed to create new snapshot.
- **Commited Files :** These are the unmodified files which are same since last commit.



Git

Installing

On Linux

```
sudo apt install git
```

On Mac

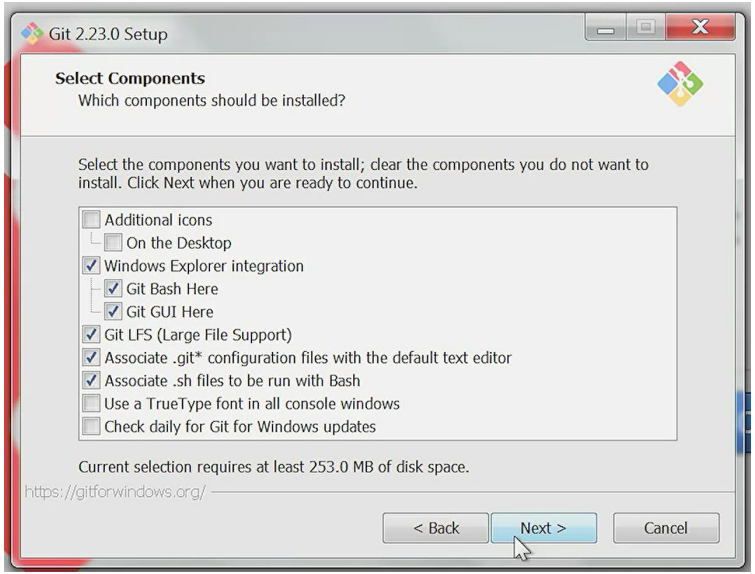
```
sudo brew install git
```

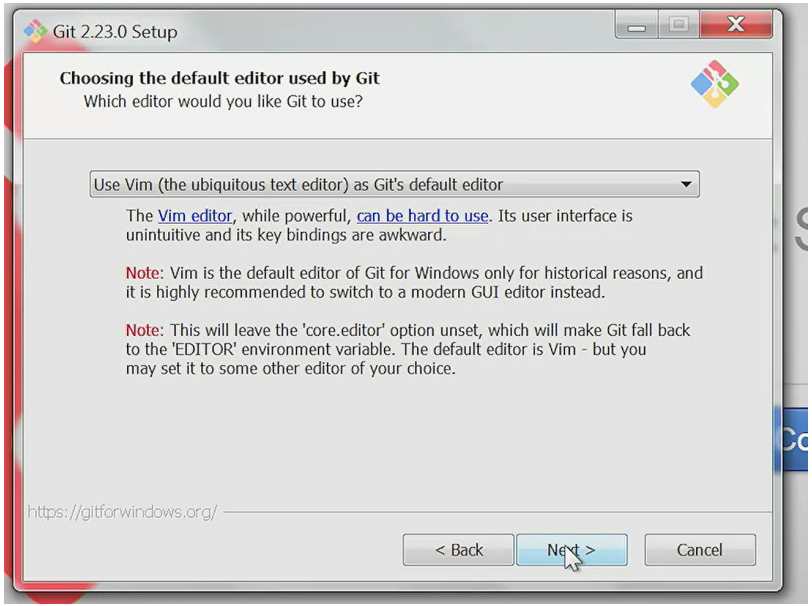
On Windows

Git - Downloading Package (git-scm.com)

Git

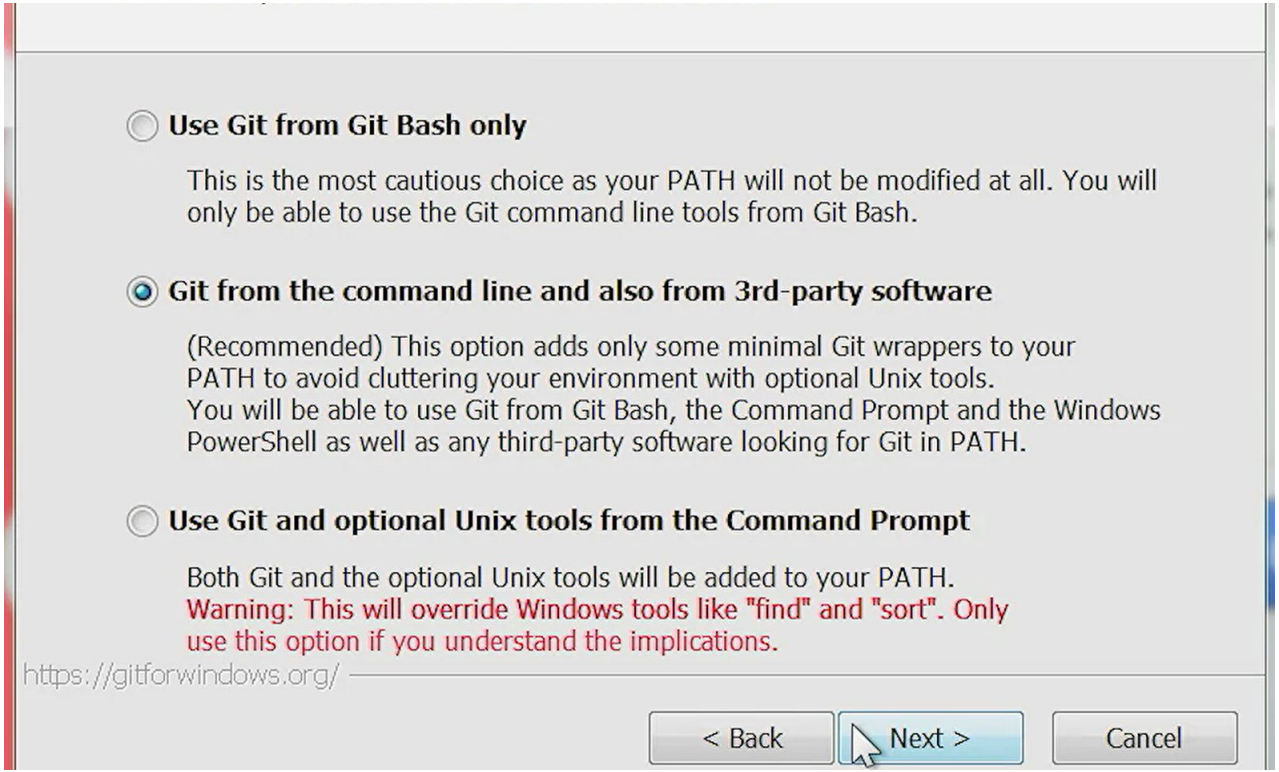
Installing on Windows





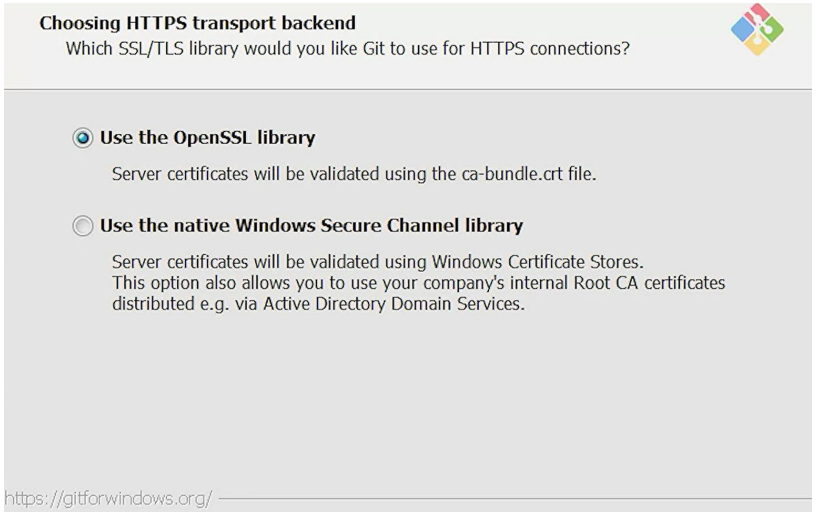
Git

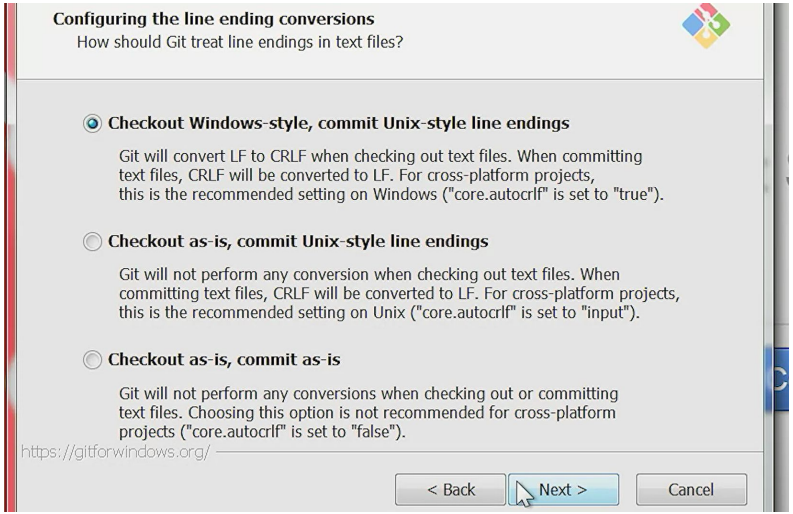
Installing on Windows



Git

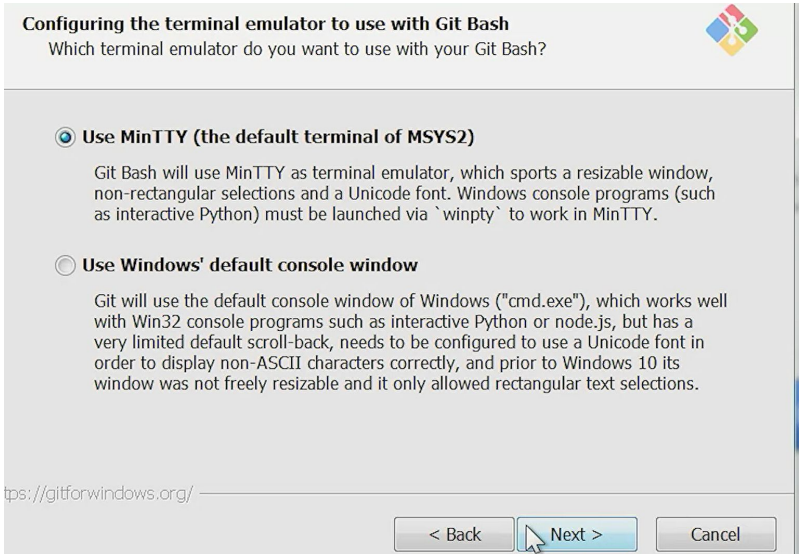
Installing on Windows

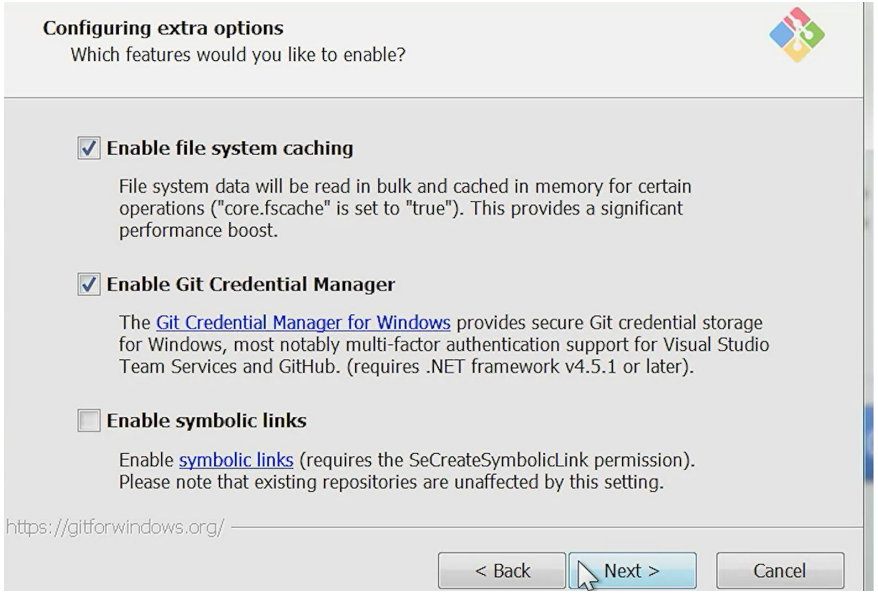




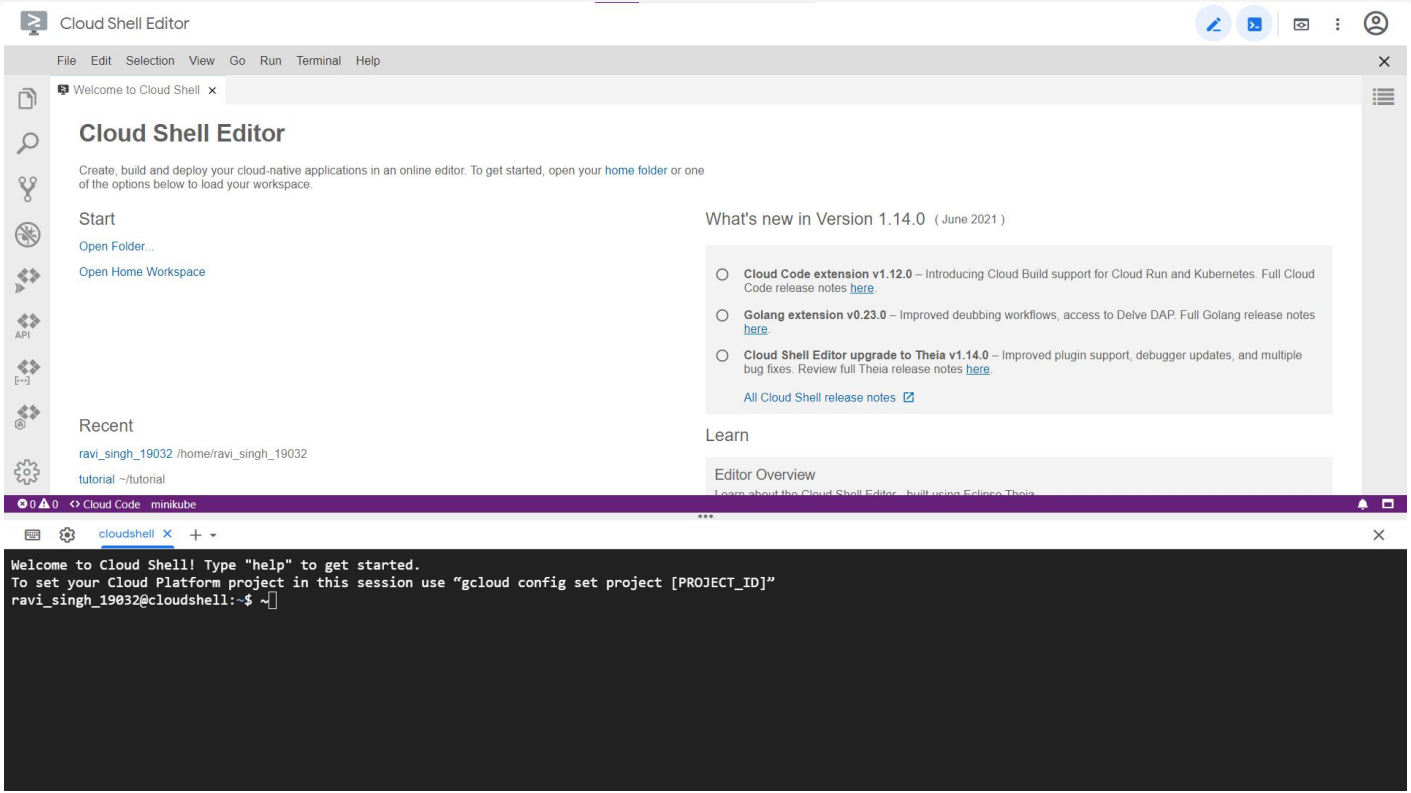
Git

Installing on Windows





Git



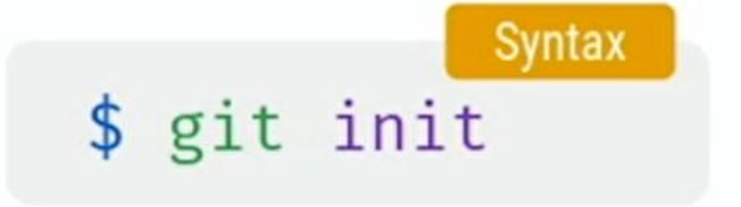
[Google Cloud Shell]

Working with Git

Initializing Repository

git init command is used to initialize a local repository

This command initializes an empty repository in working directory.



Working with Git

Working Directory

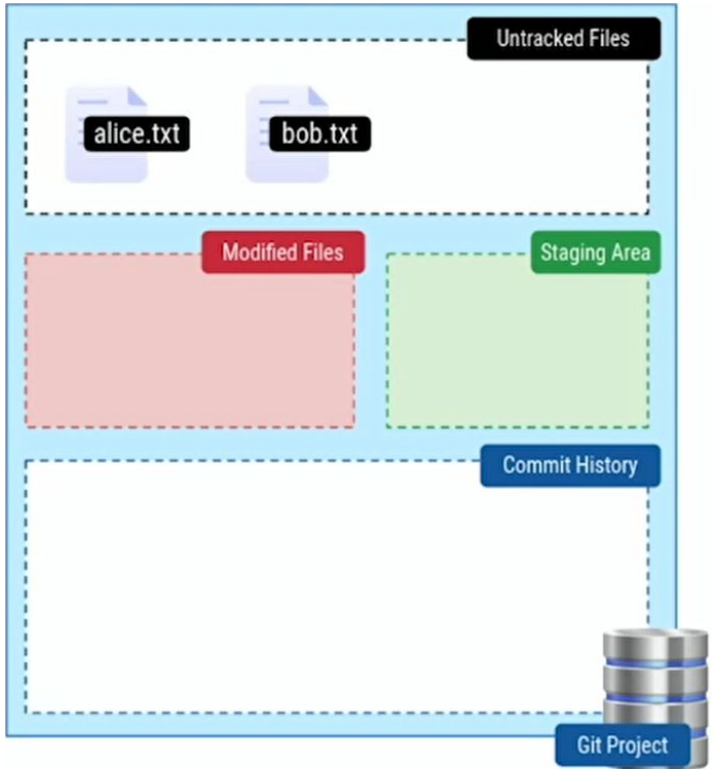
Consider creating two files **alice.txt** and **bob.txt** in tutorial folder

Bash

```
$ cd tutorial
$ touch alice.txt
$ touch bob.txt
$ ls
alice.txt
bob.txt
```

Working with Git

Git's view of Repository



Working with Git

Inspecting a Repository

Git track changes in the working directory.

git status command show changes in working directory.

Working with Git

Commit

Git commit command takes a snapshot representing the staged changes.

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

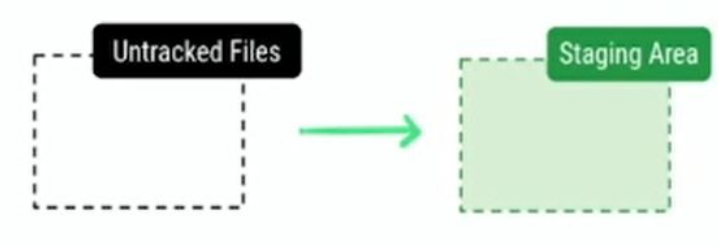
To skip staging area

```
git commit -a
```

Working with Git

Creating a commit

1. Add changes to staging area



2. Creating changes with commit in staging area

Working with Git

Creating a Commit

Adding changes to Staging Area

git add command adds the changes to the staging area.

Syntax

```
$ git add file_path
```

Bash

```
$ git add alice.txt
$ git status
Changes to be committed:
      new file:   alice.txt
Untracked files:
      bob.txt
```

Working with Git

Setting up Author Info

Configure who credits for the changes made from your device by setting the author details

```
git config --global user.name "Your Name"

git config --global user.email "youremailaddress@_Example_ .com"
```

Working with Git

Commiting Changes

Commit is a snapshot of the project's currently staged changes.

Here **message** provide useful information about what has changed and why.

```
ravi_singh_19032@cloudshell:~/tutorial$ git commit -m "adds alice file"

[master (root-commit) 303ab35] adds alice file
 1 file changed, 3 insertions(+)
 create mode 100644 alice.txt
```

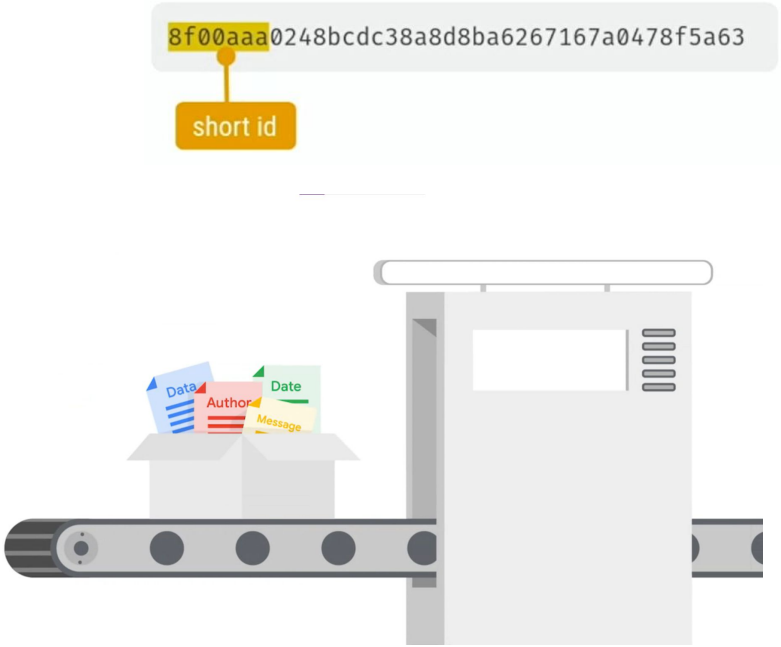
Working with Git

Commit ID

Commit IDs are unique strings(hashes) that are created whenever new commit is recorded.

Commit IDs are unique SHA-1 hashes.

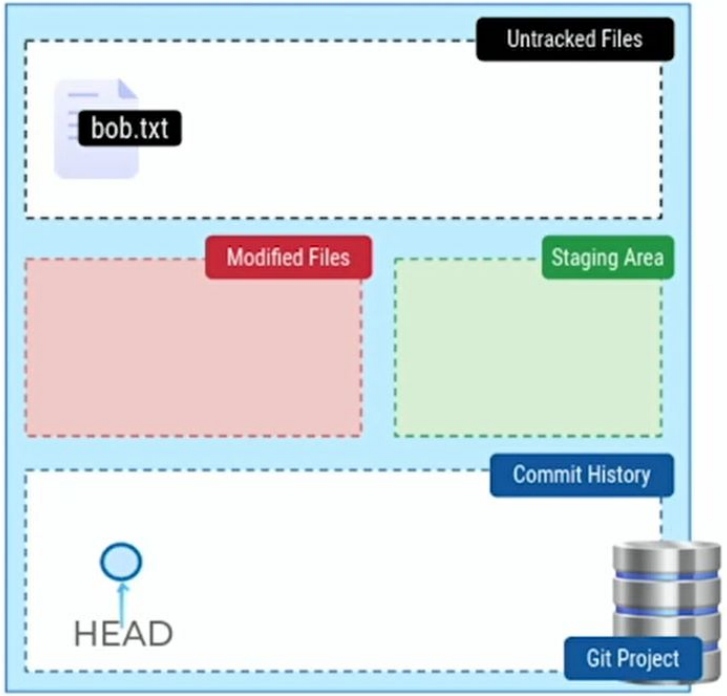
It is unique for every Commit.



Working with Git

Head

HEAD refers to the current commit



Working with Git

Commit Listing

git log list all the commits.

```
ssingh_raviprakash@cloudshell:~/tutorial$ git log
commit 0b6d3036b41a1efb3425f7d057742b20a83f69fd (HEAD -> master)
Author: Ravi Prakash Singh <ssingh.raviprakash@gmail.com>
Date:   Sat Jun 12 10:53:08 2021 +0000

    Removed Alice File

commit 138ded3e0a8e063810481dd319d7f9bc8f68721c
Author: Ravi Prakash Singh <ssingh.raviprakash@gmail.com>
Date:   Sat Jun 12 10:49:46 2021 +0000

    Adds Bob's file

commit f10af64eddc8ca0cb2aa786aeb2f545d48eca177
Author: Ravi Prakash Singh <ssingh.raviprakash@gmail.com>
Date:   Sat Jun 12 04:56:54 2021 +0000

    Adds Alice file
ssingh_raviprakash@cloudshell:~/tutorial$
```

Working with Git

log Command

git log --oneline displays as one commit per line.

```
ssingh_raviprakash@cloudshell:~/tutorial$ git log --oneline
0b6d303 (HEAD -> master) Removed Alice File
138ded3 Adds Bob's file
f10af64 Adds Alice file
ssingh_raviprakash@cloudshell:~/tutorial$
```

```
ssingh_raviprakash@cloudshell:~/tutorial$ git log --stat
commit 0b6d3036b41a1efb3425f7d057742b20a83f69fd (HEAD -> master)
Author: Ravi Prakash Singh <ssingh.raviprakash@gmail.com>
Date:   Sat Jun 12 10:53:08 2021 +0000

    Removed Alice File

 Alice.txt | 3 ---
 1 file changed, 3 deletions(-)

commit 138ded3e0a8e063810481dd319d7f9bc8f68721c
Author: Ravi Prakash Singh <ssingh.raviprakash@gmail.com>
Date:   Sat Jun 12 10:49:46 2021 +0000

    Adds Bob's file

 Bob.txt | 2 ++
 1 file changed, 2 insertions(+)
```

git log --stat displays files that have been modified. It shows summary of no of lines modified.

git log --patch displays files that have been modified, with location of modification.

```
ssingh_raviprakash@cloudshell:~/tutorial$ git log -p
commit 0b6d3036b41a1efb3425f7d057742b20a83f69fd (HEAD -> master)
Author: Ravi Prakash Singh <ssingh.raviprakash@gmail.com>
Date:   Sat Jun 12 10:53:08 2021 +0000

    Removed Alice File

diff --git a/Alice.txt b/Alice.txt
deleted file mode 100644
index 7037bf6..0000000
--- a/Alice.txt
+++ /dev/null
@@ -1,3 +0,0 @@
-Hi
-What about your learning
-Is it going good?
\ No newline at end of file
```

Working with Git

diff Command

git diff shows changes between commit, commits and working tree.

```
ssingh_raviprakash@cloudshell:~/tutorial$ git diff
diff --git a/Bob.txt b/Bob.txt
index a321560..cae6ca8 100644
--- a/Bob.txt
+++ b/Bob.txt
@@ -1,2 +1,3 @@
 Hi
-What's your name?
\ No newline at end of file
+What's your name?
+This is new line added.
\ No newline at end of file
```

Working with Git

.gitignore files

.gitignore files are used to tell the git tool to intentionally ignore some files in a given Git repository.

A few common examples of file patterns to exclude can be found here.

Working with Git

Renaming and Deleting Files

This command is similar to Linux rm and mv commands, except the changes are also staged.

git mv <filename>

git rm <filename>

```
ssingh_raviprakash@cloudshell:~/tutorial$ git rm Alice.txt
rm 'Alice.txt'
ssingh_raviprakash@cloudshell:~/tutorial$ git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        deleted:    Alice.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        .gitignore

ssingh_raviprakash@cloudshell:~/tutorial$
```

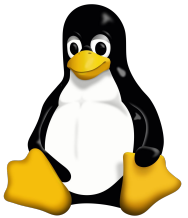
Cheat Sheet

git commit -a	Stages files automatically
git log -p	Produces patch text
git show	Shows various objects
git diff	Is similar to the Linux `diff` command, and can show the differences in various commits
git diff --staged	An alias to --cached, this will show all staged files compared to the named commit
git add -p	Allows a user to interactively review patches to add to the current commit
git mv	Similar to the Linux `mv` command, this moves a file
git rm	Similar to the Linux `rm` command, this deletes, or removes a file

Linux

Basic Commands

- pwd
- ls
- cd
- mkdir & rmdir
- rm
- touch
- cp
- mv
- echo
- cat
- nano



Linux

Basic Commands

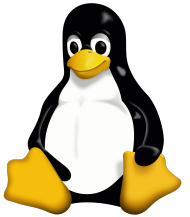
pwd- This command lets you know the current working directory

```
leomajor@ROG-Strix:~$ pwd
/home/leomajor
leomajor@ROG-Strix:~$
```

ls - list all the files in current directory

```
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents$ ls
'My Music'   'My Pictures'   'My Videos'   Untitled.ipynb
```

You can use -a flag to list hidden files and -l flag to get more information about files.



Linux

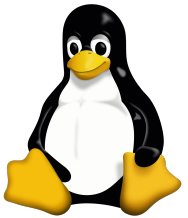
Basic Commands

cd - This command is used to change directory

```
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents$ cd My\ Music
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/My Music$
```

mkdir - Used to create a folder
rmdir - Delete a Directory

```
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$ mkdir Git
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$ ls
Git
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$ rmdir Git
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$ ls
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$
```



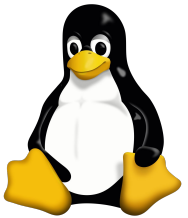
Linux

Basic Commands

rm - Used to delete files and directories.

You can use -r flag to delete a directory

```
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom/Git$ ls
a.txt  b.txt  c.txt
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom/Git$ rm a.txt
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom/Git$ ls
b.txt  c.txt
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom/Git$ cd ..
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$ rm -r Git
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$ ls
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$
```



Linux

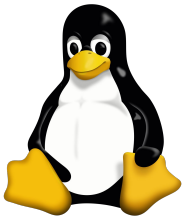
Basic Commands

touch - Used to create a file.

```
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$ touch Hello.txt
leomajor@ROG-Strix:/mnt/c/Users/ssing/Documents/Classroom$ ls
Hello.txt
```

cp - Used to copy files

```
n$ cp Hello.txt "New Folder"
```



Linux

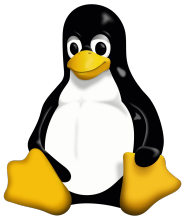
Basic Commands

echo - Used to move some data to text files or simply display on terminal.

```
$ echo "Hello World"
Hello World
```

```
$ echo "This line will be entered in file" >> new.txt
$ cat new.txt
This line will be entered in file
```

cat - Used to display content of a file



Key Takeaways

- What is Version Control System?
- Git
- Installing Git
- Using Git
  - First Step with Git
  - Basic Workflow
  - Tracking Files and Committing
- Git Interaction
  - Using Git Locally
  - Staging Area
  - Info about changes (**log** and **diff** command)