What is Git?

Git is a popular version control system. It was created by Linus Torvalds in 2005, and has been maintained by Junio Hamano since then.

It is used for:

* Tracking code changes
* Tracking who made changes
* Coding collaboration

What does Git do?

* Manage projects with **Repositories**
* **Clone** a project to work on a local copy
* Control and track changes with **Staging** and **Committing**
* **Branch** and **Merge** to allow for work on different parts and versions of a project
* **Pull** the latest version of the project to a local copy
* **Push** local updates to the main project

Working with Git

* Initialize Git on a folder, making it a **Repository**
* Git now creates a hidden folder to keep track of changes in that folder
* When a file is changed, added or deleted, it is considered **modified**
* You select the modified files you want to **Stage**
* The **Staged** files are **Committed**, which prompts Git to store a **permanent** snapshot of the files
* Git allows you to see the full history of every commit.
* You can revert back to any previous commit.
* Git does not store a separate copy of every file in every commit, but keeps track of changes made in each commit! Why Git?
* Over 70% of developers use Git!
* Developers can work together from anywhere in the world.
* Developers can see the full history of the project.
* Developers can revert to earlier versions of a project.

What is GitHub?

* Git is not the same as GitHub.
* GitHub makes tools that use Git.
* GitHub is the largest host of source code in the world, and has been owned by Microsoft since 2018.
* In this tutorial, we will focus on using Git with GitHub.

git config --global user.name "w3schools-test"

git config --global user.email [test@w3schools.com](mailto:test@w3schools.com)

git init

Initialized empty Git repository in /Users/user/myproject/.git/

## Git Staging Environment

One of the core functions of Git is the concepts of the Staging Environment, and the Commit.

As you are working, you may be adding, editing and removing files. But whenever you hit a milestone or finish a part of the work, you should add the files to a Staging Environment.

**Staged** files are files that are ready to be **committed** to the repository you are working on. You will learn more about commit shortly.

For now, we are done working with index.html. So we can add it to the Staging Environment:

git add index.html

git add –all

**Note:** The shorthand command for git add --all is git add –A

## Git Commit

Since we have finished our work, we are ready move from stage to commit for our repo.

Adding commits keep track of our progress and changes as we work. Git considers each commit change point or "save point". It is a point in the project you can go back to if you find a bug, or want to make a change.

When we commit, we should **always** include a **message**.

By adding clear messages to each commit, it is easy for yourself (and others) to see what has changed and when.

**Note:** Short status flags are:

* ?? - Untracked files
* A - Files added to stage
* M - Modified files
* D - Deleted files

We see the file we expected is modified. So let's commit it directly:

git commit -a -m "Updated index.html with a new line"

[master 09f4acd] Updated index.html with a new line

1 file changed, 1 insertion(+)

**Warning:** Skipping the Staging Environment is not generally recommended.

Skipping the stage step can sometimes make you include unwanted changes.

git remote add origin <https://github.com/w3schools-test/hello-world.git>

git push --set-upstream origin master

Git Ignore

When sharing your code with others, there are often files or parts of your project, you do not want to share.

Examples

* log files
* temporary files
* hidden files
* personal files
* etc.

Git can specify which files or parts of your project should be ignored by Git using a .gitignore file.

Git will not track files and folders specified in .gitignore. However, the .gitignore file itself **IS** tracked by Git.

## What is SSH

SSH is a secure shell network protocol that is used for network management, remote file transfer, and remote system access.

SSH uses a pair of SSH keys to establish an authenticated and encrypted secure network protocol. It allows for secure remote communication on unsecured open networks.

SSH keys are used to initiate a secure "handshake". When generating a set of keys, you will generate a "public" and "private" key.

The "public" key is the one you share with the remote party. Think of this more as the lock.

The "private" key is the one you keep for yourself in a secure place. Think of this as the key to the lock.

SSH keys are generated through a security algorithm. It is all very complicated, but it uses prime numbers, and large random numbers to make the public and private key.

It is created so that the public key can be derived from the private key, but not the other way around.

## Git Revert

revert is the command we use when we want to take a previous commit and add it as a new commit, keeping the log intact.

## Git Revert HEAD

We revert the latest commit using git revert HEAD (revert the latest change,  and then commit), adding the option --no-edit to skip the commit message editor (getting the default revert message):

git revert HEAD --no-edit

Now let's check the log again:

git log --oneline

## Git Reset

reset is the command we use when we want to move the repository back to a previous commit, discarding any changes made after that commit.

git reset 9a9add8

**Warning:** Messing with the commit history of a repository can be dangerous. It is usually ok to make these kinds of changes to your own local repository. However, you should avoid making changes that rewrite history to remote repositories, especially if others are working with them.

## Git Undo Reset

Even though the commits are no longer showing up in the log, it is not removed from Git.

## Git commit --amend

commit --amend is used to modify the most recent commit.

It combines changes in the staging environment with the latest commit, and creates a new commit.

This new commit replaces the latest commit entirely.