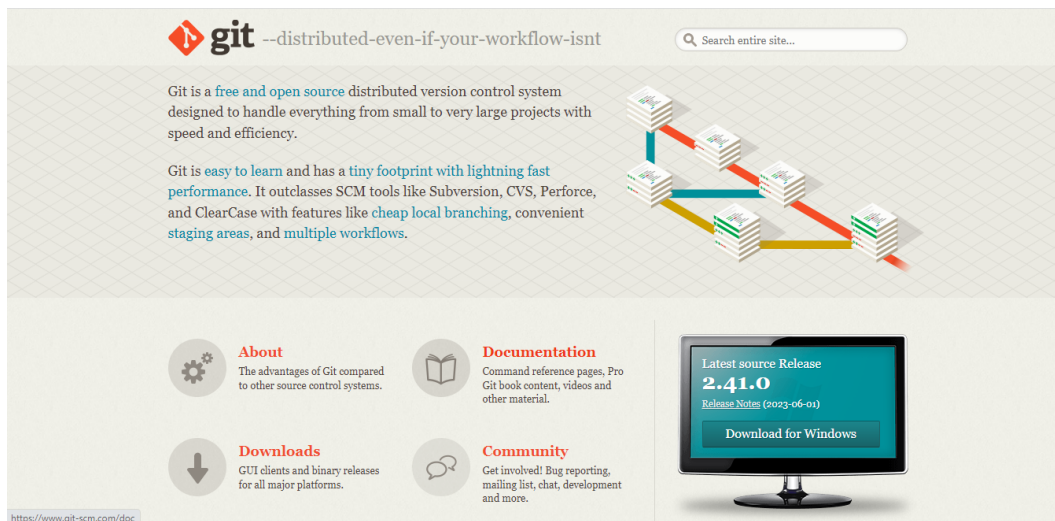


What is Git?

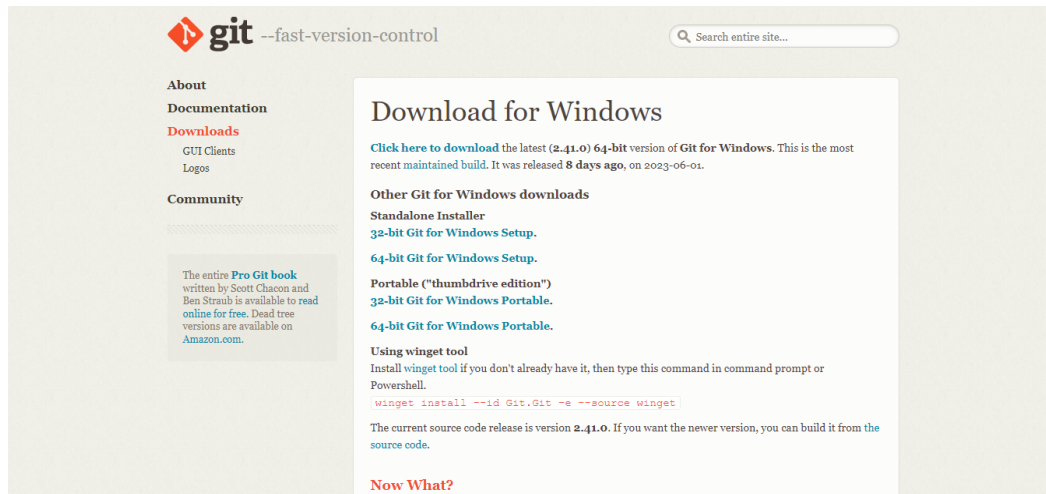
Git is an open-source distributed **version control system**. It is designed to handle minor to major projects with high speed and efficiency. It is developed to coordinate the work among the developers. The version control allows us to track and work together with our team members at the same workspace.

Download and Installing GIT

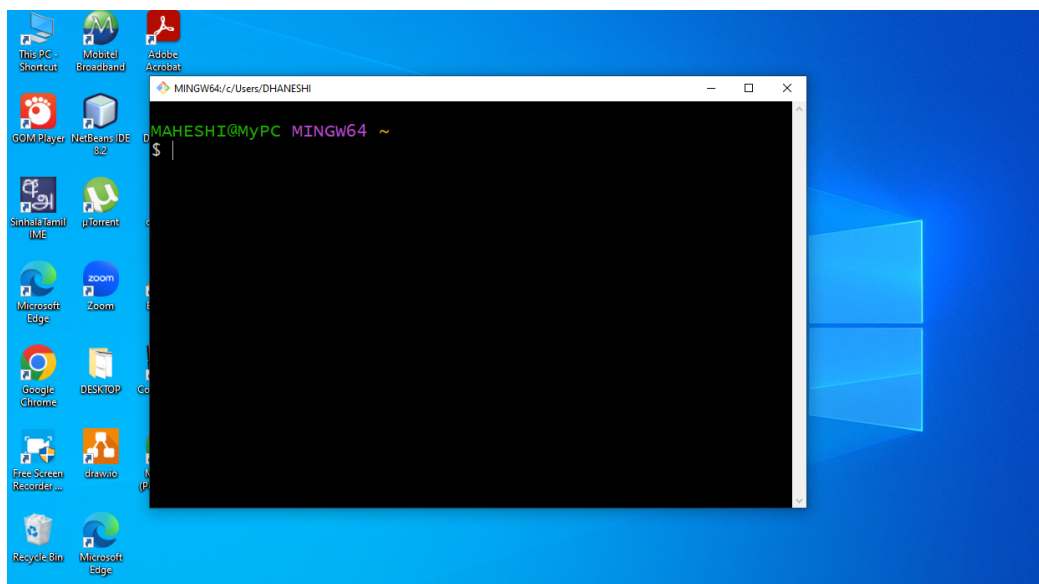
- Go to the link <https://www.git-scm.com/>
- Click **download for Windows**



- Download the GIT by selecting the suitable version for your pc.

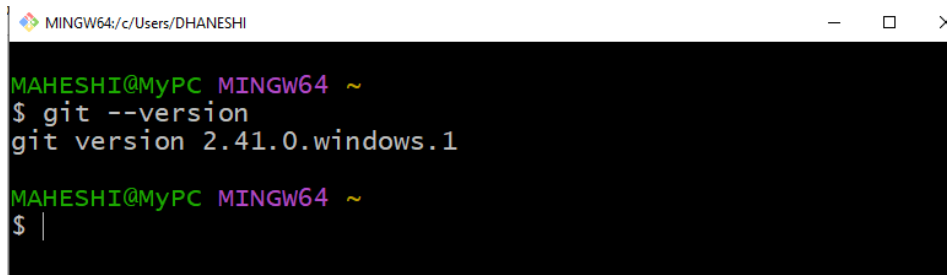


- Install the downloaded software by giving default settings in installing wizard.
- Open the **Command Prompt**
- Those who are using windows → **Git Bash** is also can be used



Check the GIT version

- Code : ***git --version***

A terminal window titled 'MINGW64:/c/Users/DHANESHI' with standard window controls. The prompt is 'MAHESHI@MyPC MINGW64 ~'. The command '\$ git --version' is entered, and the output is 'git version 2.41.0.windows.1'. The prompt returns to '\$ |'.

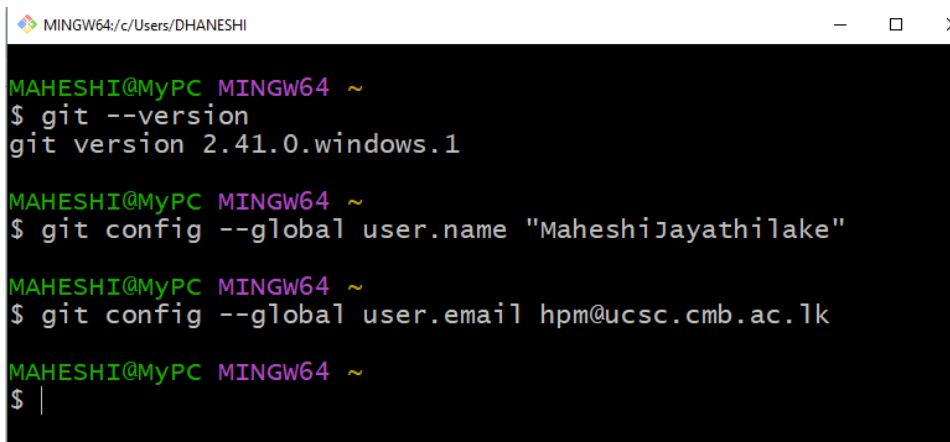
Configure GIT

- For name

Code : ***git config --global user.name "ABC Silva"***

- For email

Code: ***git config --global user.email abc@gmail.com***

A terminal window titled 'MINGW64:/c/Users/DHANESHI' with standard window controls. The prompt is 'MAHESHI@MyPC MINGW64 ~'. The command '\$ git --version' is entered, and the output is 'git version 2.41.0.windows.1'. The prompt returns to 'MAHESHI@MyPC MINGW64 ~'. The command '\$ git config --global user.name "MaheshiJayathilake"' is entered. The prompt returns to 'MAHESHI@MyPC MINGW64 ~'. The command '\$ git config --global user.email hpm@ucsc.cmb.ac.lk' is entered. The prompt returns to 'MAHESHI@MyPC MINGW64 ~'. The command '\$ |' is entered.

- If you are assigning these values for current repository, remove **--global**
- All these configurations are included into an editable text file. (.gitconfig file)
- Documentation : git-scm.com/docs/git-config

Verify the configuration

- Check the configuration list

Code : ***git config --list***

- If your configuration is done properly you can see your user name and email at the end of the list.

```

MAHESHI@MyPC MINGW64 ~
$ git config --global user.email hpm@ucsc.cmb.ac.lk

MAHESHI@MyPC MINGW64 ~
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-
bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=Maheshi Jayathilake
user.email=hpm@ucsc.cmb.ac.lk

MAHESHI@MyPC MINGW64 ~
$ |

```

- If you want to remove your name or email

Code : ***git config --global --unset user.name***
git config --global --unset user.email

Initializing a repository

Code : ***mkdir test*** ← ***directory name***

- **mkdir** makes a new directory.

Code : ***cd test***

- **cd** changes the current working directory
- To initialize the repository that we are working
Code : ***git init***

```

MAHESHI@MyPC MINGW64 ~
$ mkdir test

MAHESHI@MyPC MINGW64 ~
$ cd test

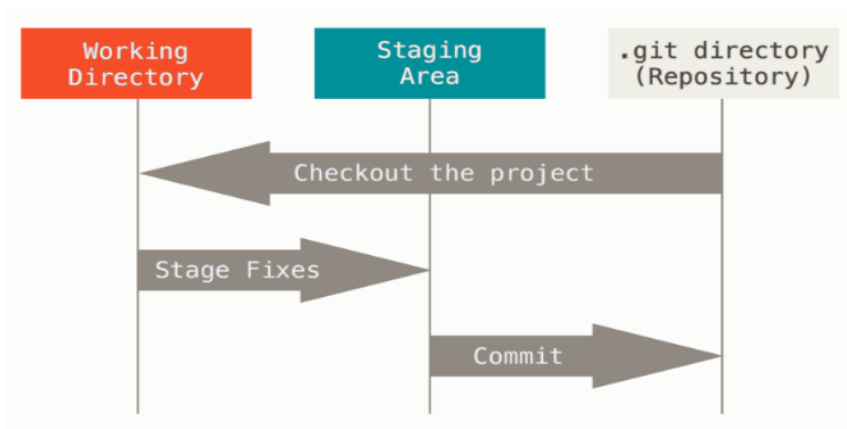
MAHESHI@MyPC MINGW64 ~/test
$ git init
Initialized empty Git repository in C:/Users/DHANESHI/test/.git/

MAHESHI@MyPC MINGW64 ~/test (master)
$ |

```

- Path to the relevant repository is underlined in **red** color

Workflow



Staging Area : The staging area is a file, generally contained in your Git directory, that stores information about what will go into your next commit. It is the middle ground between what you have done to your files (also known as the working directory) and what you had last committed (the HEAD commit).

Create new files

- Go to the relevant path and manually create a file.
- Then add it to the staging area.

Code : ***git add* file name**

Ex: *git add test.txt*

git add *.txt

git add -all

```
MAHESHI@MyPC MINGW64 ~/test (master)
$ git add *.txt

MAHESHI@MyPC MINGW64 ~/test (master)
$ |
```

- Check the status through *git status*

```
MAHESHI@MyPC MINGW64 ~/test (master)
$ git status
On branch master

No commits yet

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

MAHESHI@MyPC MINGW64 ~/test (master)
$ |
```

- By this it indicates one new file is added into the staging area.

Insert data to file

Code : **echo word to be inserted > file name**

Ex: echo hello > test.txt

- If adding words with symbols such as > or <

EX: echo "<html>" > test.txt

- Prevent replacing

EX: echo "<html>" >> test.txt

- Can be added to a list by entering a line.

EX: `echo "<head >
> <title>this is test </title>
> </head>" >> test.txt`

```
MAHESHI@MyPC MINGW64 ~/test (master)
$ echo hello >> test.txt

MAHESHI@MyPC MINGW64 ~/test (master)
$ |
```

Commit

- Save files in repository permanently

Code : `git commit -m "this is a commit"`

- -m used for inserting a message/description
- It is not a must. But recommended for collaboration.
 - Check the status through `git status`
 - If you add something to a file manually once it is saved also the status will be changed to modified. Then one more time use `"git add"` and `"commit"`

```
MAHESHI@MyPC MINGW64 ~/test (master)
$ git commit -m "Created a simple html file"
[master (root-commit) 1467e61] Created a simple html file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 test.txt
```

Remove and rename a file

Remove

- Code : `rm file name`

EX: `rm test.txt`

- Check the status through `git status`
- If you remove the file, the status will also be changed. Then one more time, use `"git add"` and `"commit"`

Rename

- Code : ***mv old filename new filename***
EX: ***rm test.txt main.txt***
- Check the status through ***git status***
- If you rename the file, the status will also be changed. Then one more time, use “***git add***” and “***commit***”

Git repository folder states

- **Tracked** : files that Git knows about and are added to the repository
- **Untracked** : files that are in your working directory, but not added to the repository

History

- To check all the commits.
Code : ***git log***
- Provide all the commits starting from recent.
- All commit has unique ID, message, Date/Time of committed, Author
- To exit from log press **q**.

```
MAHESHI@MyPC MINGW64 ~/test (master)
$ git log
commit 1f96204806aa7cca9ba59ae964f21015ceecd822 (HEAD -> master)
Author: MaheshiJayathilake <hpm@ucsc.cmb.ac.lk>
Date: Sat Jun 10 13:56:25 2023 +0530

    created a simple html file

commit 1467e61968856dedf20682b67410aac2ae2f31a0
Author: MaheshiJayathilake <hpm@ucsc.cmb.ac.lk>
Date: Sat Jun 10 13:55:09 2023 +0530

    Created a simple html file
```


GIT Branch

- Create a branch

Code : ***git branch branch name***

Ex : ***git branch task***

- Moving into branch

Code : ***git checkout task***

```
MAHESHI@MyPC MINGW64 ~/test (master)
$ git branch task

MAHESHI@MyPC MINGW64 ~/test (master)
$ git checkout task
Switched to branch 'task'

MAHESHI@MyPC MINGW64 ~/test (task)
$ |
```

- Create a new file and insert data to it.
- Now you are inside the **task** branch

```
MAHESHI@MyPC MINGW64 ~/test (master)
$ git checkout task
Switched to branch 'task'

MAHESHI@MyPC MINGW64 ~/test (task)
$ git add index.txt

MAHESHI@MyPC MINGW64 ~/test (task)
$ echo helo >> index.txt

MAHESHI@MyPC MINGW64 ~/test (task)
$ git add *.txt
warning: in the working copy of 'index.txt', LF will be replaced by CRLF t
he next time Git touches it

MAHESHI@MyPC MINGW64 ~/test (task)
$ git commit -m "Created a branch called task"
[task 2a4b9d2] Created a branch called task
3 files changed, 1 insertion(+)
create mode 100644 index.txt
```

Git merge

- Once you've completed work on your branch, it is time to merge it into the main branch.
- First go to master
EX: ***git checkout master***
- Merge files in to master
EX: ***git merge task -m "merge test file"***

```
MAHESHI@MyPC MINGW64 ~/test (task)
$ git checkout master
Switched to branch 'master'

MAHESHI@MyPC MINGW64 ~/test (master)
$ git branch
* master
  task

MAHESHI@MyPC MINGW64 ~/test (master)
$ git merge task -m "task finished by combined with master"
Updating 1f96204..2a4b9d2
Fast-forward (no commit created; -m option ignored)
 index.txt      | 1 +
 test2.txt      | 0
 test2.txt.txt  | 0
 3 files changed, 1 insertion(+)
 create mode 100644 index.txt
```

- To delete a branch

EX: ***git branch -d task***

```
MAHESHI@MyPC MINGW64 ~/test (master)
$ git branch -d task
Deleted branch task (was 2a4b9d2).

MAHESHI@MyPC MINGW64 ~/test (master)
$ git branch
* master
```

Activity

Do the Activity as a Group.

- **Create a repository providing any name**
- **Create branches individually.**
- **Create an HTML page “About me” (no need to use styles or dynamic)**
- **Merge all the branches together.**
- **Provide screenshots of the full process in a PDF format.**
- **Include below details into it.**
 - Cover page
 - Table of group members (Full name, index,email)
 - Screenshots of each process separately
 - Rename file with group number

**** Note that late submissions are not allowed***