# Install GIT

This file contains instructions for installing GIT, observing its components, and performing one-time initializations. *IMPORTANT: read and follow the instructions at the end of this document for personalization that you should perform at the end of installation.*

I have installed git version 2.27.0 on my Windows 7 computer. The URL for this installation is in the Downloads folder.

For a later version, or to install a version for a different computer, refer to <https://git-scm.com/downloads>

After Git is installed Click *StartMenu > All Programs > Git* (folder). You will see several programs (5 programs in my current version of git). Three of these are of particular interest.

1. Git CMD – a Windows command prompt to be used with the “git” command.
2. Git Bash - a Unix-style command prompt for most standard unix commands plush “git” commands.
3. Git GUI – a Graphic Unser Interface dialog that provides many of the git facilities.

At this stage (8/6/2020) I have only brief experiences using Git. I have tried both Git CMD and Git Bash.

The major disadvantages of using Git Bash (instead of Git CMD) are

* The separator between a folder and its subfolder is forward-slash (/) contrary to the back-slash (\) in Windows.
* Commands in GitBash are case-sensitive. This makes little difference for git commands, which are always case-sensitive. But folder commands (such as “cd” and “dir”) must be typed lower-case.

The major advantages of using using Git Bash (instead of Git CMD) are

* The displayed response is multicolored, which improves readability.
* The displayed response is similar to what is described in the printed instructions ([Git - Reference](https://git-scm.com/docs)and [Git - Book](https://git-scm.com/book/en/v2)); in particular the command prompt is “$” instead of the fully qualified directory name.
* The displayed response contains “(master)” - colored blue – when the current folder is one of the folders of a git repository in the “master” branch. Git CMD does not provide this status.

For the reasons above I have chosen Git Bash as the preferred command prompt.

When Git Bash is invoked, its starting folder (by default) is C:\Users\sncole. To make it easy to start in a different folder,

* Copy a shortcut of Git Bash to the desktop.
* Right-click the shortcut copy, and then click **Properties**.
* Choose the **Shortcut** tab.
* Type the directory name of the desired starting folder in the **Start in:** text box.

You might have some text that you want to paste into the Git Bash command line. When there is text in the clipboard, simply move the cursor to the place where you want to copy the text, and then type the **INSERT** key to copy that text into the command line.

Some git commands produce a great deal of response text. By default Git Bash pauses when it reaches the point when it runs out of room in the window to display additional text, and it displays “:” to indicate that it waiting for you to permit it to proceed. You can type . . .

the **ENTER** key to let Git Bashdisplay one additional line of text

the **SPACE** key to let Git Bash display one additional page of text

“q” to terminate the display and return to the command prompt “$”.

When Git Bash finishes displaying all of the response text, it displays **END**; type “q” to return to the command prompt “$”.

Personalization

The instructions for personalization are in the git Book, Chapter “1.6 Getting Started – First-Time Git Setup”. Its URL is <https://git-scm.com/book/en/v2/Getting-Started-First-Time-Git-Setup>. In case this URL no longer works , use Google with keywords from the previous section to find this material.

In particular I typed the following commands at a command prompt:

$ git config - -global user.name SNCole

$ git config - -global user.email sncole@post.harvard.edu

$ git config - -global core.editor c:/Windows/System32/notepad.exe

Git uses the editor when it prompts you for a description of the reason for updating one or more files.

When using Git for revision control with Visual Studio source files, much of the work can be done without the need for Git CMD, Git Bash, or Git GUI. Instead, Visual Studio, itself provides a friendly user interface. Refer to the Pluralsight course “Using Git For Source Control In Visual Studio 2019”, or refer to my notes at **sncole\source\GitRepositories\WebAppCourseNotes\** **UsingGitForSourceControlInVisualStudio2019\** **UsingGit4SrcCtlInVS2019.docx**. An easy way to get started with a repository and a solution is described in the section titled Create a Local Git Repository.

Github

Github is a free service available on the internet. Using Github provides (1) backup storage for the Git revision-control information and (2) allows access that information from more than one computer.

One sets up a Github account at <https://github.com/>. In the process of creating a new account, Github will request a user ID and a password. As a 2nd layer of protection it will also be convenient to set up an SSH key, and it will also be necessary to supply a passphrase to use with the SSH key. When I tell git to copy information from the repository to the Github web site (the “push” command), git asks me for the SSH passphrase.

Use <https://docs.github.com/en/github/authenticating-to-github/connecting-to-github-with-ssh> or <https://git-scm.com/book/en/v2/Git-on-the-Server-Generating-Your-SSH-Public-Key#_generate_ssh_key> to create an SSH key, to add it to ssh-agent, and to add it to the GitHub account.