

Using Git and Github

Following slides show you (step-by-step) how to create repository on github.com and push your files to github.com repository.

Prerequisites

You need following

1. You need to create an account at github.com
2. You need to install git tool to your machine
 - a. You can download the git from here: <https://git-scm.com/downloads>

Step 1: Create Repository

The screenshot shows the GitHub homepage. At the top, there is a navigation bar with links: Pull requests, Issues, Codespaces, Marketplace, and Explore. On the right side of this bar, there is a bell icon, a plus sign, and a profile icon. A red dashed box highlights the plus sign, and a red arrow points from it to a dropdown menu that is open. The dropdown menu contains the following options: New repository (highlighted in blue), Import repository, New codespace, New gist, New organization, and New project. Another red arrow points from the 'New repository' option to the 'Popular repositories' section. The 'Popular repositories' section displays two repositories: 'python_for_kids' and 'this_is_a_test_repo'. Below this, there is a section titled '6 contributions in the last year' which includes a calendar grid showing contributions for the months of March through February. The grid shows contributions for Monday, Wednesday, and Friday. A legend at the bottom right of the grid indicates the number of contributions with green squares: Less, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, More.

Pull requests Issues Codespaces Marketplace Explore

Overview Repositories 2 Projects Packages Stars

Popular repositories

python_for_kids Public This is a python materials I used to teach kids including my own :)

this_is_a_test_repo Public

Customize your pins

6 contributions in the last year Contribution settings

Mar Apr May Jun Jul Aug Sep Oct Nov Dec Jan Feb

Mon

Wed

Fri

Learn how we count contributions



Less More

From top right corner, Click + sign, and Click **New repository**

Step 2: Create Repository -- fill blanks

Create a new repository



A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner *  jmatai / Repository name * 

Great repository name Your new repository will be created as MyPythonProjects-. about bookish-parakeet?

Write the name of your project. For example, I decided that my project will be named “MyPythonProject”

Description (optional)

- ☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

- ☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

Check “Add a README file”

Add .gitignore

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: **None** ▼

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

License: **None** ▼

This will set  **main** as the default branch. Change the default name in your [settings](#).

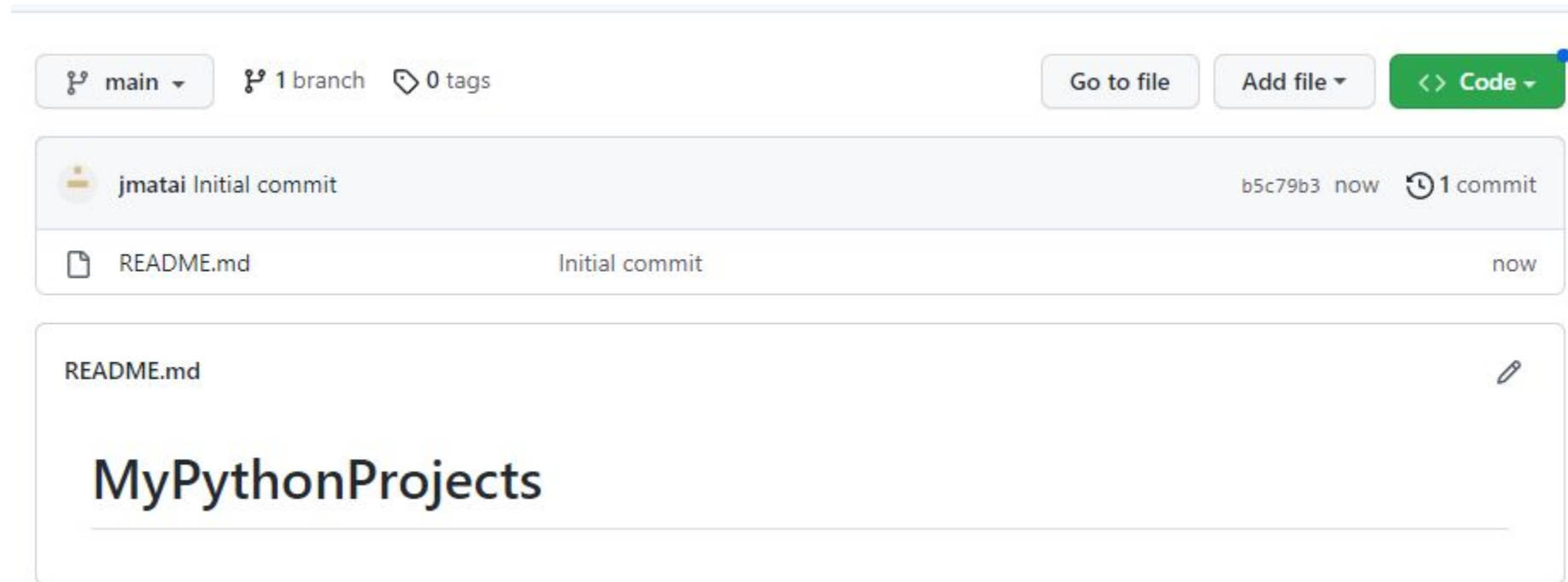
 You are creating a public repository in your personal account.

Create repository

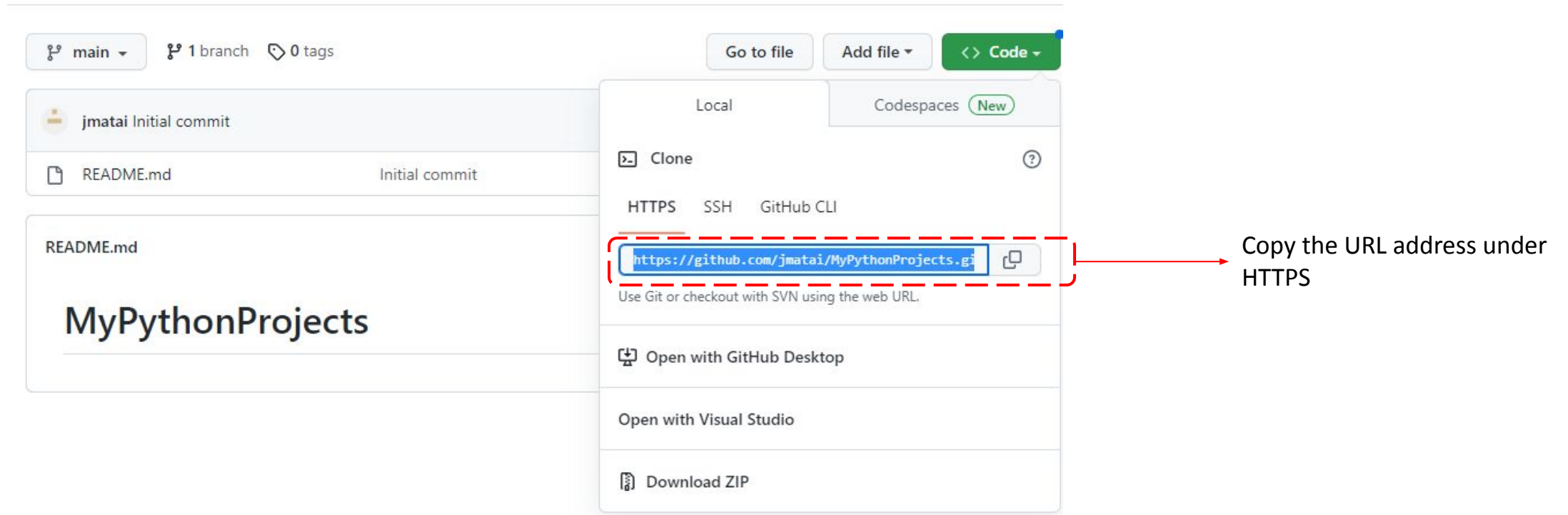
Click Create Repository

Step 2: Create Repository -- Created Repo

After clicking “Create Repository”, you see following screen



Step 3a: Clone Repository



The screenshot shows the GitHub interface for a repository named 'MyPythonProjects'. The repository is on the 'main' branch, has 1 branch, and 0 tags. The 'Code' button is open, showing options to clone the repository. The 'Clone' section is active, and the 'HTTPS' URL is highlighted with a red dashed box. A red arrow points from the highlighted URL to the text 'Copy the URL address under HTTPS'.

main 1 branch 0 tags

Go to file Add file <> Code

Local Codespaces New

Clone ?

HTTPS SSH GitHub CLI

<https://github.com/jmatai/MyPythonProjects.git> Copy

Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

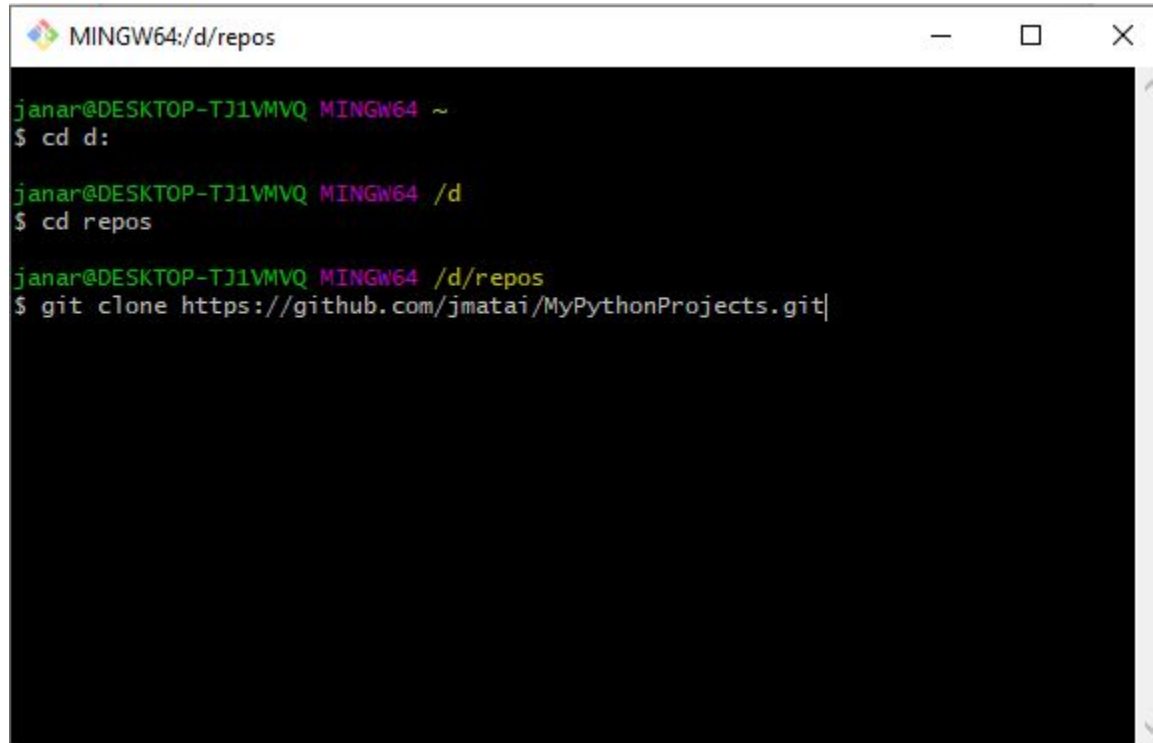
Open with Visual Studio

Download ZIP

Copy the URL address under HTTPS

Step 3b: Clone Repository

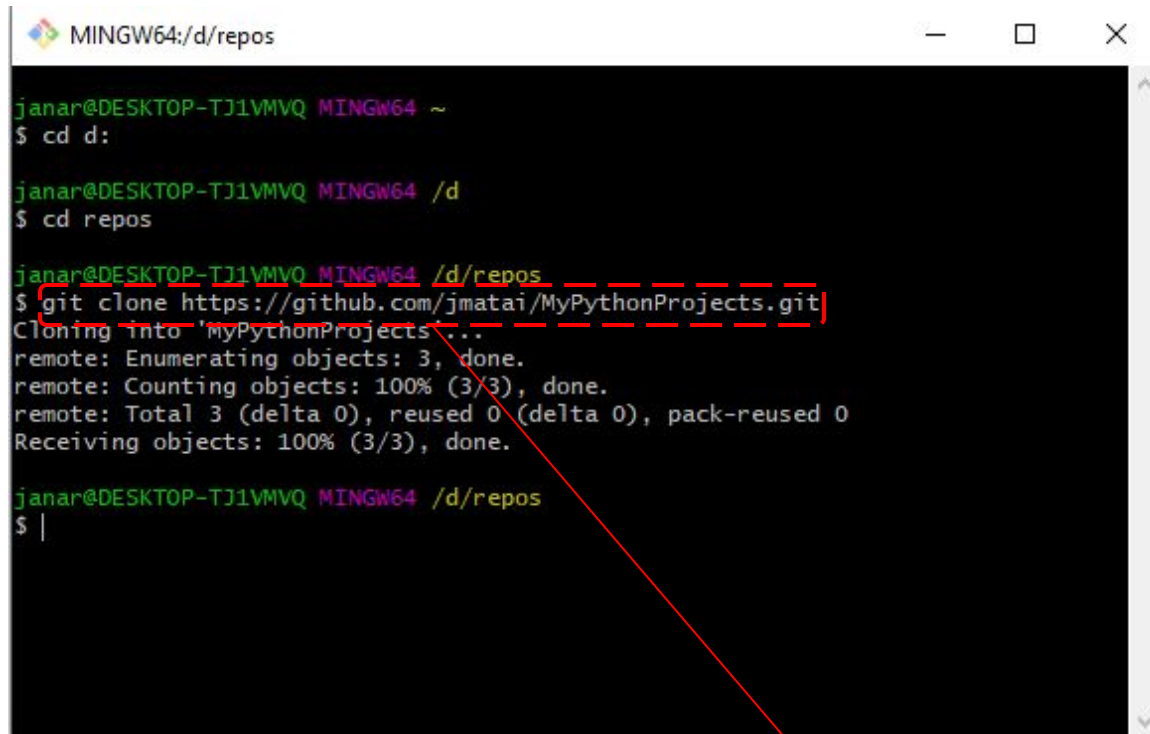
1. Open git tool
2. Go to the folder you want to store your repo
3. In my case, my repo is stored under D:\repos (it is shown as **/d/repos**)

A screenshot of a MINGW64 terminal window. The title bar shows 'MINGW64:/d/repos' and standard window controls. The terminal text shows the user navigating to the 'd' drive, then to the 'repos' directory, and finally running the 'git clone' command to fetch a repository from GitHub.

```
MINGW64:/d/repos  
janar@DESKTOP-TJ1VMVQ MINGW64 ~  
$ cd d:  
  
janar@DESKTOP-TJ1VMVQ MINGW64 /d  
$ cd repos  
  
janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos  
$ git clone https://github.com/jmatai/MyPythonProjects.git
```

Step 3c: Clone Repository

1. Open git tool
2. Go to the folder you want to store your repo
3. In my case, my repo is stored under D:\repos (it is shown as **/d/repos**)
4. *git clone https://github.com/jmatai/MyPythonProjects-.git* ⇒ Enter



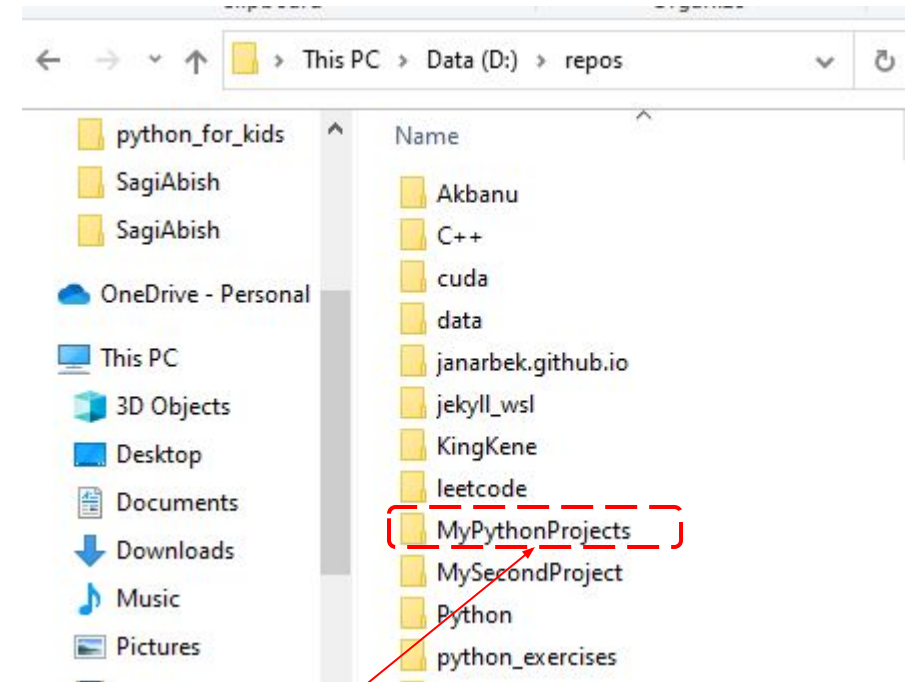
```
MINGW64:/d/repos

janar@DESKTOP-TJ1VMVQ MINGW64 ~
$ cd d:

janar@DESKTOP-TJ1VMVQ MINGW64 /d
$ cd repos

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos
$ git clone https://github.com/jmatai/MyPythonProjects-.git
Cloning into 'MyPythonProjects'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

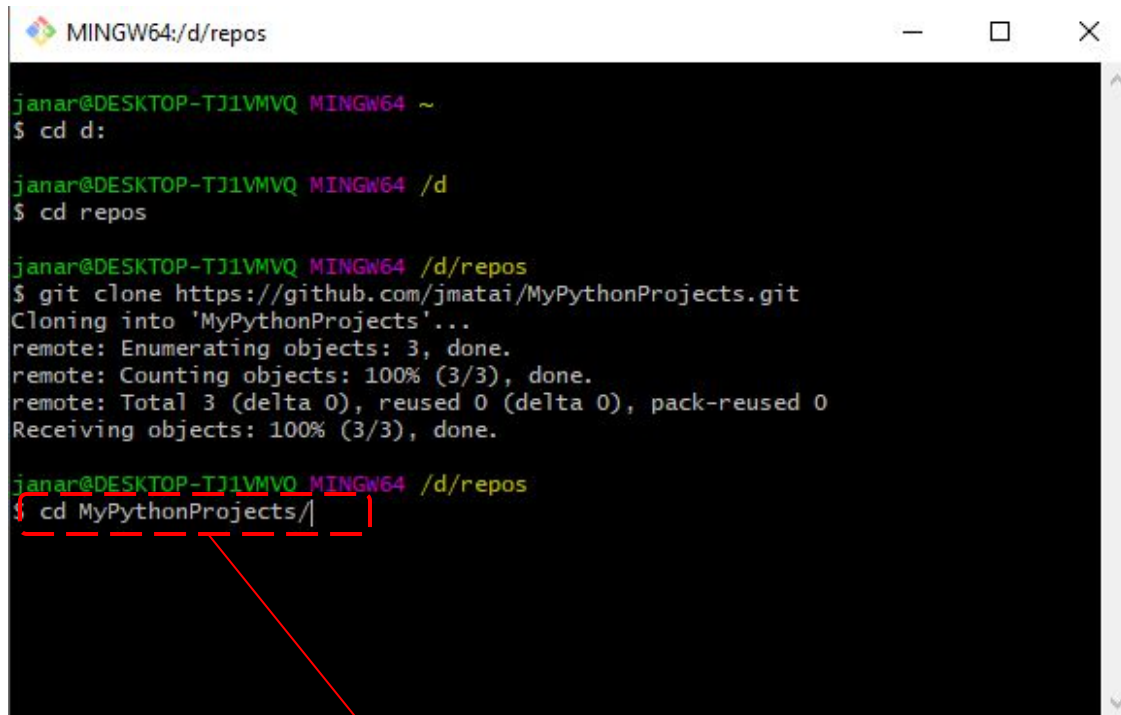
janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos
$ |
```



type this command and press Enter. This will copy the repo as shown on the right side

Step 4a: Modify the files in the Repo

- We are going to modify the files in the MyPythonProject push changes to git repo
- Steps are as follows
 - a. go to the “MyPythonProject” ⇒ We can do `cd MyPythonProject`
 - b. Modify the README.md file and save



```
MINGW64:/d/repos
janar@DESKTOP-TJ1VMVQ MINGW64 ~
$ cd d:

janar@DESKTOP-TJ1VMVQ MINGW64 /d
$ cd repos

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos
$ git clone https://github.com/jmatai/MyPythonProjects.git
Cloning into 'MyPythonProjects'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos
$ cd MyPythonProjects/
```

```
# MyPythonProjects
After adding this text, I will save it!
```

Example: type the command and press ENTER. , and modify the file README.md file as shown on the right

Step 4b: Push Files to Repo (git status)

- We are going to modify the files in the MyPythonProject push changes to git repo
- Steps are as follows
 - a. **git status** ⇒ This will show what files changed

```
MINGW64:/d/repos/MyPythonProjects
$ git clone https://github.com/jmatai/MyPythonProjects.git
Cloning into 'MyPythonProjects'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos
$ cd MyPythonProjects/

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$
```

git status shows what changed in the repo. For example, we modified README.md file

Step 4b: Push Files to Repo (git add)

- We are going to modify the files in the MyPythonProject push changes to git repo
- Steps are as follows
 - a. `git status` ⇒ This will show what files changed
 - b. `git add README.md` ⇒ This will add the file to staging area (or local repo)

```
MINGW64:/d/repos/MyPythonProjects
Your branch is up to date with 'origin/main'.

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:   README.md

no changes added to commit (use "git add" and/or "git commit -a")

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ git add README.md

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ |
```

`git add REAME.md` adds the modified README.md to git staging area (or local repo)

type `git status` again to see what changed. As you see README.md file changed to green. This means README.md added to the local repo.

Step 4b: Push Files to Repo (git commit)

- We are going to modify the files in the MyPythonProject push changes to git repo
- Steps are as follows
 - a. `git status` ⇒ This will show what files changed
 - b. `git add README.md` ⇒ This will add the file to staging area (or local repo)
 - c. **`git commit -m 'Updated the README.md file'`** ⇒ commit the changes to repo.

```
MINGW64:/d/repos/MyPythonProjects
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ git push origin main
Everything up-to-date

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ git commit -m 'Updated the README.md file'
```

`git commit -m "message"` pushes the message into our history of remote repo

Step 4b: Push Files to Repo (git push)

- We are going to modify the files in the MyPythonProject push changes to git repo
- Steps are as follows
 - a. `git status` ⇒ This will show what files changed
 - b. `git add README.md` ⇒ This will add the file to staging area (or local repo)
 - c. `git commit -m 'Updated the README.md file'` ⇒ commit the changes to repo.
 - d. `git push origin main`

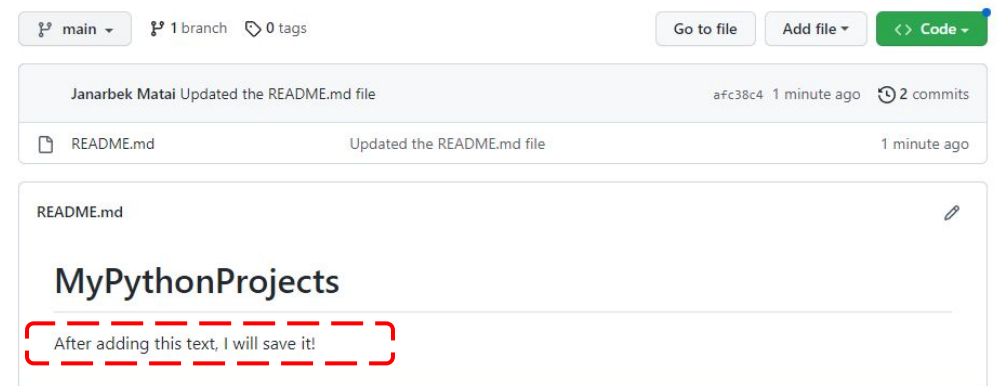
```
MINGW64:/d/repos/MyPythonProjects

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ git commit -m 'Updated the README.md file'
[main afc38c4] Updated the README.md file
1 file changed, 3 insertions(+), 1 deletion(-)

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 317 bytes | 317.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/jmatai/MyPythonProjects.git
 b5c79b3..afc38c4  main -> main

janar@DESKTOP-TJ1VMVQ MINGW64 /d/repos/MyPythonProjects (main)
$ |
```



git push command pushes the local changes with commit message to remote repo. Now, you can see your changes by refreshing your repo as shown on the right!

Summary

- After you create your repo, you can continuously use that repo
- You can use following set of commands (git is huge). These commands are good enough for now.
 - a. `git status` ⇒ This will show what files changed
 - b. `git add README.md` ⇒ This will add the file to staging area (or local repo)
 - c. `git commit -m 'Updated the README.md file'` ⇒ commit the changes to repo.
 - d. `git push origin main`
- Next slide, we are going to show how to add your Python files to the repo

PyCharm

In following slides, I will show how to download and setup python development environment using PyCharm

Install PyCharm Community Edition

1. Download PyCharm Community edition


a. The link is here:

<https://www.jetbrains.com/pycharm/download/#section=windows>

2. Install PyCharm Community Edition

PyCharm

Coming in 2023.1 What's New Features Learn Pri



Version: 2022.3.3
Build: 223.8836.34
8 March 2023

[System requirements](#)
[Installation instructions](#)
[Other versions](#)
[Third-party software](#)

Download PyCharm

[Windows](#) [macOS](#) [Linux](#)

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

[Download](#) [.exe](#) ▼

Free 30-day trial available


Community

For pure Python development

[Download](#) [.exe](#) ▼

Free, built on open-source

1. Click Download
2. Click .exe to install



Get the Toolbox App to download PyCharm and its future updates with ease

PyCharm with Git

Go to this youtube tutorial to learn how to install and use PyCharm with git: <https://www.youtube.com/watch?v=-Shjg0Q6-j8>