Obtain and install Git and Creating a new local project and Pushing it to a new Repository on github

1. First, make sure to obtain the git version for your OS (i.e. download from the git website at git-scm.com) and install/configure it on your computer.
   1.1. Obtain and install git from https://git-scm.com/downloads (As an option, you may also download and setup the Github desktop gui client app from - https://desktop.github.com/ or some other git GUI client app (e.g. GitKraken, Atlassian SourceTree etc.), if you like.You may also use the built-in git client from your chosen IDE/Code Editor tool (Eclipse, IntelliJ, Visual Studio Code etc).

   1.2. Follow these steps:
      1.2.1. For Microsoft Windows, do the following...
         1.2.1.1. step 1: Download the executable Git file from git-scm.com/downloads
         1.2.1.2. Step 2 : Run the installation file that is found in your downloads folder(from the directory you downloaded it to)/needs administrator privileges/
         1.2.1.3. Step 3: Choose an appropriate installation location such as C:\Program FilesStep 4: Install the default components, including “Git GUI Here” and “Git Bash Here”
         1.2.1.4. Step 5: Choose your preferred Git default editor(Notepad++, Microsoft Visual Studio Code, etc)
         1.2.1.5. Step 6: Allow Git to be added to the Windows PATH
         1.2.1.6. Step 7 : Accept the default line ending conversion for Unix and Windows Compatibility
         1.2.1.7. Step 8: Choose the extra option to enable system caching
         1.2.1.8. Step 9: Click Finish to complete the install
         1.2.1.9. Step 10: Choose to open a Git Bash Shell and start using Git!
      1.2.2. For Mac OS...
         1.2.2.1. ...
      1.2.3. For Linux
         1.2.3.1. Step 1: Got to https://git-scm.com/download/linux
         1.2.3.2. Step 2: Open your terminal and copy the installation command for the linux distribution (**Debian/Ubuntu, Fedora, Gentoo...)** you are currently working on.
         For example to install git on Ubuntu/Debian Distribution, type in the following commands below

```
sudo apt-get update
sudo apt-get install git
```

         1.2.3.3. Step 3: Check your installation

Type in the following command

```
git --version
```

You will see something like this on your terminal

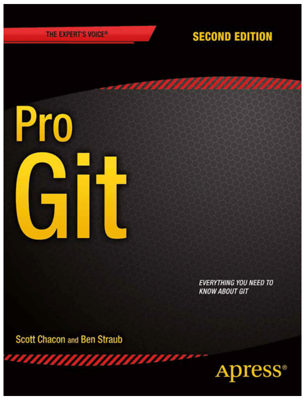
```
git version 2.xx.x
```

Depending on the version of git you installed you might see different numbers

Congratulations, you have successfully installed git on your linux machine. You can start practicing git commands by using this cheat sheet **https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet**

1.2.3.4. Step 4 (Optional): Adding SSH key to github
Watch this tutorial on how to add SSH key to github
https://www.youtube.com/watch?v=H5qNpRGB7Qw

If you want to explore more about git and how it works, you can read this book. It will teach you all the ins and outs of git and how to master it.


THE EXPERT'S VOICE
SECOND EDITION
Pro
Git
EVERYTHING YOU NEED TO KNOW ABOUT GIT
Scott Chacon and Ben Straub
Apress

https://git-scm.com/book/en/v2

1.3. A couple of global configs to do after installing Git:

1.4.

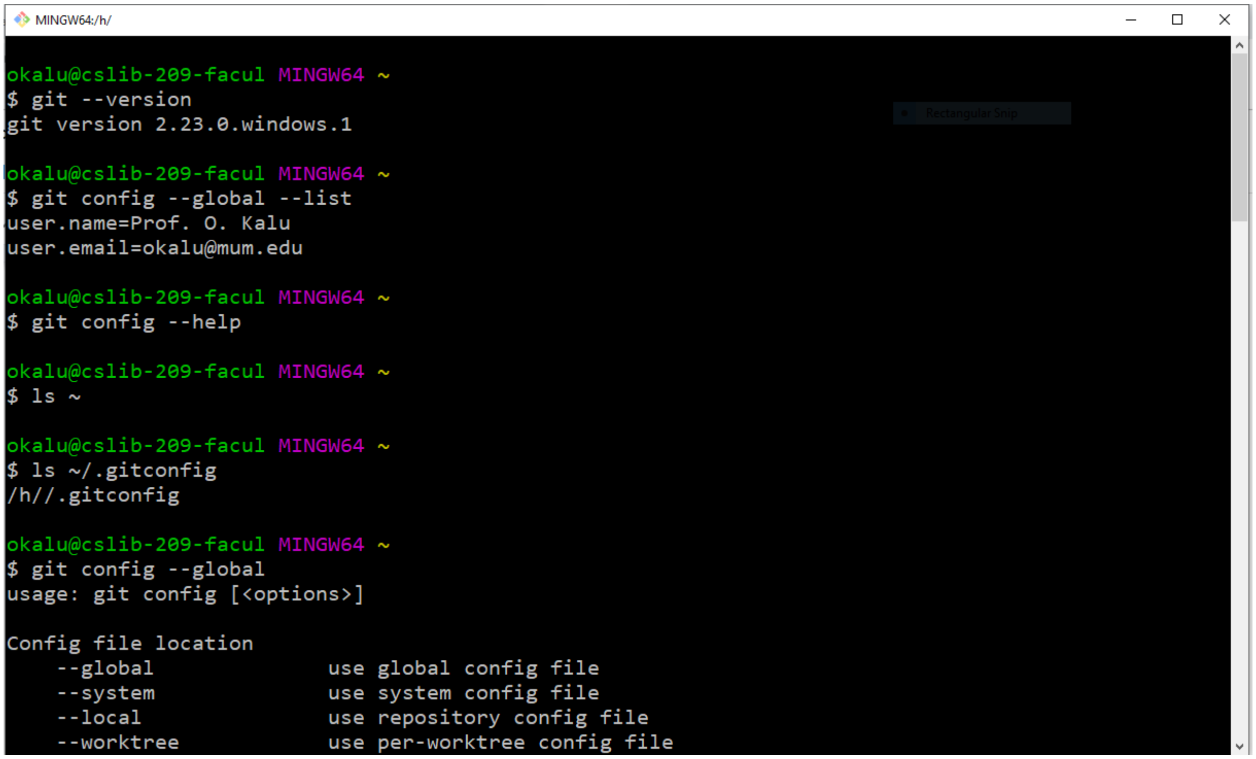
1.4.1. c:\> **git config --global user.name "YourName"**

1.4.2. c:\> **git config --global user.email "your-email@some-domain.com"**

2. Test the install & config by executing on the cmd-window/terminal, the command:

2.1. c:\> **git --version**

3. On the cmd-window/terminal, create a project folder and chdir to the project folder.
4. Initialize a git repository in the project folder by executing the cmd - *c:\project-folder>**git init***
5. Create/Add a new file to the project folder. Any filename is ok. e.g. document.txt
6. Display the list of existing source files that need to be added to the repo by running the *cmd - c:\project-folder>**git status***
7. Add all relevant files in preparation for commit - *c:\project-folder>**git add .***
8. Commit the files added - *c:\elibrary>**git commit -m "First cut - minimal springboot webapp"***
9. Sign-in or Sign-up on github here - https://github.com/
10. On github.com, create a new remote repository for the project (preferably with the same project name) and copy the url of the repository.
11. On the command window/terminal, run the following command to add/link the remote repository url to the local repo - *c:\elibrary>**git remote add origin https://github.com/okalu-cs425/elibrary.git***
12. Before pushing the commits up to the remote repo, first fetch any updated source files from the remote repo and have it synched/merged to the local repo by running - *c:\elibrary>**git pull origin master*** (Note: This step may not be necessary, if the remote repo on github was created such that it is totally empty with no source files such as LICENSE, README.md, .gitignore etc. If the repo isn't empty but contains any file(s), then this "git pull..." step becomes necessary; before a git push can be executed successfully).
13. Now, push the committed source files to the remote repository, by running the command - *c:\elibrary>**git push origin master***
14. **Additional Practice**
15. Clone the demo project repository at https://github.com/okalu-cs425-swe-202003/cs425-swe-demo-project-1
16. Open the project in your code editor or IDE
17. Add your name and/or student ID to the list-of-contributors.txt file
18. Commit your change and push the updated project to the remote repository
19. **Questions**:
    - 19.1. Was your push to the remote repository successful?
    - 19.2. What happened? And why?
    - 19.3. How else (as in, which other means) can you successfully add your change and have it incorporated into this repository?
    - 19.4. Now, go ahead and do it.
20. The End!!!



```
okalu@cslib-209-facul MINGW64 ~
$ git --version
git version 2.23.0.windows.1

okalu@cslib-209-facul MINGW64 ~
$ git config --global --list
user.name=Prof. O. Kalu
user.email=okalu@mum.edu

okalu@cslib-209-facul MINGW64 ~
$ git config --help

okalu@cslib-209-facul MINGW64 ~
$ ls ~

okalu@cslib-209-facul MINGW64 ~
$ ls ~/.gitconfig
/h//.gitconfig

okalu@cslib-209-facul MINGW64 ~
$ git config --global
usage: git config [<options>]

Config file location
    --global              use global config file
    --system              use system config file
    --local               use repository config file
    --worktree            use per-worktree config file
```