



# Programming Fundamental

Programming Day - Week 01



## Introduction

Welcome to your first programming day. In this lab manual, we shall work together to learn and implement new programming concepts including setting up GitHub Account and using GitBash to remotely store your files. In addition, you will attempt to implement the complex programming task through hands-on experience.

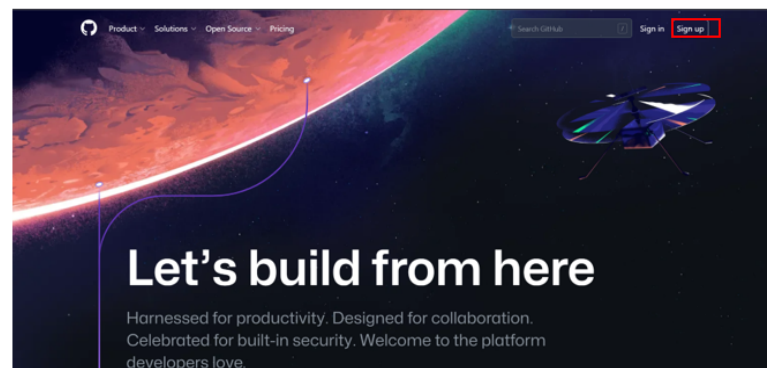
## Skills to be learned:

- Compiling and Executing a Program using MinGW
- Adding, Committing, and Pushing files remotely to the GitHub repositories

## Let's do some coding.

First Thing First, Let's create a Github Account.

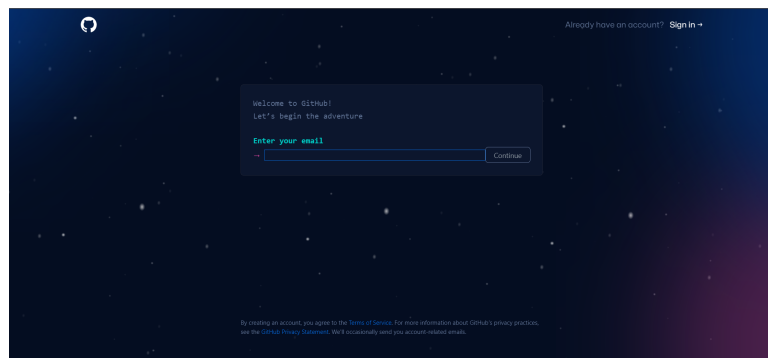
Go to <https://github.com/> and Click on **Sign Up** Button



You will need an email account for creating your GitHub account.

Enter the email address of your account.  
Choose a password and **complete the profile**.

**Note:** In case you already have a GitHub Account, just sign in using that account.



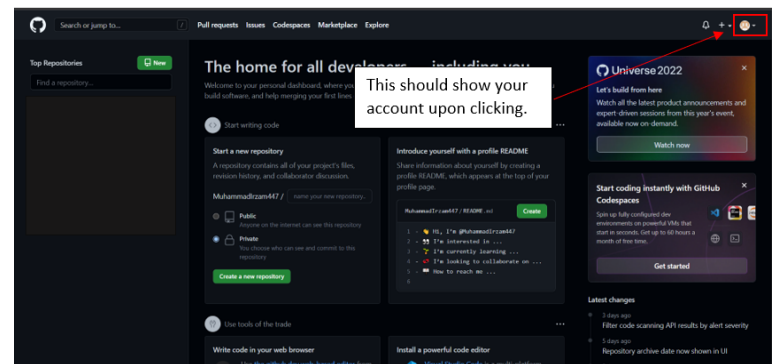


# Programming Fundamental

Programming Day - Week 01



Welcome to your GitHub Account.

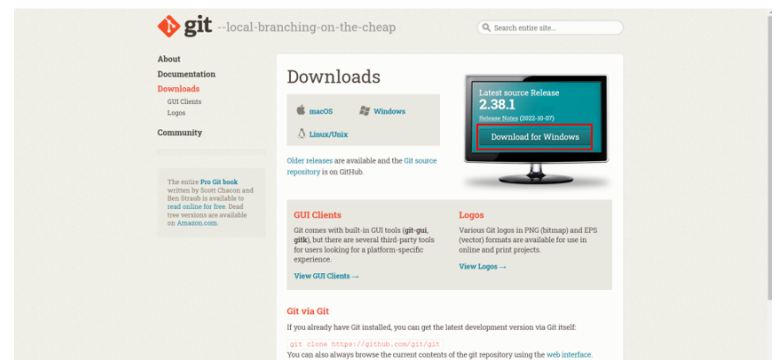


Now Let's Install GitBash.

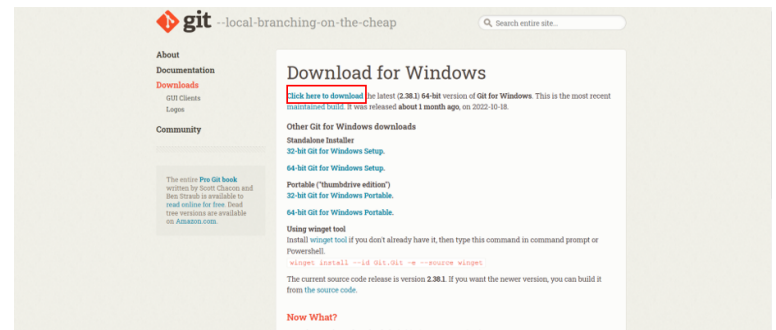
Visit

<https://git-scm.com/downloads/>

To download GitBash which will be used for uploading files to your git account.

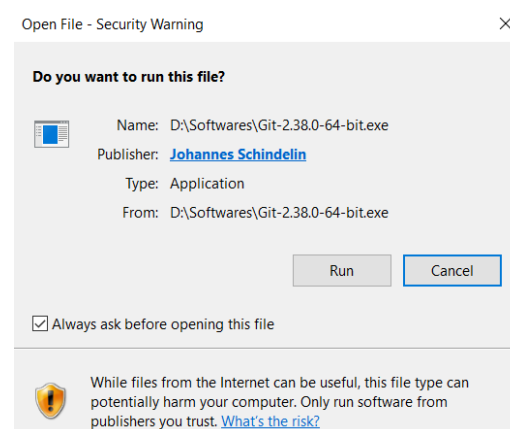


Install the Latest 64bit Setup for Windows.



Sit tight, it may take a minute to complete the download.

Upon successful completion, Double Click on the downloaded file and choose **Run** from the menu.



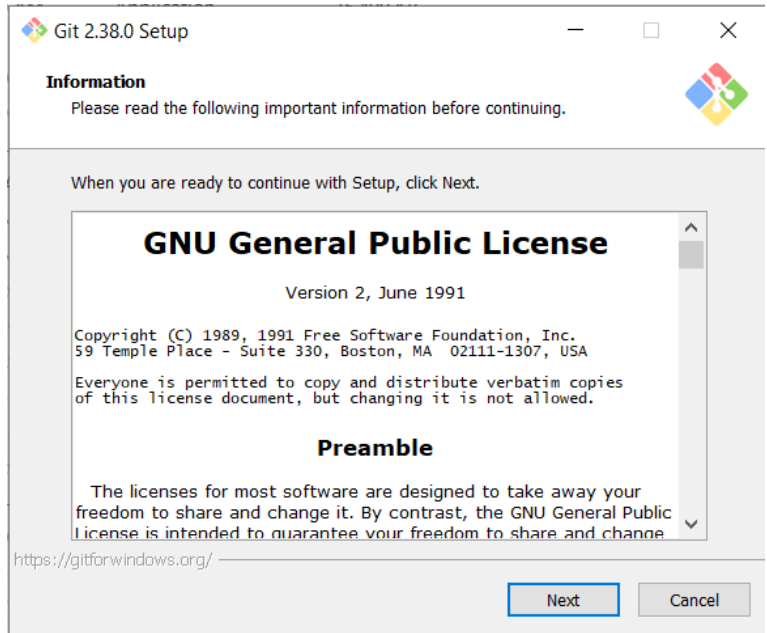


# Programming Fundamental

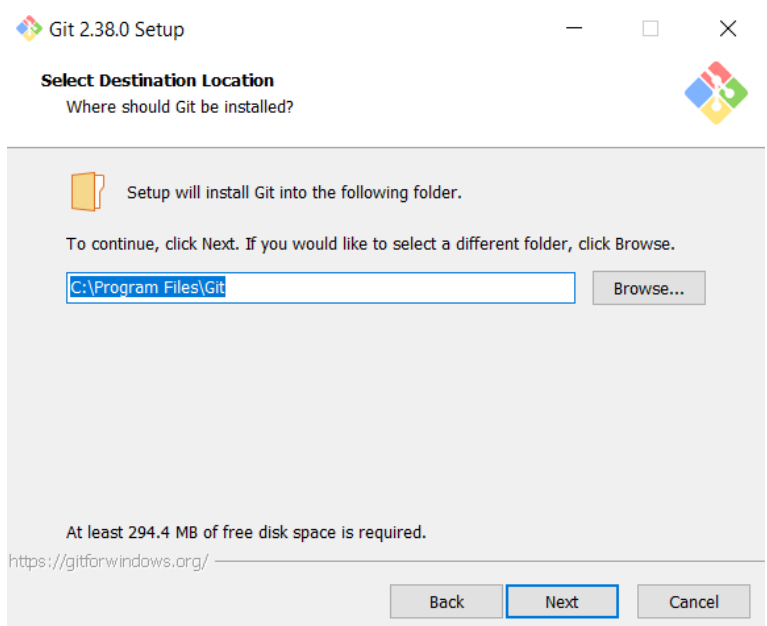
Programming Day - Week 01



Choose Next



Choose Next





# Programming Fundamental

Programming Day - Week 01



Choose Next

Git 2.38.0 Setup

**Select Components**  
Which components should be installed?

Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- ☐ Additional icons
  - ☐ On the Desktop
- ☒ Windows Explorer integration
  - ☒ Git Bash Here
  - ☒ Git GUI Here
- ☒ Git LFS (Large File Support)
- ☒ Associate .git\* configuration files with the default text editor
- ☒ Associate .sh files to be run with Bash
- ☐ Check daily for Git for Windows updates
- ☐ (NEW!) Add a Git Bash Profile to Windows Terminal

Current selection requires at least 294.4 MB of disk space.

<https://gitforwindows.org/>

Back Next Cancel

Choose Next

Git 2.38.0 Setup

**Select Start Menu Folder**  
Where should Setup place the program's shortcuts?

Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.

Git Browse...

☐ Don't create a Start Menu folder

<https://gitforwindows.org/>

Back Next Cancel



# Programming Fundamental

Programming Day - Week 01



Choose Next

Git 2.38.0 Setup

**Choosing the default editor used by Git**  
Which editor would you like Git to use?

Use Vim (the ubiquitous text editor) as Git's default editor

The [Vim editor](#), while powerful, [can be hard to use](#). Its user interface is unintuitive and its key bindings are awkward.

**Note:** Vim is the default editor of Git for Windows only for historical reasons, and it is highly recommended to switch to a modern GUI editor instead.

**Note:** This will leave the 'core.editor' option unset, which will make Git fall back to the 'EDITOR' environment variable. The default editor is Vim - but you may set it to some other editor of your choice.

<https://gitforwindows.org/>

Back Next Cancel

Select Let Git decide and Choose Next

Git 2.38.0 Setup

**Adjusting the name of the initial branch in new repositories**  
What would you like Git to name the initial branch after "git init"?

☒ **Let Git decide**  
Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project [intends](#) to change this default to a more inclusive name in the near future.

☐ **Override the default branch name for new repositories**  
**NEW!** Many teams already renamed their default branches; common choices are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:

This setting does not affect existing repositories.

<https://gitforwindows.org/>

Back Next Cancel



# Programming Fundamental

Programming Day - Week 01



Choose Next

Git 2.38.0 Setup

**Adjusting your PATH environment**

How would you like to use Git from the command line?

☐ Use Git from Git Bash only

This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

☒ Git from the command line and also from 3rd-party software

**(Recommended)** This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.

☐ Use Git and optional Unix tools from the Command Prompt

Both Git and the optional Unix tools will be added to your PATH.  
**Warning:** This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

<https://gitforwindows.org/>

Back Next Cancel

Choose Next

Git 2.38.0 Setup

**Choosing the SSH executable**

Which Secure Shell client program would you like Git to use?

☒ Use bundled OpenSSH

This uses ssh.exe that comes with Git.

☐ Use external OpenSSH

**NEW!** This uses an external ssh.exe. Git will not install its own OpenSSH (and related) binaries but use them as found on the PATH.

<https://gitforwindows.org/>

Back Next Cancel



# Programming Fundamental

Programming Day - Week 01



Choose Next

Git 2.38.0 Setup

**Choosing HTTPS transport backend**  
Which SSL/TLS library would you like Git to use for HTTPS connections?

☒ **Use the OpenSSL library**  
Server certificates will be validated using the ca-bundle.crt file.

☐ **Use the native Windows Secure Channel library**  
Server certificates will be validated using Windows Certificate Stores.  
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.

<https://gitforwindows.org/>

Back Next Cancel

Choose Next

Git 2.38.0 Setup

**Configuring the line ending conversions**  
How should Git treat line endings in text files?

☒ **Checkout Windows-style, commit Unix-style line endings**  
Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

☐ **Checkout as-is, commit Unix-style line endings**  
Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").

☐ **Checkout as-is, commit as-is**  
Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").

<https://gitforwindows.org/>

Back Next Cancel



# Programming Fundamental

Programming Day - Week 01



Choose Next

Git 2.38.0 Setup

## Configuring the terminal emulator to use with Git Bash

Which terminal emulator do you want to use with your Git Bash?

☒ **Use MinTTY (the default terminal of MSYS2)**

Git Bash will use MinTTY as terminal emulator, which sports a resizable window non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via ``winpty`` to work in MinTTY.

☐ **Use Windows' default console window**

Git will use the default console window of Windows (`"cmd.exe"`), which works with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

<https://gitforwindows.org/>

Back

Next

Cancel

Choose Next

Git 2.38.0 Setup

## Choose the default behavior of ``git pull``

What should ``git pull`` do by default?

☒ **Default (fast-forward or merge)**

This is the standard behavior of ``git pull``: fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.

☐ **Rebase**

Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.

☐ **Only ever fast-forward**

Fast-forward to the fetched branch. Fail if that is not possible.

<https://gitforwindows.org/>

Back

Next

Cancel





# Programming Fundamental

Programming Day - Week 01



Choose Next

Git 2.38.0 Setup

**Choose a credential helper**  
Which credential helper should be configured?

☒ **Git Credential Manager**  
Use the [cross-platform Git Credential Manager](#).  
See more information about the future of Git Credential Manager [here](#).

☐ **None**  
Do not use a credential helper.

<https://gitforwindows.org/>

Back Next Cancel

Choose Next

Git 2.38.0 Setup

**Configuring extra options**  
Which features would you like to enable?

☒ **Enable file system caching**  
File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.

☐ **Enable symbolic links**  
Enable [symbolic links](#) (requires the SeCreateSymbolicLink permission).  
Please note that existing repositories are unaffected by this setting.

<https://gitforwindows.org/>

Back Next Cancel

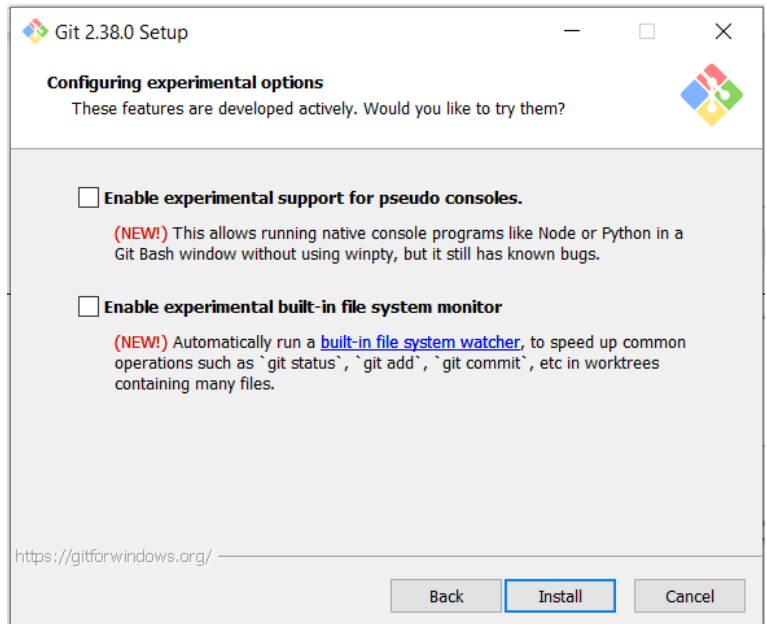


# Programming Fundamental

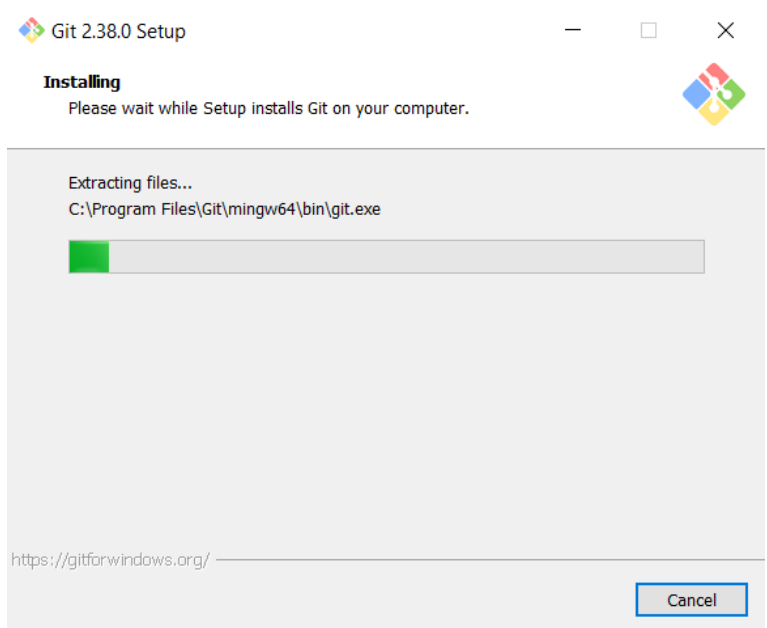
Programming Day - Week 01



Choose Next



Please Wait for the setup to install.





# Programming Fundamental

Programming Day - Week 01



Choose **Finish**

Git 2.38.0 Setup

## Completing the Git Setup Wizard

Setup has finished installing Git on your computer.

Click Finish to exit Setup.



☒ Launch Git Bash

☐ View Release Notes

Finish

GitBash cmd should appear on your screen upon successful completion.

MINGW64/c/Users/HP

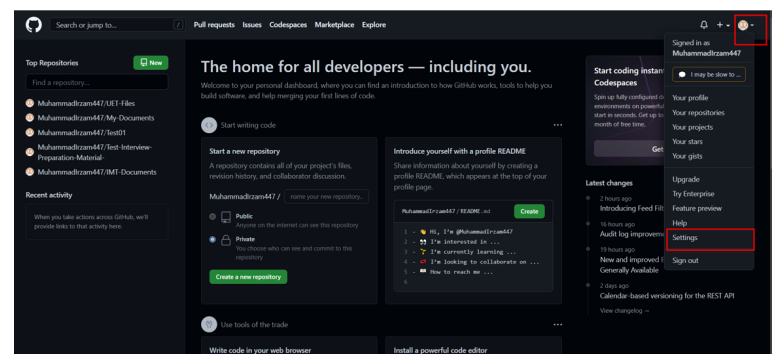
```
Muhammad Irzam@DESKTOP-BNMRNPI MINGW64 ~  
$
```

Great work. You have successfully installed GitBash.

Now, Let's create a personal access token to remotely update your GitHub account.

Go to your GitHub account on ([www.github.com/](https://www.github.com/))

Click on your **account button** and choose **settings**.



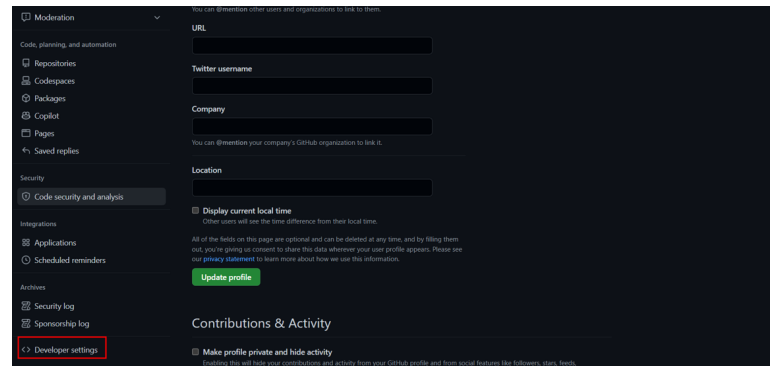


# Programming Fundamental

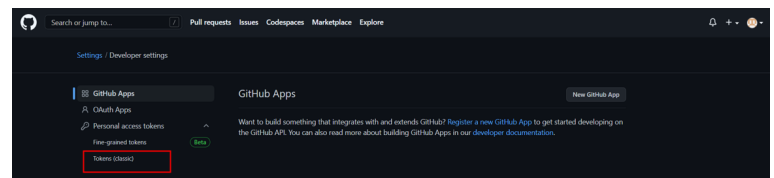
Programming Day - Week 01



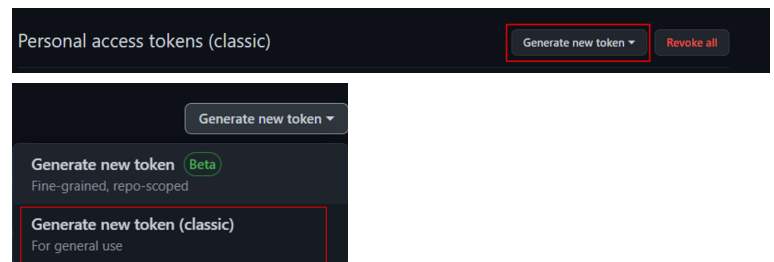
Scroll down and choose **Developer Settings**



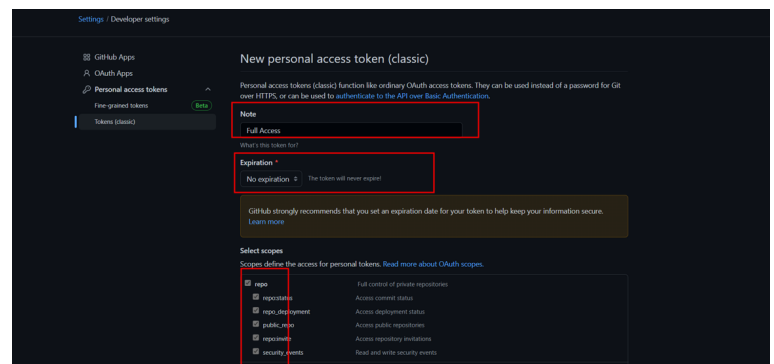
Click on **Personal Access Token** and choose **Classic**



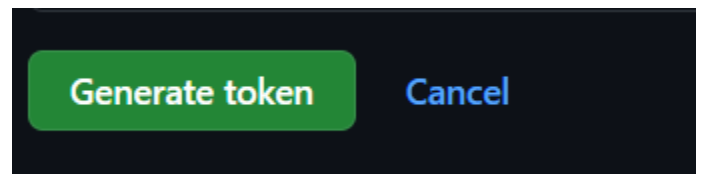
Select **Generate New Token**



Now,  
1. Insert a **note** of your desire.  
2. Set **expiration** to **No Expiration**  
3. check all the checkboxes



Once done with all the above steps, go ahead and click on **Generate Token**.





# Programming Fundamental

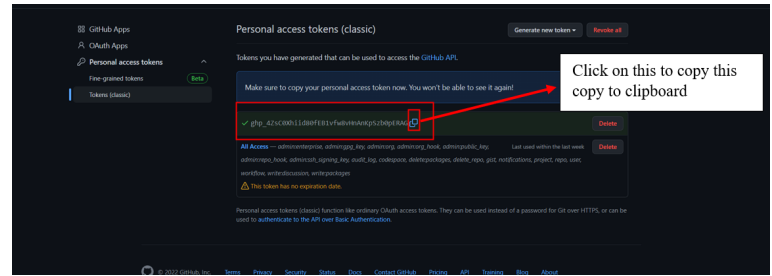
Programming Day - Week 01



Upon successful creation of a personal access token, the following screen should appear.

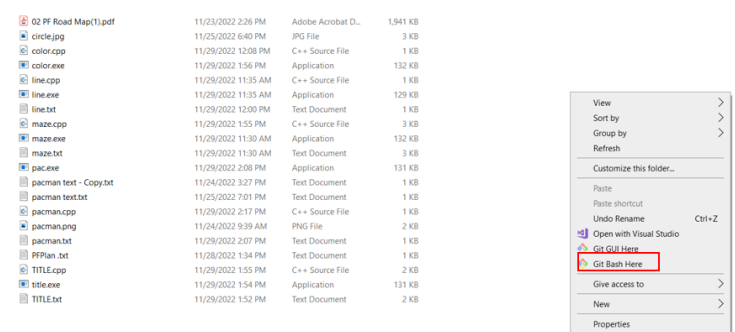
## Note:

Copy the token using the highlighted button and store it in a notepad file and save it for later use.

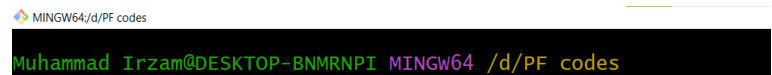


Now, Let's get back and Configure the GitBash so it would be connected to your GitHub Account.

Navigate to the folder where you have stored your codes and Right Click in the window and choose **GitBash here**.



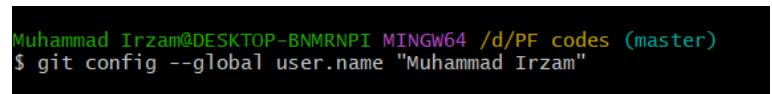
It will open the GitBash cmd in that directory



Write the following command in the cmd.

```
git config --global user.name "yourPCName"
```

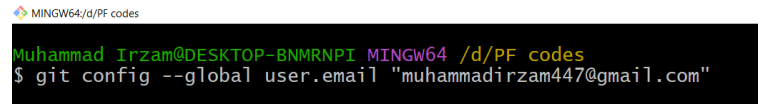
Replace the username **with your PC username**. (you can check it by pressing **Windows Key + L**) Attached is a working example:



Write the following command in the cmd.

```
git config --global user.email "yourEmailAccount"
```

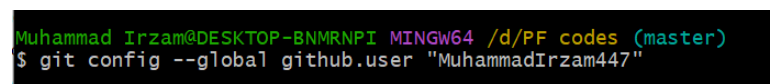
Replace the email account with **your own email account** that you are using for GitHub. Here is a working example:



Write the following command in the cmd.

```
git config --global github.user "myusername"
```

Replace the username with your own **GitHub**





# Programming Fundamental

Programming Day - Week 01



<b>username.</b> Here is a working example:	
<b>git config --global github.token "mytoken"</b>  <b>Note: It is the personal access token that you copied and stored in a notepad file earlier.</b>	<pre>\$ git config --global github.token ghp_EF2AgRjSqyLYMKtc2pz8s26</pre>
Note: These commands are a one-time task only and all the remaining steps are repeated for each new directory(folder) that you want to upload to your GitHub account.	
<b>git init</b>  Execute this command, to start your session.  <b>Note: The good thing is you need to execute this command only once to start your session in a specific directory.</b>	<pre>Muhammad Irzam@DESKTOP-BNMRNPI MINGW64 /d/PF codes \$ git init Initialized empty Git repository in D:/PF codes/.git/</pre>
<b>git add .</b>  This command is used to add all the files that are in your current working repository.	<pre>Muhammad Irzam@DESKTOP-BNMRNPI MINGW64 /d/PF codes (master) \$ git add .</pre>
<b>git commit -m "message"</b>  This command is used to make a commit (time stamp record of your file and changes in that) in your Git repository.  <b>NOTE:</b> Now, do not close this window and go to the web browser.	<pre>Muhammad Irzam@DESKTOP-BNMRNPI MINGW64 /d/PF codes (master) \$ git commit -m "message" [master (root-commit) 4e95968] message 3 files changed, 3 insertions(+) create mode 100644 file1.txt create mode 100644 file2.txt create mode 100644 file3.txt</pre>



# Programming Fundamental

Programming Day - Week 01

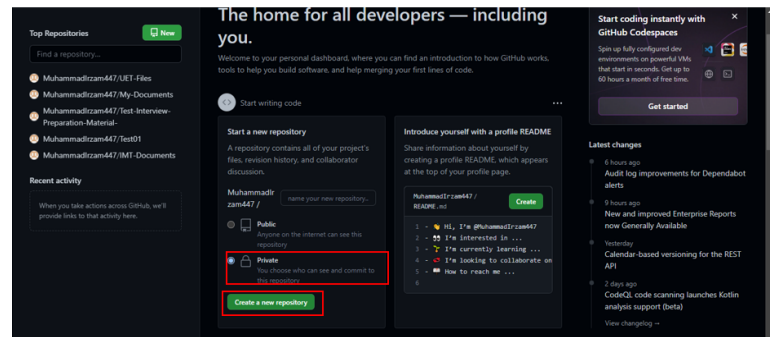


Now go to your GitHub Account([www.github.com/](https://www.github.com/)) and create a Repository.

A **Public Repository** can be accessed by anyone on the internet.

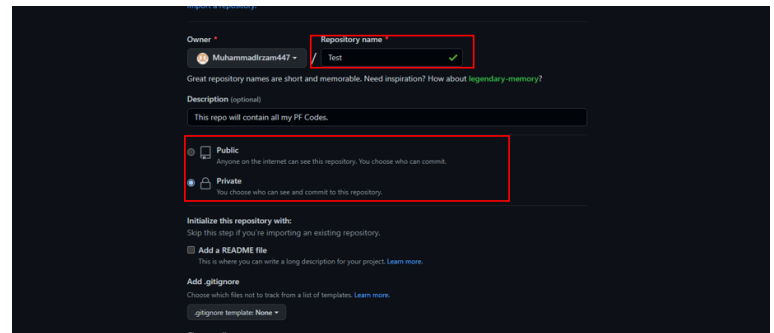
A **Private Repository** can only be accessed by you.

Look at the attached screen to create a repo.

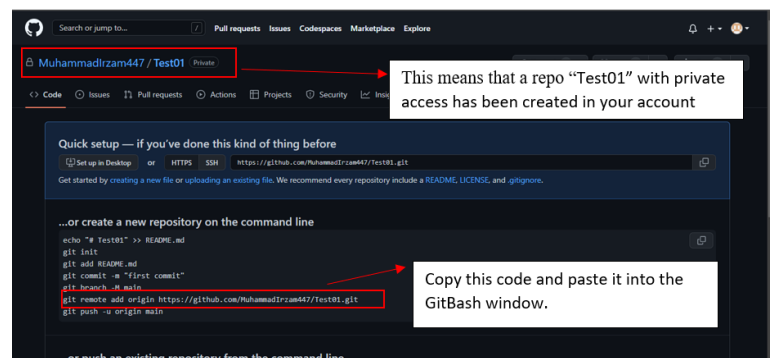


Give it a Name and Provide a description.

For now, choose **Private** and click on **Create Repository**.



On Successful creation, the attached screen should appear.



Now, **switch back to GitBash** and right click in the GitBash Window and Click on the **Paste** option and Hit **Enter**.

The command should look like this:  
**git remote add origin https://**

```
Muhammad Irzam@DESKTOP-BNMRNPI MINGW64 /d/PF codes (master)
$ git remote add origin https://github.com/MuhammadIrzam447/Test01.git
```



# Programming Fundamental

Programming Day - Week 01



Now, upload the files to the remote repo using the following command.

**git push -u origin master**

This command is used to upload all the committed files to your remote repository.

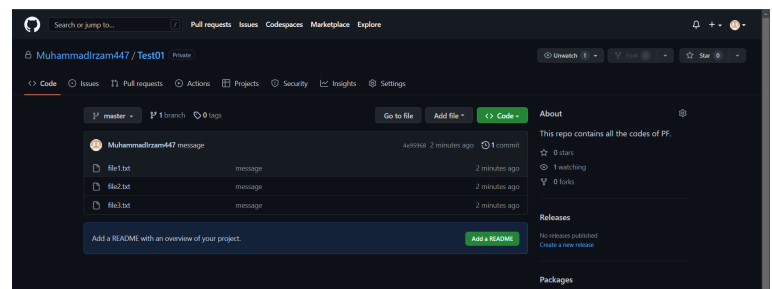
```
Muhammad Irzam@DESKTOP-BNMRNPI MINGW64 /d/PF codes (master)
$ git push -u origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (5/5), 315 bytes | 157.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/MuhammadIrzam447/Test01.git
 * [new branch] master -> master
branch 'master' set up to track 'origin/master'.
```

The first time you interact with GitBash, you may be asked to insert the token to verify it's you.

**Note: It is the personal access token that you copied and stored in a notepad file earlier.**

**Congratulations, you have created your GitHub Account and used GitBash to Upload the files to the remote repository.**

You can check that by visiting your GitHub account and choosing your created repo.



**Congratulations, you have successfully created your GitHub account and uploaded all the files in the PF Codes folder. Great Work Guys! You have just added another skill to your skill set.**

## Conclusion

Command	Description
git config --global user.email "email"	It's a <b>one time use only</b> command, that is used to connect the GitBash with your GitHub Account
git config --global user.name "name"	This is a <b>one time use only</b> command that is used to connect the user remotely with their GitHub Account.
git config --global github.user "myusername"	This is a one time use only command that is used to verify your <b>GitHub account user name</b> .
git config --global github.token "mytoken"	This is a <b>one time use only</b> command that is used to give access to the GitBash so it can access your account.
git init	This is used to initialize the session. It is used <b>every time you want to create a new repo</b> on the GitHub account.





# Programming Fundamental

Programming Day - Week 01



<b>git add .</b>	<p>This command is used to <b>add all the files</b> in the current working directory to the list of files that are to be uploaded on the GitHub account.</p> <p>There are various other versions on this command that you can find out on your own.</p>
<b>git commit -m "m"</b>	This command <b>makes the commit along with the message</b> that you want to associate with that commit.
<b>git remote add origin https://...</b>	This command associates all the <b>committed files</b> to the defined <b>origin repo</b> .
<b>git push -u origin master</b>	This command is used to <b>upload all the committed files</b> to the defined origin repo.
<b>Ctrl + L</b>	This is used to <b>clear the screen</b> in GitBash
<b>git ls-files</b>	This lists all the files in the current local directory.
<b>git clone https://...</b>	This command is used to <b>download the remote repo files</b> to your personal computer.

**Task 01(OP):** Upload all your NotePad files to a public repository on your GitHub account.

**Good Luck and Best Wishes !!**

**Happy Coding ahead :)**