GitHub is now the largest online storage space of collaborative works that exists in the world. Whether you’re interested in participating in this global mind meld or in researching this massive file dump of human knowledge, you need to be here.

Simply by being a member, you can brush elbows with the likes of [Google](https://github.com/google) and [Facebook](https://github.com/facebook). Before GitHub existed, major companies created their knowledge mainly in private. But when you access their GitHub accounts, you’re free to download, study, and build upon anything they add to the network.

There are plenty of reasons to use GitHub if you’re not a programmer. According to GitHub’s educational videos, any knowledge worker can benefit, with “knowledge worker” defined as most any profession that makes use of a computer.

One of the main misconceptions about GitHub is that it’s a development tool, as much a part of coding as computer languages and compilers. However, GitHub itself isn’t much more than a social network like Facebook or Flickr. You build a profile, upload projects to share and connect with other users by “following” their accounts. And while many users store programs and code projects, there’s nothing preventing you from keeping text documents or other file types in your project folders.

What’s more, you can actually use GitHub without knowing ANY code at all. You don’t really need a tutorial to sign up and click around.

**What Is Git?**

The software that runs at the heart of GitHub was developed by [Linus Torvalds](http://en.wikipedia.org/wiki/Linus_Torvalds)(s a Finnish-American[[2]](https://en.wikipedia.org/wiki/Linus_Torvalds#cite_note-citLinus-2)[[6]](https://en.wikipedia.org/wiki/Linus_Torvalds#cite_note-oreg100914-6) [software engineer](https://en.wikipedia.org/wiki/Software_engineering), who is the creator of the [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel) and for a long time, principal developer; that became the kernel for operating systems (and many [distributions](https://en.wikipedia.org/wiki/Linux_distribution) of each), such as [GNU](https://en.wikipedia.org/wiki/GNU) and years later [Android](https://en.wikipedia.org/wiki/Android_%28operating_system%29) and [Chrome OS](https://en.wikipedia.org/wiki/Chrome_OS).)

**Why use Git ?**

Git is version control software, which means it manages changes to a project without overwriting any part of that project. Why use something like Git? Say you and a coworker are both updating pages on the same website. You make your changes, save them, and upload them back to the website. So far, so good. The problem comes when your coworker is working on the same page as you at the same time. One of you is about to have your work overwritten and erased.

A [version control](http://git-scm.com/video/what-is-version-control) application like Git keeps that from happening. You and your coworker can each upload your revisions to the same page, and Git will save two copies. Later, you can merge your changes together without losing any work along the way. You can even revert to an earlier version at any time, because Git keeps a “snapshot” of every change ever made.

The problem with Git is that it’s so ancient that we have to use the command line—or Terminal if you’re a Mac user—in order to access it, typing in snippets of code like ‘90s hackers. This can be a difficult proposition for modern computer users. That’s where GitHub comes in.

GitHub makes Git easier to use in two ways. First, if you [download the GitHub software](http://github.com/) to your computer, it provides a visual interface to help you manage your version-controlled projects locally. Second, creating an account on GitHub.com brings your version-controlled projects to the Web, and ties in social network features for good measure.

You can browse other GitHub users’ projects, and even download copies for yourself to alter and learn from. Other users can do the same with your public projects, and even spot errors and suggest fixes. Either way, no data is lost because Git saves a “snapshot” of every change.

## Words People Use When They Talk About Git

**Command Line:** The computer program we use to input Git commands. On a Mac, it’s called Terminal. On a PC, it’s a non-native program that you download when you download Git for the first time (we’ll do that in the next section). In both cases, you type text-based commands, known as prompts, into the screen, instead of using a mouse.

**Repository:** A directory or storage space where your projects can live. Sometimes GitHub users shorten this to “repo.” It can be local to a folder on your computer, or it can be a storage space on GitHub or another online host. You can keep code files, text files, image files, you name it, inside a repository.

**Version Control:** Basically, the purpose Git was designed to serve. When you have a Microsoft Word file, you either overwrite every saved file with a new save, or you save multiple versions. With Git, you don’t have to. It keeps “snapshots” of every point in time in the project’s history, so you can never lose or overwrite it.

**Commit:** This is the command that gives Git its power. When you commit, you are taking a “snapshot” of your repository at that point in time, giving you a checkpoint to which you can reevaluate or restore your project to any previous state.

**Branch:** How do multiple people work on a project at the same time without Git getting them confused? Usually, they “branch off” of the main project with their own versions full of changes they themselves have made. After they’re done, it’s time to “merge” that branch back with the “master,” the main directory of the project.

## Setting Up GitHub And Git For The First Time

First, you’ll need to sign up for an account on GitHub.com. It’s as simple as signing up for any other social network. Keep the email you picked handy; we’ll be referencing it again soon.

You could stop there and GitHub would work fine. But if you want to work on your project on your local computer, you need to have Git installed. In fact, GitHub won’t work on your local computer if you don’t install Git. Install Git for Windows, Mac or Linux as needed.

Now it’s time to go over to the command line. On Windows, that means starting the Git Bash app you just installed, and on OS X, it’s regular old Terminal. It’s time to introduce yourself to Git. Type in the following code:

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

Of course, you’ll need to replace “Your Name Here” with your own name in quotations. It can be your legal name, your online handle, anything. Git doesn’t care, it just needs to know to whom to credit commits and future projects.

Next, tell it your email and make sure it’s the same email you used when you signed up for a GitHub.com account just a moment ago. Do it like this:

git config --global user.email "your\_email@youremail.com"

That’s all you need to do to get started using Git on your computer.

## Tools & Features

Git for Windows focuses on offering a lightweight, native set of tools that bring the full feature set of the [Git SCM](http://git-scm.com) to Windows while providing appropriate user interfaces for experienced Git users and novices alike.

### Git BASH

Git for Windows provides a BASH emulation used to run Git from the command line. \*NIX users should feel right at home, as the BASH emulation behaves just like the "git" command in LINUX and UNIX environments.

### Git GUI

As Windows users commonly expect graphical user interfaces, Git for Windows also provides the Git GUI, a powerful alternative to Git BASH, offering a graphical version of just about every Git command line function, as well as comprehensive visual diff tools.

### Shell Integration

Simply right-click on a folder in Windows Explorer to access the BASH or GUI.

Resources

<https://git-for-windows.github.io/>

<http://rogerdudler.github.io/git-guide/>

<http://lifehacker.com/5983680/how-the-heck-do-i-use-github> (Should definitely check it out )

<http://www.howtogeek.com/180167/htg-explains-what-is-github-and-what-do-geeks-use-it-for/>