# What is Git?

# “Git is a distributed version-control system for tracking changes in source code during software development. It is designed for coordinating work among programmers, but it can be used to track changes in any set of files. Its goals include speed, data integrity, and support for distributed, non-linear workflows.” ([Wikipedia](https://en.wikipedia.org/wiki/Git))

What is the purpose of Git?

“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.” ([sbf5](https://www.sbf5.com/~cduan/technical/git/git-1.shtml)). Note: A Git repository, although usually used for managing source code files, can be used to manage any other types of files.

What is GitHub?

“GitHub is a Git repository hosting service, but it adds many of its own features. While Git is a command line tool, GitHub provides a Web-based graphical interface. It also provides access control and several collaboration features, such as a wikis and basic task management tools for every project” ([techcrunch](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=2ahUKEwiCpJGyg9njAhWEfXAKHcfPDYgQFjACegQIDxAH&url=https%253A%252F%252Ftechcrunch.com%252F2012%252F07%252F14%252Fwhat-exactly-is-github-anyway%252F&usg=AOvVaw1d5iWNMNPUkbK3kXGkARho))

Git vs GitHub

“Git is a distributed version control tool that can manage a development project's source code history, while GitHub is a cloud based platform built around the Git tool. Git is a tool a developer installs locally on their computer, while GitHub is an online service that stores code pushed to it from computers running the Git tool. The key difference between Git and GitHub is that Git is an open-source tool developers install locally to manage source code, while GitHub is an online service to which developers who use Git can connect and upload or download resources.” ([theserverside](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=12&cad=rja&uact=8&ved=2ahUKEwjx84nfhNnjAhUVUd4KHXGvBSUQFjALegQIARAB&url=https%253A%252F%252Fwww.theserverside.com%252Fvideo%252FGit-vs-GitHub-What-is-the-difference-between-them&usg=AOvVaw2mtAZlaQiaYC5cDsyJ5xHM)). Basically speaking, Git is the tool used on your local machine and GitHub is the tool that allows you to extend the benefits of the Git tool to a remote cloud which allows for you to share the files on your local machine to the GitHub cloud for easy collaboration with others.

Before we jump into setting up and learning how to use Git and GitHub, one prerequisite before we get started is that you have at least a very basic knowledge on how to navigate a unix based command line. For a basic Unix cheat sheet click [here](https://www.tjhsst.edu/~dhyatt/superap/unixcmd.html). For a list of all Unix commands click [here](https://en.wikipedia.org/wiki/List_of_Unix_commands).

Now that we have a basic knowledge of Git, GitHub and the command line, let’s get started with how to set up your machine to be able to utilize the Git and GitHub services.

1. Install Git on your machine
   1. [Mac OS X & Windows](https://git-scm.com/downloads)
   2. [Linux & Unix](https://git-scm.com/download/linux)
2. Create a [GitHub](https://github.com/join?source=header-home) account
3. Open terminal on Mac/Linux or open Git bash on Windows
   1. Run the commands:
      1. git config --global user.name "sample\_user"
      2. git config --global user.email "sample@email.com"
         * Recommendation: Use the username and email you used to create your GitHub account and place inside the quotes.
4. Basic Git commands and what they do:
   1. git init
      1. Initializes a local git repository in the current working directory.
   2. git remote add origin <HTTPS or SSH link>
      1. Connect your local repository to a GitHub remote server.
   3. git clone <HTTPS or SSH link>
      1. Clone a remote repository stored on GitHub in the current working directory.
   4. git status
      1. List the files you've changed and files that still need to be added or committed.
   5. git add <file>
      1. Add one or more files to the staging index before committing changes to the repository.
   6. git commit -m “commit message”
      1. Commit changes in staging index to the local repository. Note: This does not update the GitHub remote repository.
   7. git push <remote name> <branch name>
      1. Send the changes in your local repository to the specified remote branch on GitHub.
   8. git branch
      1. Lists all local branches in the current repository.
   9. git branch <branch name>
      1. Creates a new branch specified by <branch name>
   10. git checkout <branch name>
       1. Switches to the specified branch and updates the current working directory.
   11. git merge <branch>
       1. Merge the specified branch into the current active checked out branch.
   12. git pull <remote name> <branch name>
       1. Pull changes from the specified remote branch into the current active checked out branch.

For more information please refer to the official [Git Documentation](https://git-scm.com/doc).

After you have a basic understanding of how Git and GitHub works along with some of the basic commands, now all that is left is to practice using Git and GitHub yourself and deepen your knowledge on the many usage it has and to study up on some of the common best practices for using Git/GitHub for a project. I have listed some links below which I found to be good references:

[https://github.com/elsewhencode/project-guidelines#documentation](https://github.com/elsewhencode/project-guidelines%23documentation)

<https://nvie.com/posts/a-successful-git-branching-model/>

<https://git-scm.com/doc>