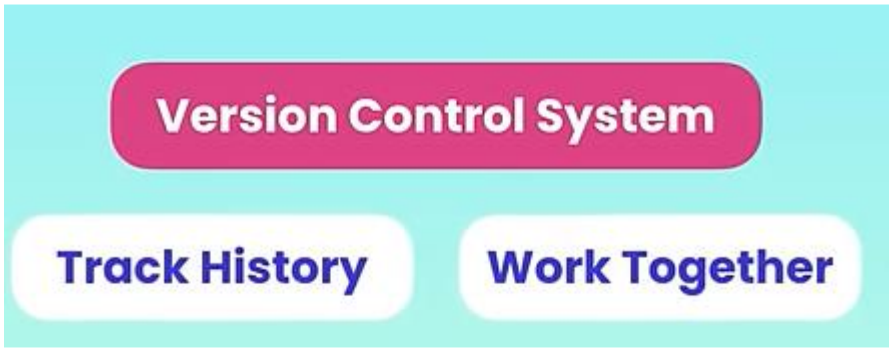
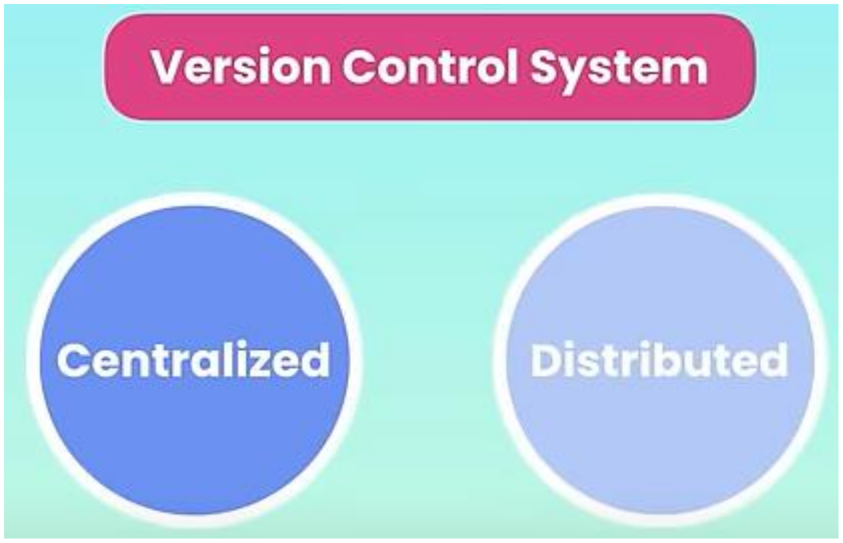
We Can Use Git For Many Things:

- **Merge** The Code Between Programmers.
- **Avoid** The Old-Way Of Organization Of Teams.
- **Avoid** The Old-Way Of Sending Emails Between Teams For Each New Version.
- **Avoid** The Bad-Ways Of Managing Versions Of Our Project.

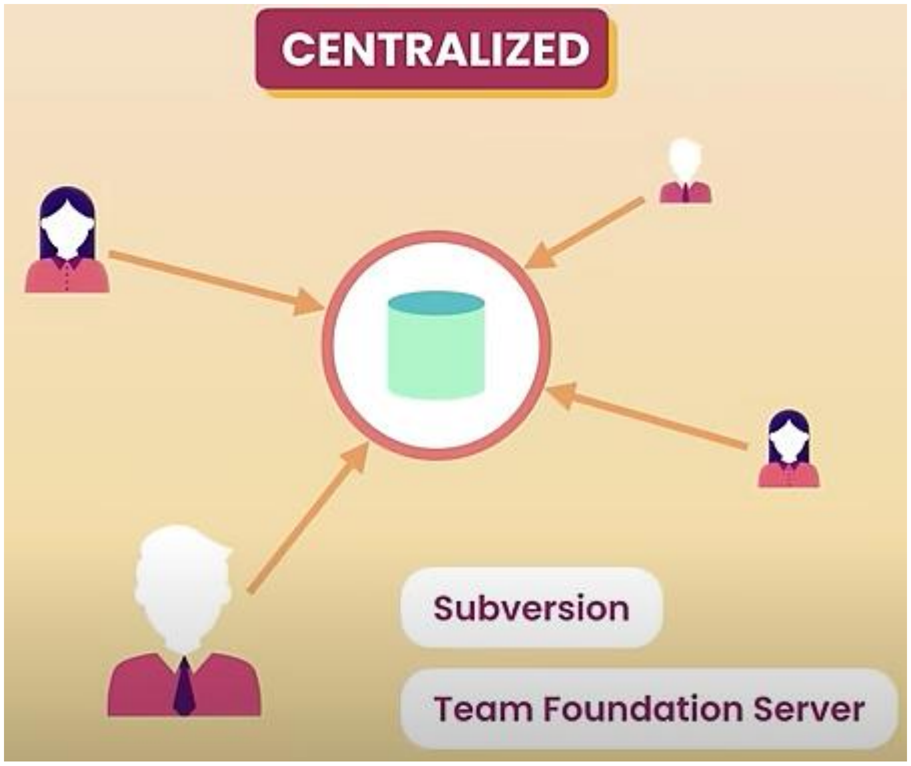
****



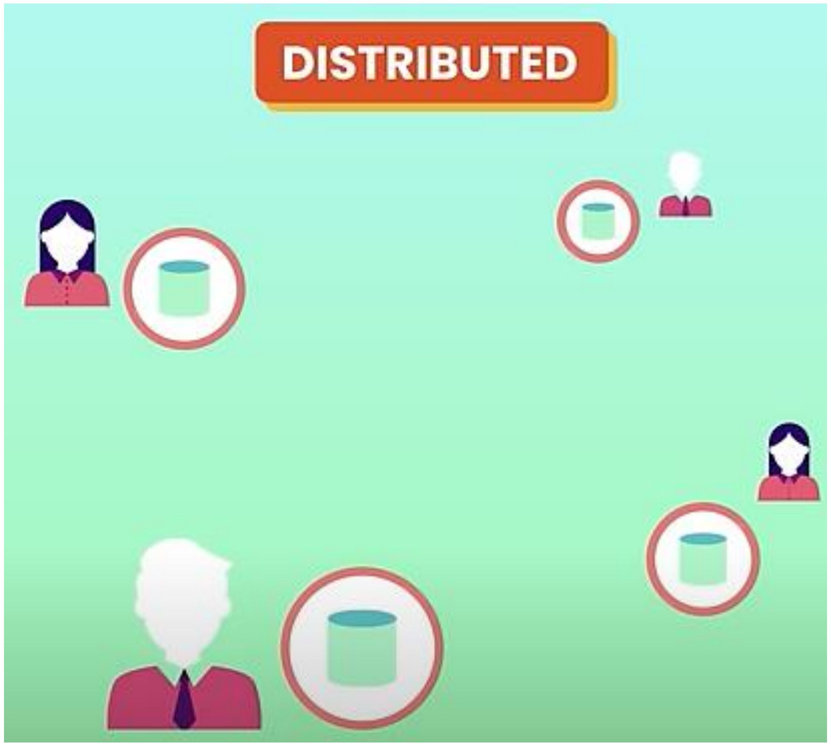
****



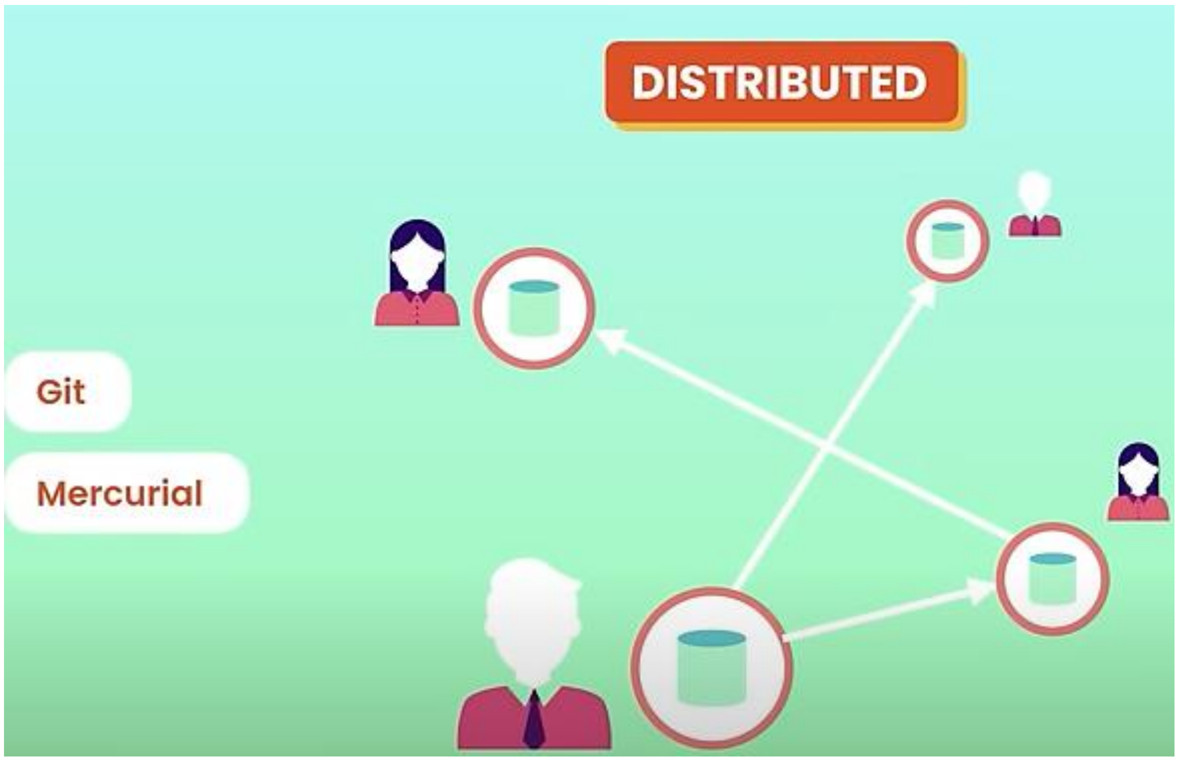
****


CENTRALIZED
Subversion
Team Foundation Server

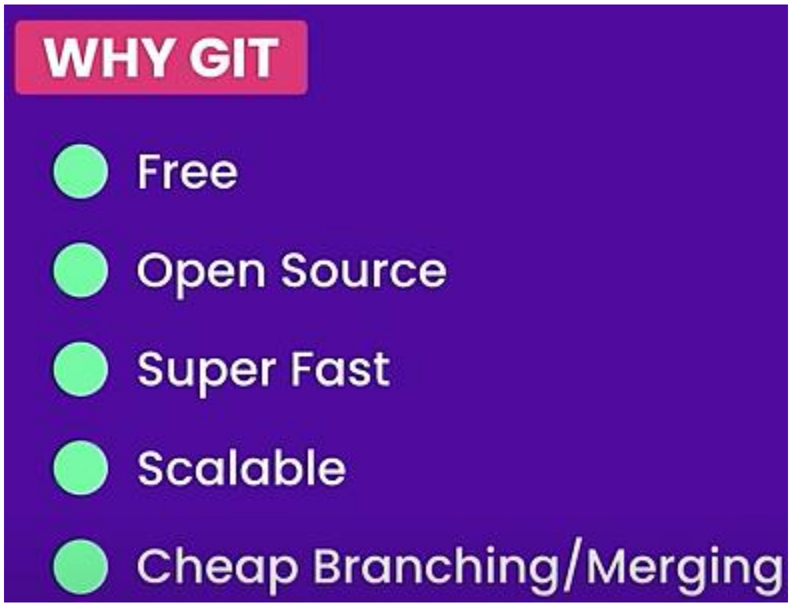
****


DISTRIBUTED

****



****

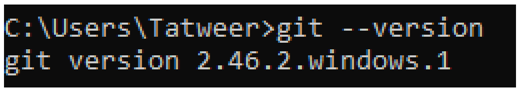


WHY GIT

- Free
- Open Source
- Super Fast
- Scalable
- Cheap Branching/Merging

****

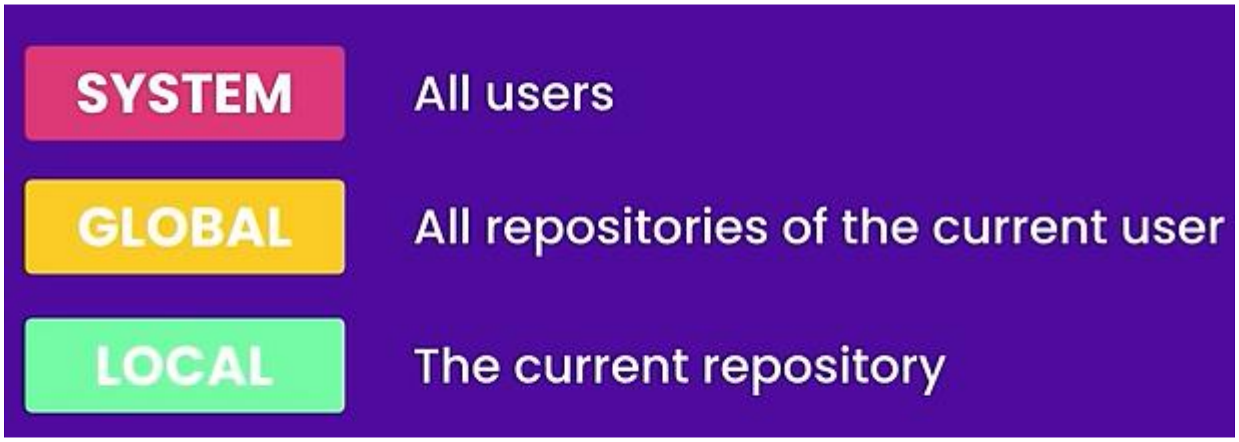
To Check If We Install Git, We Can Run The Command: **git --version**.



```
C:\Users\Tatweer>git --version
git version 2.46.2.windows.1
```

****

The Settings Of Git:

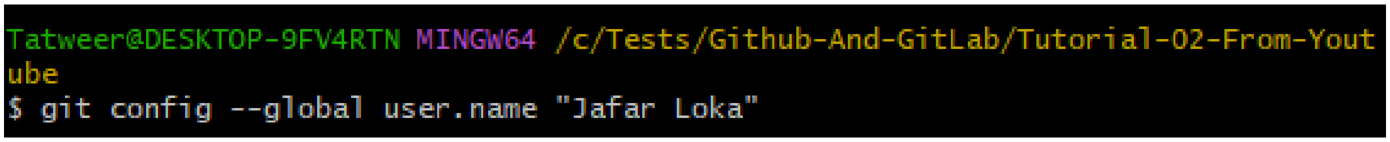


SYSTEM	All users
GLOBAL	All repositories of the current user
LOCAL	The current repository

****

To Set The ***Global Option For Name***:

Note: Here We Set "" Because The Name Contain Space.

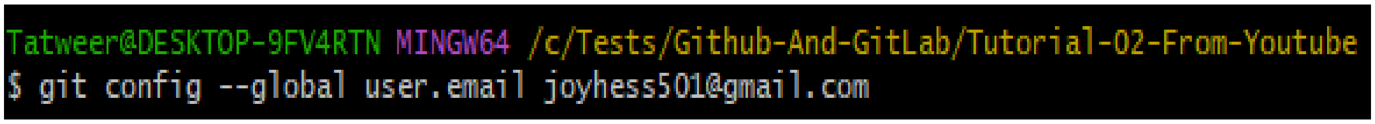


```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Yout
ube
$ git config --global user.name "Jafar Loka"
```

****

To Set The ***Global Option For Email***:

Note: Here We Don’t have Spaces In Email.

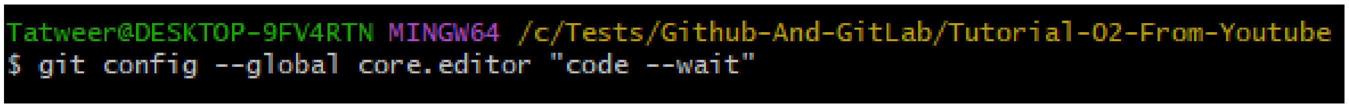


```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube
$ git config --global user.email joyhess501@gmail.com
```

****

To Set ***The Global Editor For Git***:

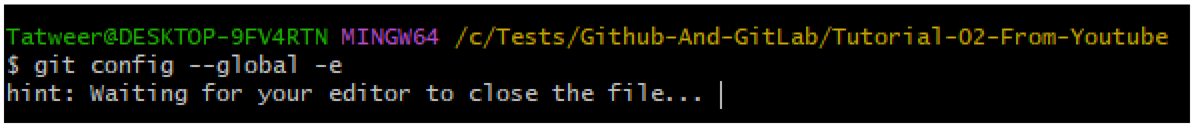
Note: Here We Set **--wait** For Telling Git Not Closing Until We Close The Editor.



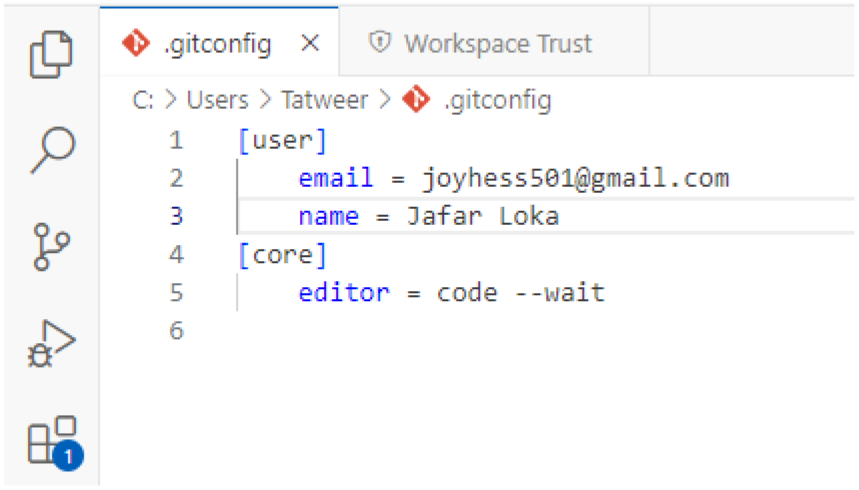
```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube
$ git config --global core.editor "code --wait"
```

****

To Open ***The Config File, Of Git***:

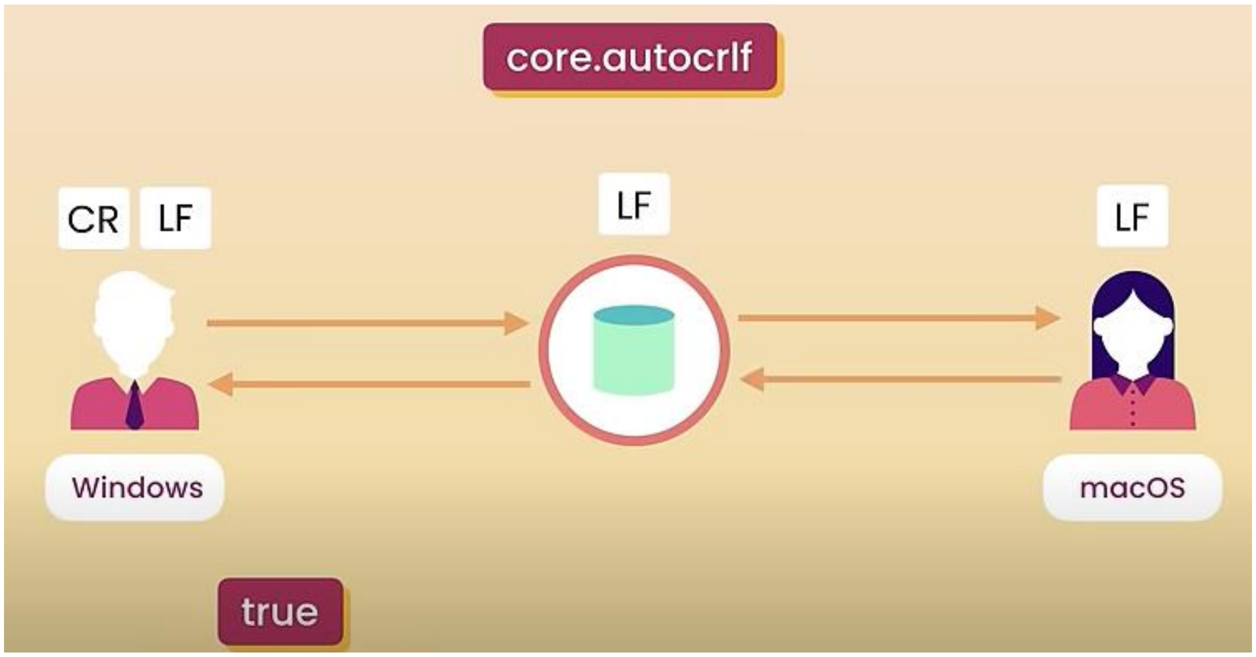


```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube
$ git config --global -e
hint: Waiting for your editor to close the file... |
```



```
[user]
    email = joyhess501@gmail.com
    name = Jafar Loka
[core]
    editor = code --wait
```

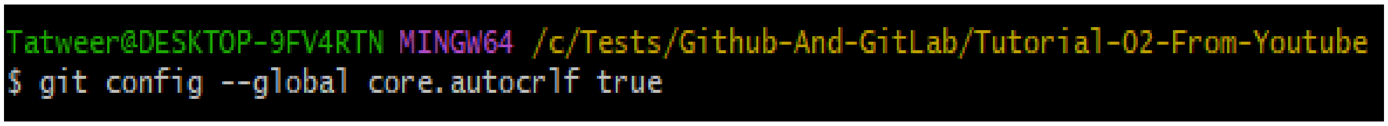
****



****

For Windows:

Note: In MacOS OR Linux We Must Set **true** **To** **input**.

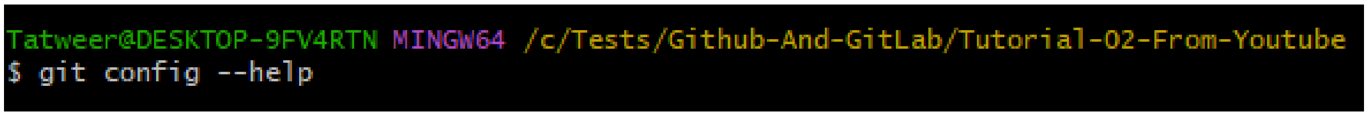


```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube
$ git config --global core.autocrlf true
```

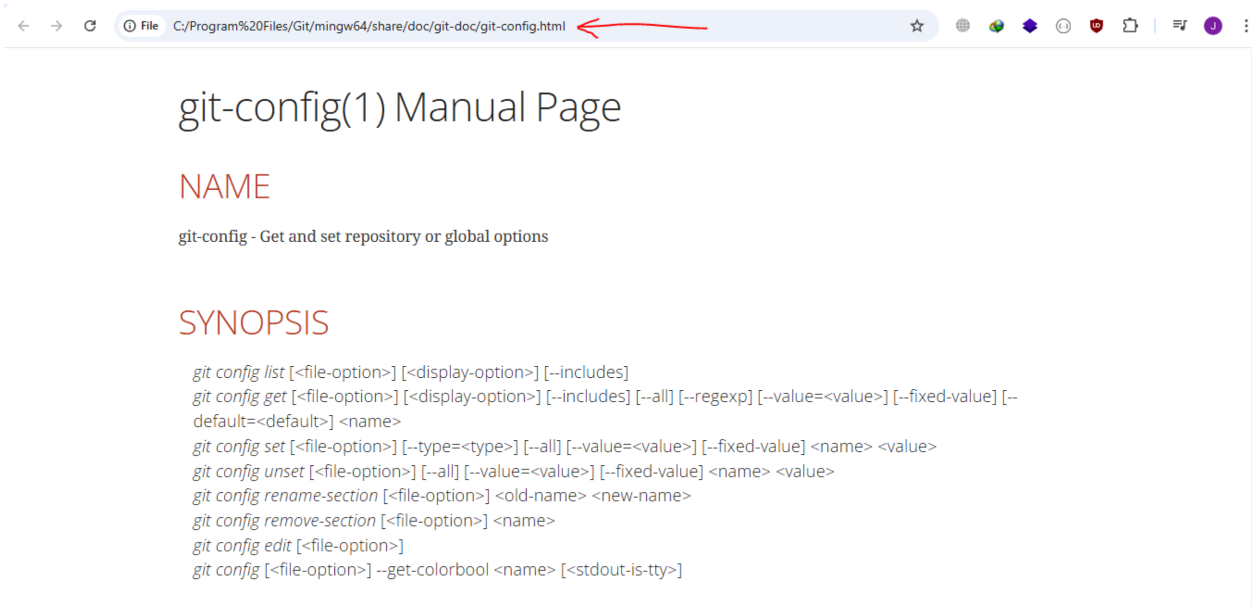
****

To Get The Help For Configuration, We Can:

- Search Google: *git config*
- Using Git Bash: *git config --help*



```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube
$ git config --help
```



C:/Program%20Files/Git/mingw64/share/doc/git-doc/git-config.html

git-config(1) Manual Page

NAME

git-config - Get and set repository or global options

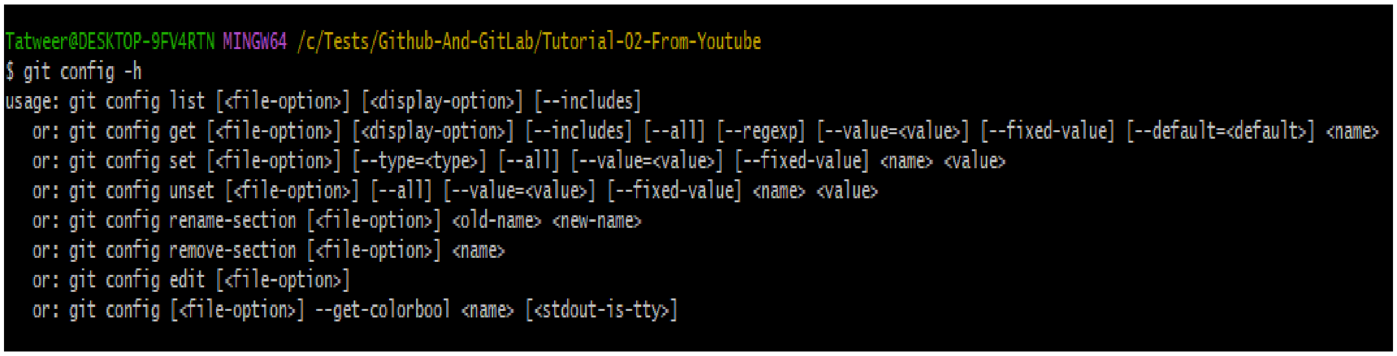
SYNOPSIS

```
git config list [<file-option>] [<display-option>] [--includes]
git config get [<file-option>] [<display-option>] [--includes] [--all] [--regexp] [--value=<value>] [--fixed-value] [--
default=<default>] <name>
git config set [<file-option>] [--type=<type>] [--all] [--value=<value>] [--fixed-value] <name> <value>
git config unset [<file-option>] [--all] [--value=<value>] [--fixed-value] <name> <value>
git config rename-section [<file-option>] <old-name> <new-name>
git config remove-section [<file-option>] <name>
git config edit [<file-option>]
git config [<file-option>] --get-colorbool <name> [<stdout-is-tty>]
```

****

To ***Get The Short Help For Configuration***:

- We Can Run: *git config -h*

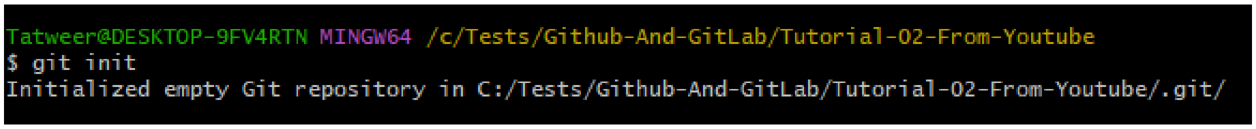


```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube
$ git config -h
usage: git config list [<file-option>] [<display-option>] [--includes]
   or: git config get [<file-option>] [<display-option>] [--includes] [--all] [--regexp] [--value=<value>] [--fixed-value] [--default=<default>] <name>
   or: git config set [<file-option>] [--type=<type>] [--all] [--value=<value>] [--fixed-value] <name> <value>
   or: git config unset [<file-option>] [--all] [--value=<value>] [--fixed-value] <name> <value>
   or: git config rename-section [<file-option>] <old-name> <new-name>
   or: git config remove-section [<file-option>] <name>
   or: git config edit [<file-option>]
   or: git config [<file-option>] --get-colorbool <name> [<stdout-is-tty>]
```

****

To ***Initialize Our Project Using Git***:

- Run Command: ***git init***



```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube
$ git init
Initialized empty Git repository in C:/Tests/Github-And-GitLab/Tutorial-02-From-Youtube/.git/
```

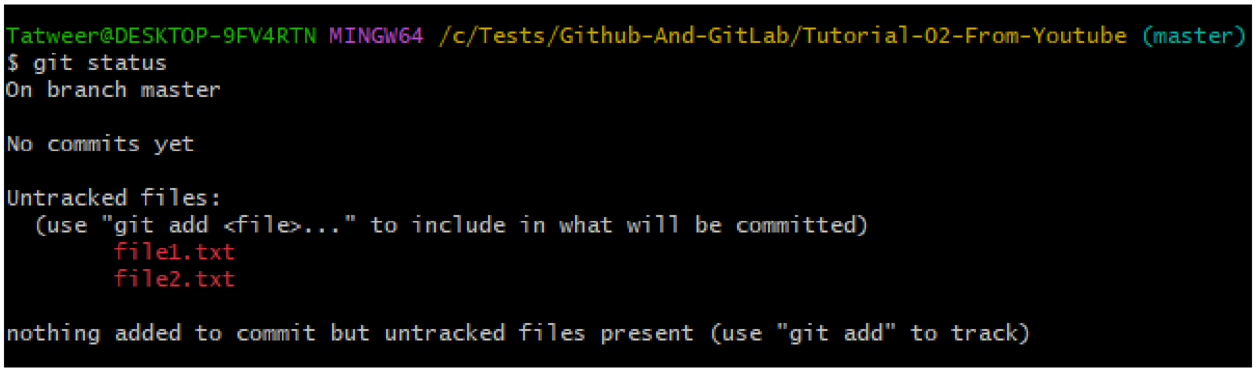
****

In Git We Have Area, Called: Staging Area, That Contains Also Index, For Each Change We Made To Our Project, So We Can Change Our Snapshots Of Code, In Each Change We Made, We Can Return To Specific Point, OR Push Specific Content To The Repository:

****

To *Get The Status Of Our Files IN Our Project*:

- We Run The Command: ***git status***



```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file1.txt
        file2.txt

nothing added to commit but untracked files present (use "git add" to track)
```

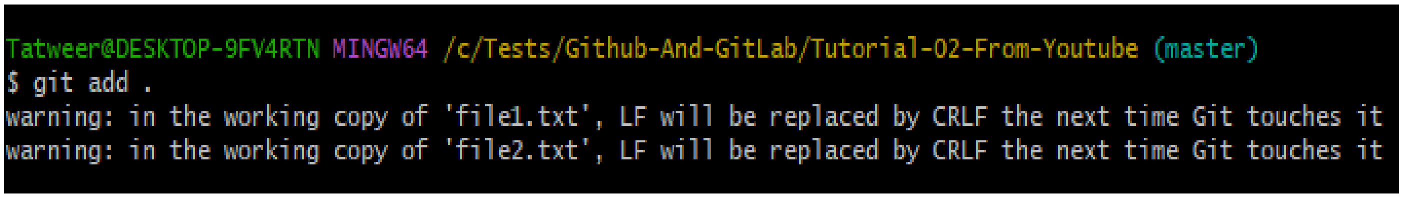
****

If We Want To *Add The Files To Repositories*, We Can:

- Run The Command: ***git add file1.txt file2.txt***
  - Here We Separate The Files By Space

Note 1: We Can Use Pattern To Add Files: ***git add *.txt***

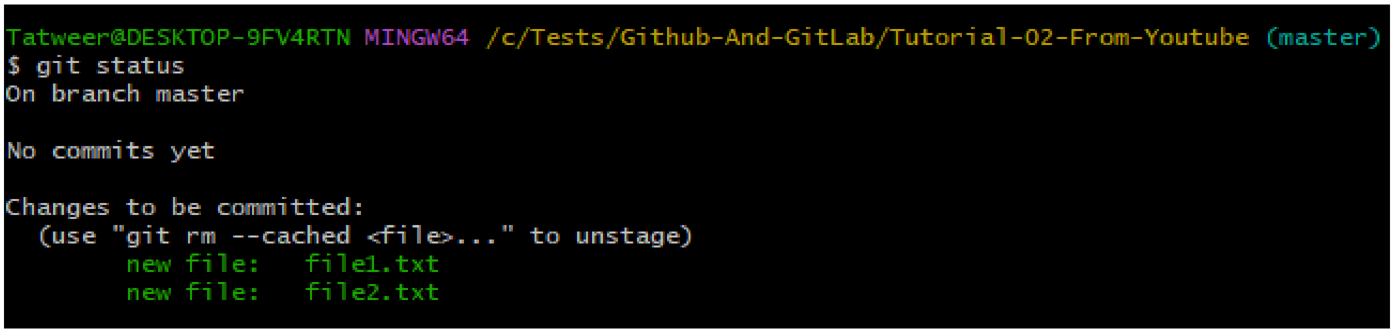
Note 2: We Can Use . To Add Files: ***git add .***



```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube (master)
$ git add .
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'file2.txt', LF will be replaced by CRLF the next time Git touches it
```

****

To Get The Status Of Changed Files:

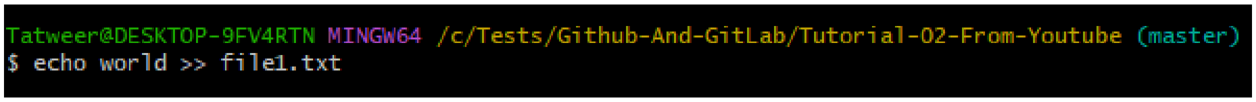


```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file1.txt
        new file:   file2.txt
```

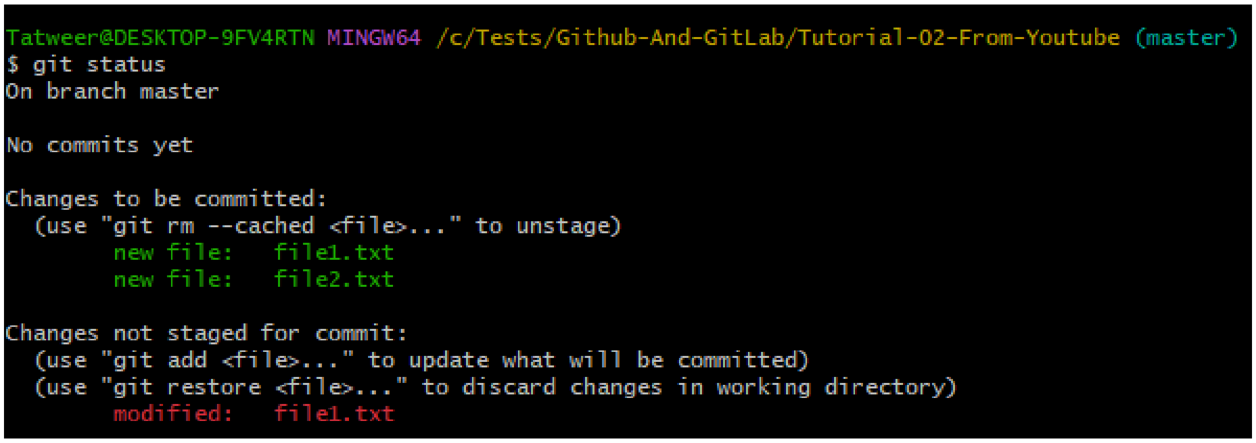
****

Note: The ***Green Color Of Files*** Mean They Are ***In Staging Area***.

****



```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube (master)
$ echo world >> file1.txt
```



```
Tatweer@DESKTOP-9FV4RTN MINGW64 /c/Tests/Github-And-GitLab/Tutorial-02-From-Youtube (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file1.txt
        new file:   file2.txt

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   file1.txt
```

****