

GitHub & Git

助教：余嘉袖、蘇育琳、謝名評



Outline



1 認識 Git / GitHub 與基本環境設定

2 Git 版本控制基礎指令與檔案管理實務

3 同步儲存庫

4 協作

5 分支與合併

6 Git GUI (GitHub Desktop)

1-1 何謂Git



- Git 是一種分散式版本的版本控制系統

- e.g.:

假設你有一個 resume 的目錄(存放面試者的資料)





➤ 分散式系統：

每台使用者的電腦都擁有完整的版本庫，因此即使沒有伺服器或沒有網路，也能正常做版本控制。

➤ 版本：

這每一個「resume 目錄的狀態變化」，不管是新增或刪除檔案，亦或是修改檔案內容，都稱之為一個「版本」，例如上一張投影片圖例的版本 1 ~ 5。

➤ 版本控制系統：

指會幫你記錄這些所有的狀態變化，並且可以像搭乘時光機一樣，隨時切換到過去某個「版本」時候的狀態。

1-2 Git處理檔案方式



➤ e.g.:

Git:

像拍照 (snapshot) 一樣，在每次版本變化的時候，Git 會更新並記錄整個目錄跟檔案的樹狀結構。



1-3 學習Git的理由



- 透過版本控制系統，可以清楚的記錄每個檔案是誰在什麼時候加進來、什麼時候被修改或刪除。Git 就是一種版本控制系統，也是目前業界最流行的版本控制系統
- e.g.:

整理或備份檔案範例



1-4 何謂GitHub&GitHub註冊

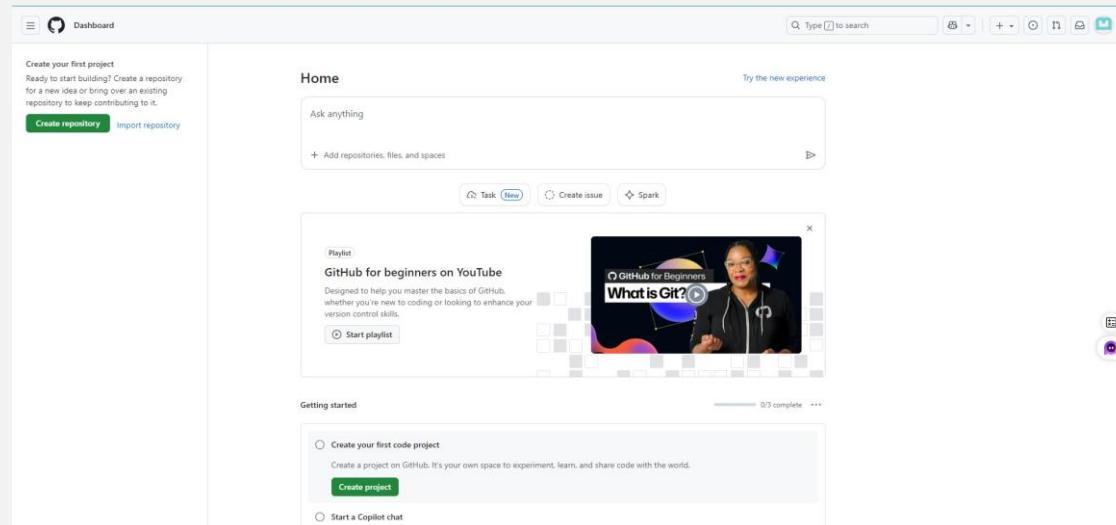


➤ GitHub 是一個基於 Git 的「線上程式碼託管平台」，用來存放、分享、管理程式碼。簡單說，就是把你的 Git 專案放到雲端，方便協作、備份、與別人一起開發。

➤ **GitHub註冊：**

進入 GitHub 官方網站，點擊「Sign up」註冊帳號；已註冊過者，點擊「Sign in」登入帳號

➤ GitHub官方網站畫面：
(已註冊過的)



1-5 Git VS GitHub



項目	Git	GitHub
是什麼？	一套 版本控制工具 （在你的電腦本地端運作）	一個 雲端程式碼託管平台 （用來放 Git 專案）
主要用途	管理程式碼版本、歷史、分支、合併	上傳 Git 專案、分享、備份、協作、多人開發
是否需要網路？	不需要（離線也能完整運作）	需要網路才能存取倉庫
安裝位置	安裝在本機電腦	網站平台 (github.com)

1-5 Git VS GitHub



項目	Git	GitHub
存放方式	本地端保存完整版本庫	雲端保存遠端版本庫
核心特色	分散式：每人電腦都有完整備份	集中協作：大家一起連同一個雲端倉庫
開發者支援功能	版本控制、快照、回溯、合併	Issue、Pull Request、Code Review、Wiki、CI/CD (GitHub Actions)



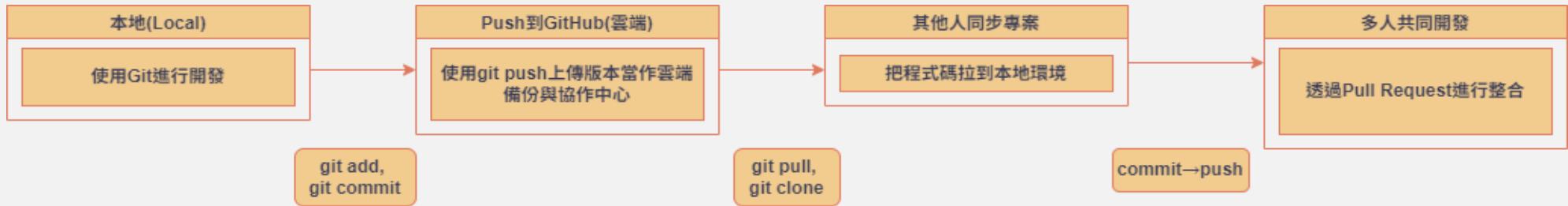
Git × GitHub 協作流程：

- I. 在本地 (Local) 使用 Git 進行開發
- II. Push 到 GitHub (雲端)
- III. 其他人從 GitHub 同步專案
- IV. 多人共同開發

1-6 Git & GitHub協作關係



Git × GitHub 協作流程圖：



1-7 Git安裝-Windows



Step1.請到 Git官方網站 點擊 “Download for Windows”

The screenshot shows the official Git website (git-scm.com). The header features the Git logo and the tagline “--local-branching-on-the-cheap”. A search bar and a moon icon for dark mode are also present. The main content area includes a brief introduction about Git being a free and open source distributed version control system, followed by a statement that it is lightning fast and has a large ecosystem. To the right is a diagram illustrating the distributed nature of Git with multiple repositories connected by bidirectional arrows. Below this, there are several navigation links: “About”, “Tools”, “Install”, “Learn”, “Reference”, and “Community”. On the right side, a monitor icon displays the latest source release “2.52.0” with a “Release Notes (2025-11-17)” link and a prominent “Download for Windows” button, which is highlighted with a red box. At the bottom, there are links for “Windows GUIs” and “Tarballs”.

- About**
Git's performance and ecosystem
- Tools**
Command line tools, GUIs, and hosting services
- Install**
Binary releases for all major platforms.
- Learn**
Pro Git book, videos, tutorials, and cheat sheet
- Reference**
Git's reference documentation
- Community**
Get involved! Bug reporting, mailing list, chat, development and more.



Step2.請在Install頁面的Windows區塊點擊 “Git for Windows/x64 Setup” 或 “Git for Windows/ARM64 Setup” 進行安裝

The screenshot shows the Git website's 'Install' page. The 'Windows' tab is active. A red box highlights the 'Git for Windows/x64 Setup' link. Below it, another link 'Git for Windows/ARM64 Setup.' is also visible. Other tabs include 'macOS', 'Linux', and 'Build from Source'. To the left, there's a sidebar with links to 'About', 'Learn', 'Tools', 'Reference', 'Install' (which is red), and 'Community'. A small note about the 'Pro Git book' is also present.



- “Git for Windows/x64 Setup” 或 “Git for Windows/ARM64 Setup” 選擇：

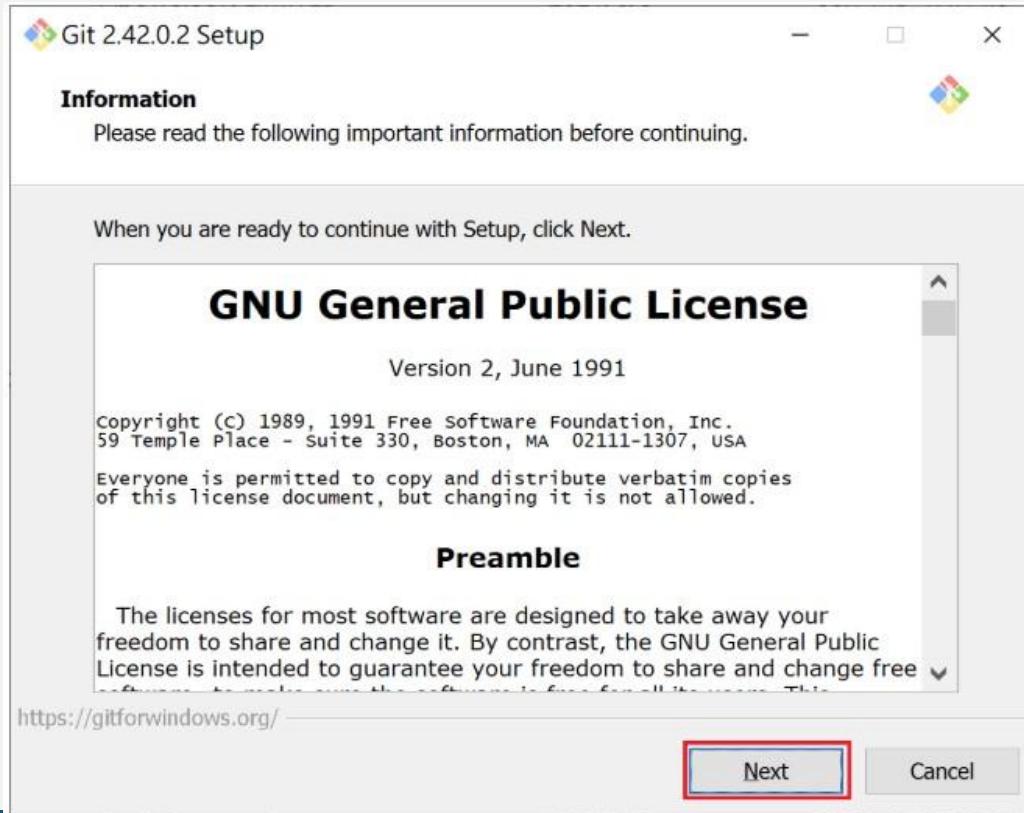
先確認自己PC/筆電的CPU架構：設定 → 系統 → 關於 → 裝置規格 → 系統類型 (System Type) (大多數是x64版)

System type 64-bit operating system, x64-based processor

1-7 Git安裝-Windows



Step3.安裝過程中原則上維持預設值，按“Next”即可



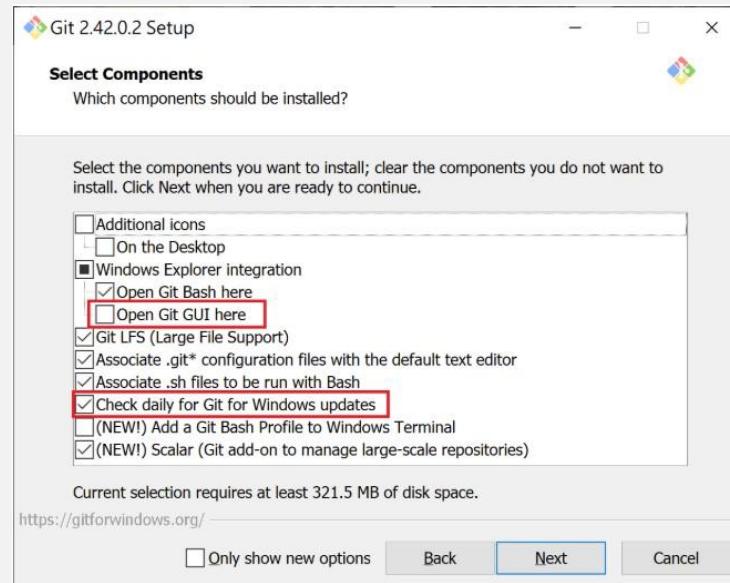
1-7 Git安裝-Windows



➤ 安裝過程中有幾個需要注意的部分：

(1)取消勾選 **Git GUI 工具**

(2)勾選 **check daily for git for windows updates**



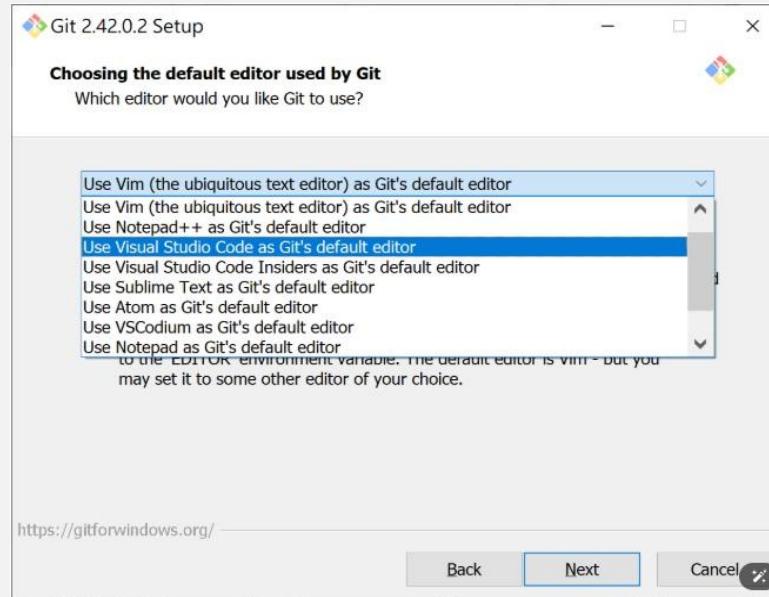
1-7 Git安裝-Windows



- 安裝過程中有幾個需要注意的部分：

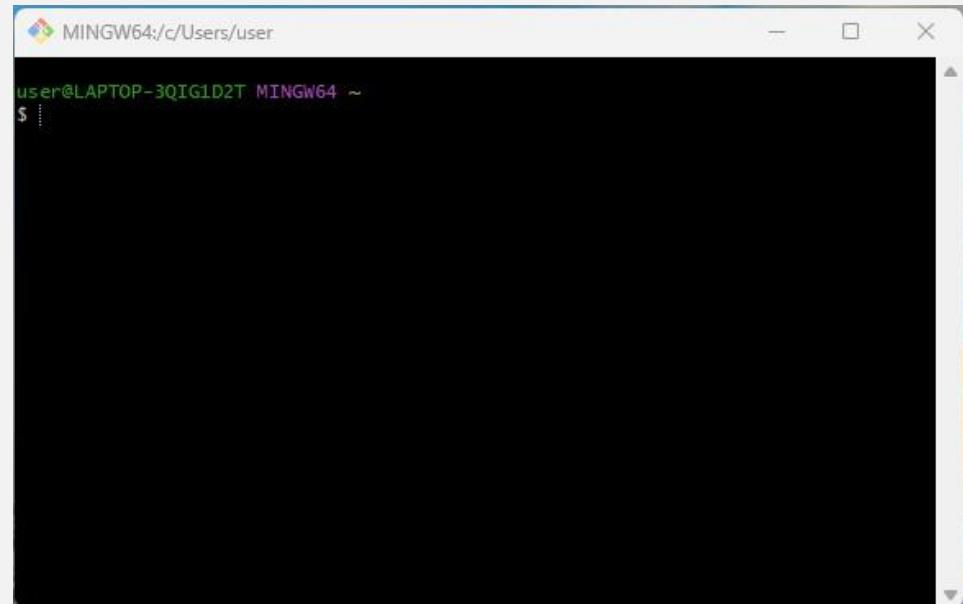
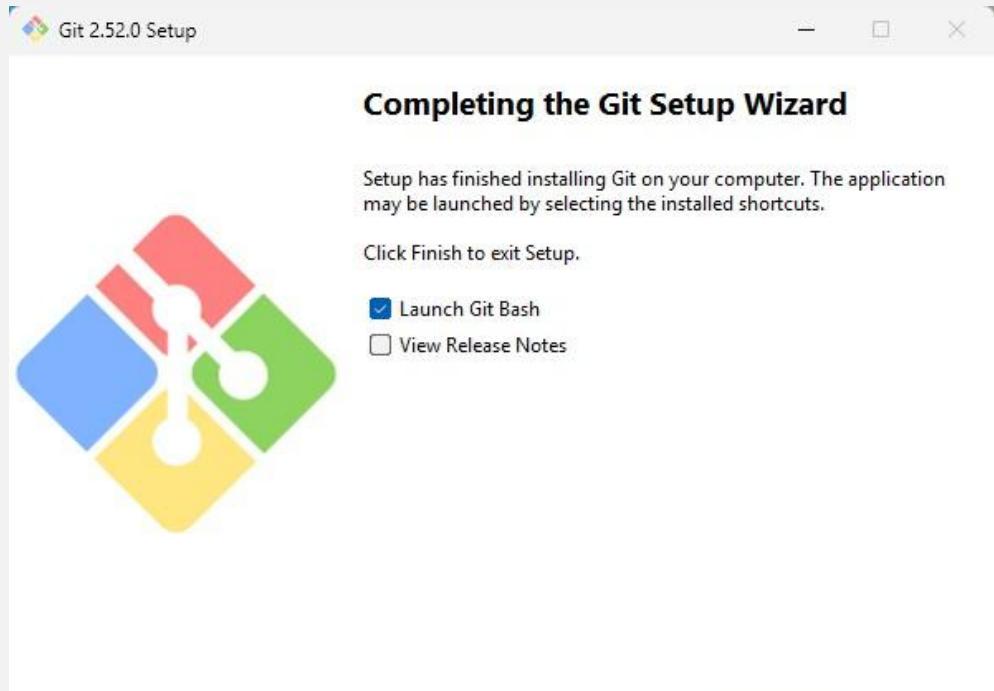
選擇 Git 預設的編輯器，Git 預設使用的編輯器是 Vim =Visual Improved

e.g.可更換為 **Use Visual Studio Code as Git's default editor**





Step4.安裝完成，啟動Git Bash(預設是Vim編輯器)





Step5.確認Git是不是有安裝起來(在VSCode編輯器)

- 在 VS Code 裡新增終端：

```
git --version
```

```
PS D:\網路資料庫程式設計(大學部助教課程)碩一上\第十四週Git&GitHub教材(帶大三學弟妹上課)\My Project> git --version
git version 2.52.0.windows.1
```



Step1.請按cmd+空白鍵→呼叫Spotlight→輸入終端機打開它

Step2.安裝Git:

Xcode-select –install

A screenshot of a macOS terminal window titled 'papayastudio ~ - zsh - 80x24'. The window shows the command 'xcode-select --install' being typed at the prompt. The text in the terminal is as follows:

```
Last login: Sat May 18 14:17:08 on ttys000
papayastudio@PapayadeMac-Studio ~ % xcode-select --install
```



Step1.請到 Homebrew官方網站 複製Install Homebrew那一行：

```
/bin/bash -c "$(curl -fsSL
```

```
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

The screenshot shows the official Homebrew website (<https://brew.sh>). At the top, there's a logo of a beer mug and the word "Homebrew" in a stylized font. Below it, a sub-header reads "The Missing Package Manager for macOS (or Linux)". There's a search bar and a language dropdown set to "English". A "Fork me on GitHub" button is visible in the top right corner.

In the center, there's a large heading "Install Homebrew". Below it, a code block contains the command `$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"`. This command is highlighted with a red rectangular box. To the left of the command, there's a note: "Paste that in a macOS Terminal or Linux shell prompt." To the right, there's a note: "The script explains what it will do and then pauses before it does it. Read about other [installation options](#)."

At the bottom, there are two links: "If you're on macOS, try our new .pkg installer." and "Download it from [Homebrew's latest GitHub release](#)".



Step2.請在終端機視窗安裝Homebrew:

- 貼上剛剛在Homebrew官網上複製的那一行並執行：

```
/bin/bash -c "$(curl -fsSL
```

```
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

The screenshot shows a terminal window titled 'Terminal' with the command `/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"` highlighted in yellow. The terminal output shows the Homebrew installation script running, including steps like creating directories, changing permissions, and downloading the Homebrew package. It ends with a message indicating the HEAD is now at a specific commit.

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"  
=> This script will install:  
/usr/local/bin/brew  
/usr/local/share/doc/homebrew  
/usr/local/share/man/man1/brew.1  
/usr/local/share/zsh/site-functions/_brew  
/usr/local/etc/bash_completion.d/brew  
/usr/local/Homebrew  
  
Press RETURN to continue or any other key to abort  
=> /usr/bin/sudo /bin/rmdir -p /Library/Caches/Homebrew  
Password:  
=> /usr/bin/sudo /bin/chmod g+rwx /Library/Caches/Homebrew  
=> /usr/bin/sudo /usr/sbin/chown eddie /Library/Caches/Homebrew  
=> Downloading and installing Homebrew...  
remote: Counting objects: 71, done.  
remote: Compressing objects: 100% (35/35), done.  
remote: Total 71 (delta 68), reused 42 (delta 34), pack-reused 0  
Unpacking objects: 100% (71/71), done.  
From https://github.com/Homebrew/brew  
+ 60d9218a...81877768 master --> origin/master (forced update)  
HEAD is now at 81877768 Merge pull request #3112 from mistydemeo/search_online_failure
```

1-8 Git安裝-macOS



Step3.請在終端機視窗呼叫Homebrew 安裝 Git，並確認Git 是否已安裝起來，以及核對版本資訊

The screenshot shows a terminal window titled 'iTerm2' with a dark theme. It displays the following command and its output:

```
iTerm2 Shell Edit View Profiles Toolkit Window Help
2. edition

高見龍: ~
brew install git
Warning: git 2.14.1 is already installed

高見龍: ~
which git
/usr/local/bin/git

高見龍: ~
git --version
git version 2.14.1

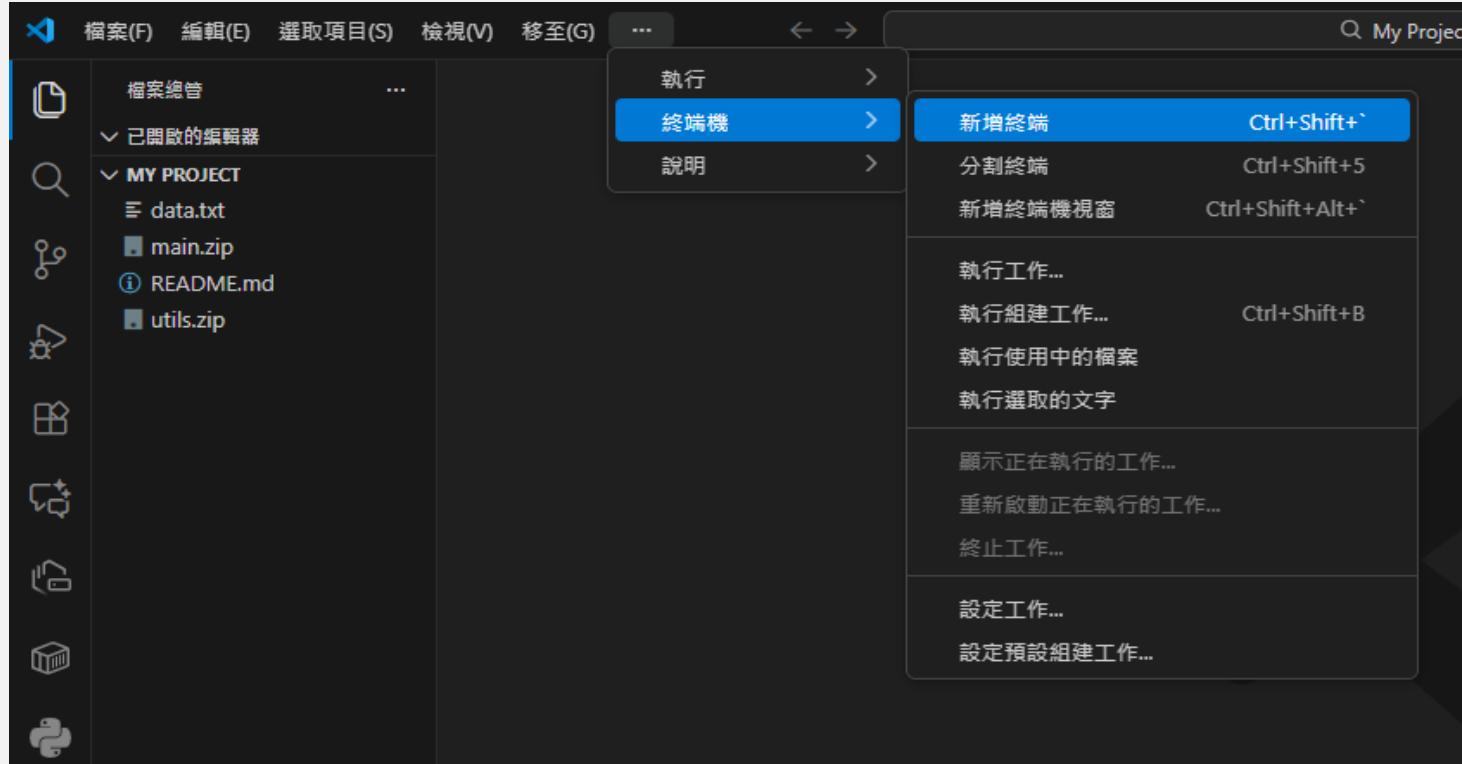
高見龍: ~
```

The command 'brew install git' is highlighted with a red box. The output 'git 2.14.1 is already installed' and the command 'git --version' are also highlighted with red boxes.

1-9 Git基本設定與初始化



Step1. 在VSCode新增終端





Step2.在VSCode終端機設定UserName&Email:

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

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

Step3.在VSCode終端機初始化檔案狀態:

```
git init
```

U:Untracked(未追蹤)

1-9 Git基本設定與初始化



```
PS D:\網路資料庫程式設計(大學部助教課程)碩一上\第十四週Git&GitHub教材(帶大三學弟妹上課)\My Project> git config --global user.name "Perry"
PS D:\網路資料庫程式設計(大學部助教課程)碩一上\第十四週Git&GitHub教材(帶大三學弟妹上課)\My Project> git config --global user.email "perry@perry.com"
PS D:\網路資料庫程式設計(大學部助教課程)碩一上\第十四週Git&GitHub教材(帶大三學弟妹上課)\My Project> git init
Reinitialized existing Git repository in D:/網路資料庫程式設計(大學部助教課程)碩一上/第十四週Git&GitHub教材(帶大三學弟妹上課)/My Project/.git/
```



(1) Git 的三個工作區域

區域	說明
Working Directory (工作區)	你正在編輯的檔案實體
Staging Area (暫存區)	你「選擇」要放進下一次 commit 的變化
Repository (版本庫)	你真正提交後永久保存的版本紀錄

(2) Git 追蹤的是「檔案變化」，不是檔案本身

- 包括新增、修改、刪除
- 刪除檔案也會變成一種「變化」，需要 commit 記錄

2-2 Git 常用指令與操作流程



(1) 查看檔案狀態指令: `git status`

狀態	意義
Untracked (未追蹤)	新增檔案, Git 尚未開始追蹤
Modified (已修改)	修改後尚未加入暫存
Staged (已暫存)	已準備加入下一次 commit
Committed (已提交)	已寫入版本歷史

main.py	M
README.md	A
test.txt	U
utils.py	

檔案總管目錄狀態

```
PS C:\Users\ASUS\OneDrive\桌面\My Project> git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    README.md
    data.txt
    main.py
    utils.py
```

未追蹤狀態

```
PS C:\Users\ASUS\OneDrive\桌面\My Project> git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   README.md
    new file:   data.txt
    new file:   main.py
    new file:   utils.py
```

已暫存狀態



(2) 暫存變化指令:

指令類型	說明
git add 檔案名稱	
git add .	(新增/修改全部加入暫存)
git add *.py	(萬用字元+檔案類型)

重點：

- 只有「加入 Staging」（暫存）的變化才會進入 commit
- 若你修改檔案，需要重新 add 才會記錄新的變化

```
Untracked files:  
(use "git add <file>  
 README.md  
 data.txt  
 main.py  
 utils.py
```



git add .

```
Changes to be committed:  
(use "git rm --cached <file>  
 new file: README.md  
 new file: data.txt  
 new file: main.py  
 new file: utils.py
```

VS Code 終端機介面狀態



(3) 提交版本指令: git commit -m "提交訊息"

重點:

- commit = 把暫存區的變化拍照寫進版本庫
- 提交訊息建議清楚描述更動內容 (如「修正登入錯誤」)

```
● PS C:\Users\ASUS\OneDrive\桌面\My Project> git commit -m "git教學範例"
[main (root-commit) 3eccb05] git教學範例
 4 files changed, 22 insertions(+)
  create mode 100644 README.md
  create mode 100644 data.txt
  create mode 100644 main.py
  create mode 100644 utils.py
● PS C:\Users\ASUS\OneDrive\桌面\My Project> git status
On branch main
nothing to commit, working tree clean
```

終端機介面狀態

2-2 Git 常用指令與操作流程



(4) 查看歷史紀錄指令: `git log` / `git log --oneline`

- 重點:
- 每一筆 commit 都有一個唯一的 commit ID(黃色雜湊數)
 - `--oneline` 版本簡潔, 方便快速找到版本

```
PS C:\Users\ASUS\OneDrive\桌面\My Project> git log
commit 6854d4083b485cb61c2883e08dc15e6ea2f65871 (HEAD -> main)
Author: joyce <sia0123789@gmail.com>
Date:   Sat Nov 15 14:08:11 2025 +0800

    修改utils中的add函式、新增sub函式、修改data

commit 3f942f20a824d2f769df1ade9176cfed6e50338e
Author: joyce <sia0123789@gmail.com>
Date:   Sat Nov 15 06:31:33 2025 +0800

    main更改成hello world

commit 3eccb052c24276048c07e6742b4981630ca4fc99
Author: joyce <sia0123789@gmail.com>
Date:   Sat Nov 15 06:17:30 2025 +0800

    git教學範例
```

```
PS C:\Users\ASUS\OneDrive\桌面\My Project> git log --oneline
6854d40 (HEAD -> main) 修改utils中的add函式、新增sub函式、修改data
3f942f2 main更改成hello world
3eccb05 git教學範例
```



(5) 比較新舊版本差異指令: `git diff/ git diff <commitID> --檔案名稱`

➤ 重點: ●綠色:新增 ●紅色:刪除 ●用來找 bug、確認變更、了解差異

```
PS C:\Users\ASUS\OneDrive\桌面\My Project> git diff  
diff --git a/data.txt b/data.txt  
index 4a8aef3..0306029 100644  
--- a/data.txt  
+++ b/data.txt  
@@ -1,3 +1,4 @@  
 sample data  
-12345  
+1234  
+5678  
  
diff --git a/utils.py b/utils.py  
index 4693ad3..411cbcb 100644  
--- a/utils.py  
+++ b/utils.py  
@@ -1,2 +1,4 @@  
-def add(a, b):  
+def add(a, b, c):  
+    return a + b + c  
+def sub(a, b):  
    return a - b  
+def mul(a, b):  
    return a * b
```

看全部尚未
暫存的修改

```
PS C:\Users\ASUS\OneDrive\桌面\My Project> git diff 3f942f2 -- utils.py  
diff --git a/utils.py b/utils.py  
index 4693ad3..6388db6 100644  
--- a/utils.py  
+++ b/utils.py  
@@ -1,2 +1,6 @@  
-def add(a, b):  
+def add(a, b, c):  
+    return a + b + c  
+def sub(a, b):  
    return a - b  
+def mul(a, b):  
    return a * b
```

單個檔案-舊版本 vs 新版本



(6) 還原檔案到過去版本指令: `git checkout <commitID> -- 檔案名稱`

重點: •把某個檔案還原到某個舊版本的狀態

•之後需要再 commit 才會正式記錄此還原

```
PS C:\Users\ASUS\OneDrive\桌面\My Project> git checkout 3f942f2 -- utils.py
PS C:\Users\ASUS\OneDrive\桌面\My Project> git commit -m "將utils改回最初狀態"
[main aee6be3] 將utils改回最初狀態
 1 file changed, 1 insertion(+), 5 deletions(-)

PS C:\Users\ASUS\OneDrive\桌面\My Project> git log --oneline
aee6be3 (HEAD -> main) 將utils改回最初狀態
5cda0ac 在utils中新增乘法函式
6854d40 修改utils中的add函式、新增sub函式、修改data
3f942f2 main更改成hello world
3eccb05 git教學範例
```

`git commit -m "將utils改回最初狀態"`

```
diff --git a/utils.py b/utils.py
index 6388db6..4693ad3 100644
--- a/utils.py
+++ b/utils.py 回到先前狀態
@@ -1,6 +1,2 @@
-def add(a, b, c):
-    return a + b +c
-def sub(a, b):
+def add(a, b):
    return a + b
-def mul(a, b):
-    return a * b
```



(7) 還原檔案到指定版本指令: `git reset --hard <commitID>`

重點:

- 這會將整個專案「完整回到」指定版本，後面所有 commit 都會消失。
- 不可逆，需謹慎使用。**



```
PS C:\Users\ASUS\OneDrive\桌面\My Project> git log --oneline
80d40fc (HEAD -> main) 新增.gitignore檔案
aee6be3 將utils改回最初狀態
5cda0ac 在utils中新增乘法函式
6854d40 修改utils中的add函式、新增sub函式、修改data
3f942f2 main更改成hello world
3eccb05 git教學範例
```

現在歷史狀態

- PS C:\Users\ASUS\OneDrive\桌面\My Project> `git reset --hard 3f942f2`
- PS C:\Users\ASUS\OneDrive\桌面\My Project> `git log --oneline`
3f942f2 (HEAD -> main) main更改成hello world
3eccb05 git教學範例



(8) 忽略不必要追蹤的檔案: `.gitignore`

`.gitignore` 檔案用途:

- 避免追蹤不必要的檔案，讓版本庫保持乾淨
- 常用於：圖片、log、暫存檔、編譯產出、環境設定檔

The screenshot shows the VS Code interface. On the left, the 'File Explorer' sidebar displays a project folder named 'MY PR...' containing files: '.gitignore', '輸出結果1.png', '輸出結果2.png', 'data.txt', 'main.py', 'README.md', and 'utils.py'. The '.gitignore' file is selected and highlighted with a red border. On the right, the 'Terminal' window shows the command-line interface with the following text:
PS C:\Users\ASUS\OneDrive\桌面\My Project> git add .
PS C:\Users\ASUS\OneDrive\桌面\My Project> git commit -m "新增.gitignore檔案"
[main 80d40fc] 新增.gitignore檔案
1 file changed, 1 insertion(+)
create mode 100644 .gitignore

VS Code 終端機介面狀態

2-3 小結：Git 單人開發的基本工作流



功能	指令
看狀態	<code>git status</code>
暫存變化	<code>git add</code> 檔案、 <code>git add ..</code> 、 <code>git add *.py</code>
建立版本	<code>git commit -m "訊息"</code>
看歷史	<code>git log</code> 、 <code>git log --oneline</code>
比較差異	<code>git diff</code> 、 <code>git diff <ID> -- 檔案</code>
檔案還原	<code>git checkout <ID> -- 檔案</code>
專案回溯	<code>git reset --hard <ID></code> (危險)
忽略檔案	<code>.gitignore</code>

3 同步儲存庫



(1) 新建儲存庫

The screenshot shows the GitHub Dashboard. On the left, there's a sidebar with 'Top repositories' and a search bar. In the center, a modal window titled 'Join GitHub Education!' is open, advertising free services for teachers and students. On the right, a dropdown menu is open with various options like 'New issue', 'New repository' (which is highlighted with a red box), and 'Import repository'. A red number '1.' is above the '+' icon in the dropdown, and a red number '2.' is above the 'New repository' option.

1.

2.

Dashboard

New

Find a repository...

SuYLilian/Wing_Forest
SuYLilian/Game_Gi-Gi-Gi
SuYLilian/Game_Archer
SuYLilian/Game_Dicetopia
SuYLilian/Game_Oven_Bombing
SuYLilian/Plant
SuYLilian/Game_Winner-In-Life

Show more

Join GitHub Education!

GitHub Education opens doors to new skills, tools, and a collaborative community eager to drive innovation. Join us and build a foundation for your future in technology.

Simple cloud hosting, built for developers

Copilot Turn natural language prompts into coding suggestions

Heroku Build, run, and operate applications entirely in the cloud.

Microsoft Azure Access to Microsoft Azure cloud services and learning resources

Join GitHub Education

Home Try the new experience

Ask anything

+ New issue

New repository

Import repository

Latest from GitHub

- 4 days ago Create a new repository as a guest
- 4 days ago Open source code
- 4 days ago New organization
- 4 days ago New project
- 4 days ago New GitHub Actions CI/CD token claims
- 4 days ago Manage Copilot coding agent tasks in Visual Studio Code

View changelog →

3 同步儲存庫



(2) 儲存庫基本設定

Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (*).

1 General

Owner * SuYLilian / Repository name * GitHub_Git_Tutorial GitHub_Git_Tutorial is available.

Great repository names are short and memorable. How about [reimagined-octo-guide?](#)

Description This is the teaching material used in class. 44 / 350 characters

2 Configuration

Choose visibility * Choose who can see and commit to this repository Public

Add README READMEs can be used as longer descriptions. [About READMEs](#) Off

Add .gitignore .gitignore tells git which files not to track. [About ignoring files](#) No .gitignore

Add license Licenses explain how others can use your code. [About licenses](#) No license

儲存庫名稱

儲存庫簡介

選擇可見性

加入說明文件

讓 Git 知道哪些檔案不應該被加入版本控制
(GitHub提供各語言/框架的範本)

加入授權條款

Public: 任何人都可以看到你的儲存庫
Private: 只有你與你授權的人才能看到

3 同步儲存庫



如果沒有加入 README、.gitignore 或 license，會看到下面有提供現成的指令，
協助我們把本地的檔案推送到 GitHub

The screenshot shows a GitHub repository page for 'GitHub_Git_Tutorial'. The 'Code' tab is selected. At the top, there are sections for 'Set up GitHub Copilot' and 'Add collaborators to this repository'. Below these, a 'Quick setup' section provides instructions for pushing an existing repository from the command line. A red box highlights the command-line instructions for pushing an existing repository. The commands listed are:

```
git remote add origin https://github.com/SuYLilian/GitHub_Git_Tutorial.git  
git branch -M main  
git push -u origin main
```

3 同步儲存庫



(3) git remote add origin <儲存庫網址>

- 連接本地與遠端的儲存庫

The screenshot shows a GitHub repository page for 'GitHub_Git_Tutorial'. The 'Code' tab is selected. A red box highlights the 'Quick setup' section, which contains instructions for setting up a local repository or adding it to GitHub via HTTPS or SSH. Below this, another red box highlights a command-line instruction for adding a remote repository:

```
git remote add origin https://github.com/SuYLilian/GitHub_Git_Tutorial.git
```

Red annotations are present: '儲存庫網址(HTTPS)' points to the HTTPS URL in the quick setup box, and '可直接複製這行指令' points to the command-line instruction in the bottom box.

3 同步儲存庫



(4) git branch -M main

- 把目前的主線分支強制改名成 main，通常用來跟 GitHub 的預設分支名稱保持一致

The screenshot shows a GitHub repository page for 'GitHub_Git_Tutorial'. The 'Code' tab is selected. In the center, there's a 'Quick setup' section with instructions for creating a new repository or pushing an existing one from the command line. A red box highlights the command 'git branch -M main'.

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH https://github.com/SuYLilian/GitHub_Git_Tutorial.git

Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

...or create a new repository on the command line

```
echo "# GitHub_Git_Tutorial" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/SuYLilian/GitHub_Git_Tutorial.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/SuYLilian/GitHub_Git_Tutorial.git
git branch -M main
git push -u origin main
```

可直接複製這行指令

3 同步儲存庫



(5) **git push -u origin main**

- 將本地檔案推送到GitHub

The screenshot shows a GitHub repository page for 'GitHub_Git_Tutorial'. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The repository name 'GitHub_Git_Tutorial' is displayed, along with its status as 'Public'. Below the repository name are sections for 'Set up GitHub Copilot' and 'Add collaborators to this repository'. A large central box contains 'Quick setup — if you've done this kind of thing before' with instructions for setting up in Desktop or via HTTPS/SSH, and for creating a new repository or pushing an existing one from the command line. The command line section includes the following text:
...or push an existing repository from the command line
git remote add origin https://github.com/SuYLilian/GitHub_Git_Tutorial.git
git branch -M main
git push -u origin main

A red box highlights the final command 'git push -u origin main', with the text '可直接複製這行指令' (You can directly copy this line of command) overlaid in red.

3 同步儲存庫



步驟	指令類型	說明
1	git remote add origin <儲存庫網址>	連接本地與遠端的儲存庫
2	git branch -M main	將主線分支改成main
3	git push -u origin main	推送到 GitHub

```
PS C:\Users\User\Desktop\My Project> git remote add origin https://github.com/SuYLilian/GitHub_Git_Tutorial.git
PS C:\Users\User\Desktop\My Project> git branch -M main
PS C:\Users\User\Desktop\My Project> git push -u origin main
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 16 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (12/12), 1.11 KiB | 1.11 MiB/s, done.
Total 12 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/suYLilian/GitHub_Git_Tutorial.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\User\Desktop\My Project>
```

3 同步儲存庫



- 可以到GitHub看到從本地推送上去的檔案

The screenshot shows a GitHub repository page for 'GitHub_Git_Tutorial'. The repository is public and has 1 branch and 0 tags. The main file listed is '.gitignore' with a commit message '新增.gitignore檔案'. Other files listed include 'README.md', 'data.txt', 'main.py', and 'utils.py', all with commit messages related to git教學範例. The 'About' section notes that this is the teaching material used in class. The 'Files' section shows the contents of 'README.md' which reads: 'Git Demo Project. This is a simple project used to demonstrate basic Git commands in VSCode.' The 'Languages' section indicates 100% Python code.

SuYLilian / GitHub_Git_Tutorial

Type ⌘ to search

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

GitHub_Git_Tutorial Public

Pin Watch 0 Fork 0 Star 0

main 1 Branch 0 Tags Go to file Add file Code

SuYLilian 新增.gitignore檔案 d6a37b7 · 2 hours ago 3 Commits

.gitignore 新增.gitignore檔案 2 hours ago

README.md git教學範例 2 hours ago

data.txt git教學範例 2 hours ago

main.py main更改成hello world 2 hours ago

utils.py git教學範例 2 hours ago

README

Git Demo Project

This is a simple project used to demonstrate basic Git commands in VSCode.

Files

About

This is the teaching material used in class.

Readme

Activity

0 stars

0 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages

Python 100.0%

3 同步儲存庫



- 若儲存庫在一開始新建時不是空的 (有 README、License 或 .gitignore)，會出現以下錯誤
- 原因：GitHub 有某個檔案，但本地端沒有這個檔案，所以不讓你覆蓋

```
PS C:\Users\User\Desktop\MyProject_2> git remote add origin https://github.com/SuYLilian/RepoNotEmpty.git
PS C:\Users\User\Desktop\MyProject_2> git branch -M main
PS C:\Users\User\Desktop\MyProject_2> git push -u origin main
To https://github.com/SuYLilian/RepoNotEmpty.git
 ! [rejected]      main -> main (fetch first)
error: failed to push some refs to 'https://github.com/SuYLilian/RepoNotEmpty.git'
hint: Updates were rejected because the remote contains work that you do not
hint: have locally. This is usually caused by another repository pushing to
hint: the same ref. If you want to integrate the remote changes, use
hint: 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
PS C:\Users\User\Desktop\MyProject_2>
```

3 同步儲存庫



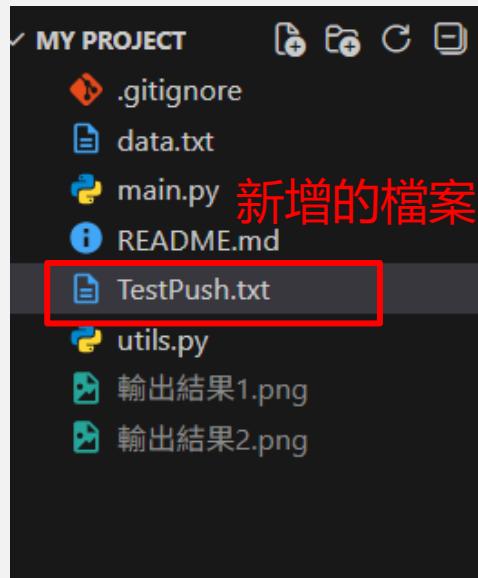
- 解決方式：在推送檔案到儲存庫之前，先把儲存庫原有的內容拉下來
- git pull origin main --allow-unrelated-histories

步驟	指令類型	說明
1	git remote add origin <儲存庫網址>	連接本地與遠端的儲存庫
2	git branch -M main	將主線分支改成main
3	git pull origin main --allow-unrelated-histories	把儲存庫原有的內容拉下來
4	git push -u origin main	推送到 GitHub

3 同步儲存庫



- 之後若有更改檔案內容 (修改內容、新增檔案、刪除檔案)，一樣要將檔案加入暫存 -> commit -> 推送到 GitHub



```
git add .  
git commit -m "提交訊息"  
git push
```

A screenshot of a GitHub repository named 'tutorial' (Public). The commit history shows:

Commit	Message	Time
c374c48	新增.gitignore檔案	3 hours ago
c374c48	git教學範例	3 hours ago
c374c48	推送測試	1 minute ago
c374c48	main變成hello world	3 hours ago
c374c48	git教學範例	3 hours ago

A red box highlights the commit '推送測試' (push test) made 1 minute ago.

4 協作



- 輸入對方的名稱或email，可以加入其他人協作專案

The screenshot shows the GitHub repository settings page for 'SuYLilian / GitHub_Git_Tutorial'. The 'Settings' tab is selected (1). On the left sidebar, the 'Collaborators' option under the 'Access' section is highlighted with a red box (2). The main content area displays the 'Collaborators and teams' section. It shows that the repository is a 'Public repository' and that there are currently 0 collaborators. A large callout box highlights the 'Manage access' section with the text 'You haven't invited any collaborators yet' and a red box around the 'Add people' button (3).

4 協作



● 被邀請者會收到訊息

The screenshot shows the GitHub Notifications inbox. At the top right, there is a toolbar with several icons, one of which is highlighted with a red box and the number '1'. Below the toolbar, a modal window titled 'Clear out the clutter.' provides instructions on marking notifications as done. The main inbox area displays a single notification from 'SuYLilian/GitHub_Git_Tutorial' with the subject 'Invitation to join SuYLilian/GitHub_Git_Tutorial from SuYLilian'. This notification is also highlighted with a red box and the number '2'. On the left side, there is a sidebar with various filters like 'Assigned', 'Participating', and 'Mentioned'. At the bottom, there are sections for 'Repositories' and 'Manage notifications'.

Notifications

Inbox 2 All Unread Search notifications

Saved

Done

Filters

Assigned

Participating

Mentioned

Team mentioned

Review requested

+ Add new filter

Repositories

SuYLilian/GitHub_Git_Tutorial 1

Manage notifications

Clear out the clutter.
Get the most out of your new inbox by quickly and easily marking all of your previously read notifications as done.

Dismiss Get started

1. Group by: Date

2. Select all

SuYLilian/GitHub_Git_Tutorial Invitation to join SuYLilian/GitHub_Git_Tutorial from SuYLilian

subscribed now

ProTip! When viewing a notification, press e to mark it as Done.

1-1 of 2 Prev Next

4 協作

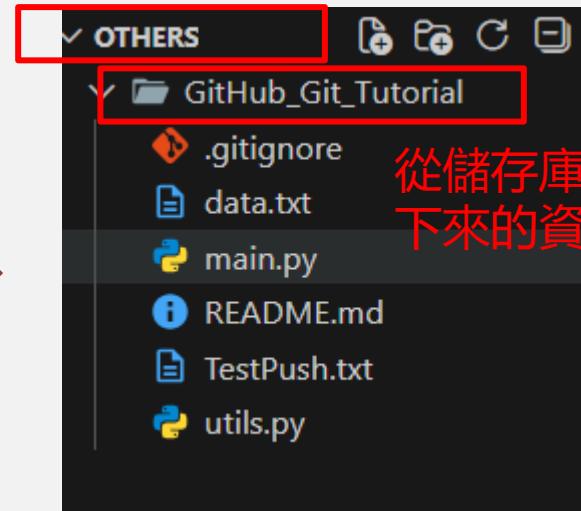


- 被邀請者需新建一個資料夾，將儲存庫的內容複製到本地端
- `git clone <儲存庫網址>`

The screenshot shows a GitHub repository named 'GitHub_Git_Tutorial'. The 'Code' dropdown menu is open, with the 'HTTPS' option highlighted. A red box highlights the 'Code' button and the 'Clone' button. Another red box highlights the 'HTTPS' link.

```
PS C:\Users\User\Desktop\Others> git clone https://github.com/SuYLilian/GitHub_Git_Tutorial.git
Cloning into 'GitHub_Git_Tutorial'.
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 15 (delta 4), reused 15 (delta 4), pack-reused 0 (from 0)
Receiving objects: 100% (15/15), done.
Resolving deltas: 100% (4/4), done.
PS C:\Users\User\Desktop\Others>
```

新建的資料夾

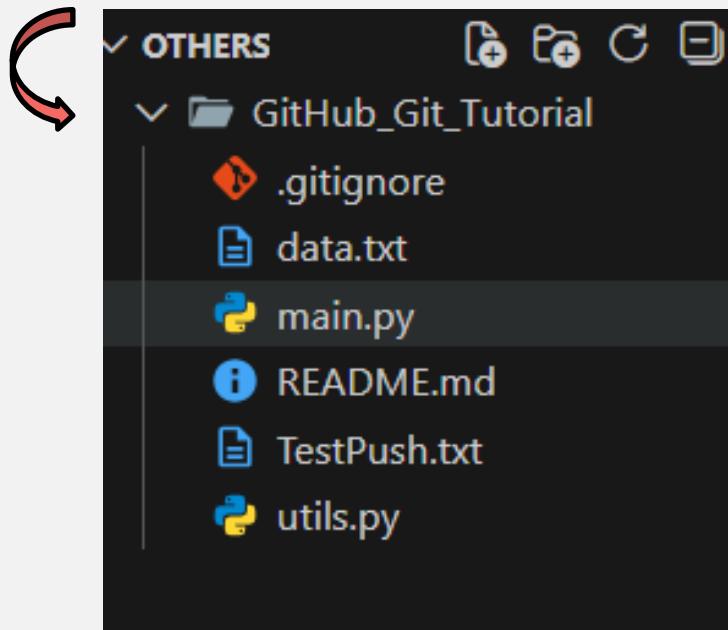


從儲存庫複製
下來的資料夾

4 協作



- 若被邀請者要編輯儲存庫，需要將程式執行路徑更改至儲存庫的資料夾
- cd 資料夾名稱



```
PS C:\Users\User\Desktop\Others> cd GitHub_Git_Tutorial  
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial>
```



維持同步

- 如果其他人有推送修改後的檔案至GitHub，我們也要更新自己的本地端，
把新紀錄的檔案拉下來
- `git pull`



為什麼要建立分支 (Branch) ?

- 避免影響主線 (main) 程式
→ 在分支上開發新功能，不會破壞現有版本。
- 多人協作更安全
→ 每個人都有自己的分支，避免互相干擾。
- 方便測試與實驗
→ 想嘗試新想法？先在分支測試，失敗也不會影響主專案。
- 清楚管理不同功能與版本
→ 每個功能一條分支，最後再合併。

5 分支與合併



指令類型	說明
git switch -c 分支名稱	建立新分支並切換過去
git branch 分支名稱	建立新分支
git switch 分支名稱	切換分支
git branch	查看分支清單/目前所在分支

*代表目前所在分支

```
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git switch -c branch2
Switched to a new branch 'branch2'
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git branch
* branch2
  main

PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git branch branch3
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git switch branch3
Switched to branch 'branch3'
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git branch
  branch3
* branch3
  main

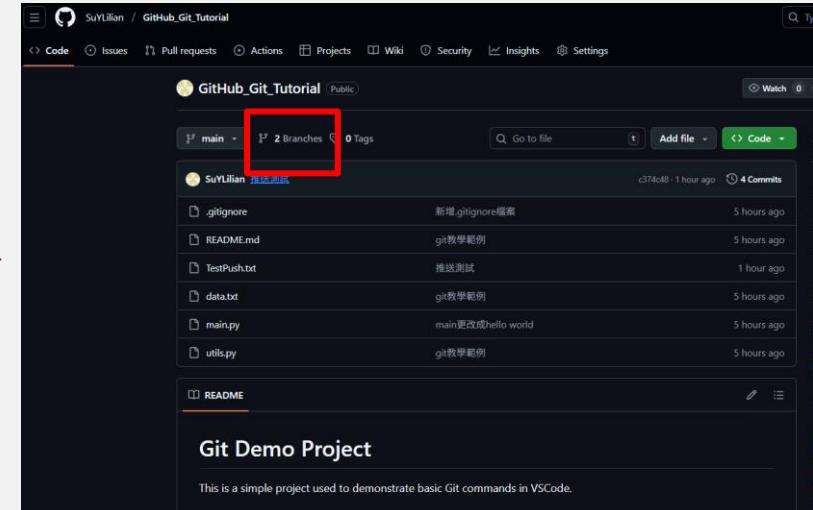
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial>
```

5 分支與合併



- 將當前分支所變更的紀錄推送到GitHub
- git push origin 分支名稱

```
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git switch branch2
Switched to branch 'branch2'
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git add .
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git commit -m "utils新增乘法"
[branch2 e3171a7] utils新增乘法
 1 file changed, 3 insertions(+)
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git push origin branch2
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 311 bytes | 311.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'branch2' on GitHub by visiting:
remote:     https://github.com/SuYLilian/GitHub_Git_Tutorial/pull/new/branch2
remote:
To https://github.com/SuYLilian/GitHub_Git_Tutorial.git
 * [new branch]      branch2 -> branch2
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial>
```



5 分支與合併



- 我們可以請求管理者將分支合併至主線
- Pull request

The screenshot shows the GitHub 'Branches' page. At the top, there are navigation links: Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the navigation is a search bar labeled 'Search branches...'. The main area is divided into 'Default' and 'Active branches' sections. In the 'Default' section, there is a branch named 'main' with an update timestamp of '1 hour ago'. In the 'Active branches' section, there is a branch named 'branch2' with an update timestamp of '16 minutes ago'. On the right side of the page, there is a sidebar with a 'New branch' button. A red box highlights the 'New pull request' button in the sidebar. A red arrow points from the 'Create pull request' button in the bottom-left modal to the 'New pull request' button in the sidebar.

The screenshot shows the 'Comparing changes' modal for creating a pull request. The modal has a title 'Comparing changes' and instructions: 'Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#)'. It shows the 'base: main' and 'compare: branch2' dropdowns, and a note: 'Able to merge. These branches can be automatically merged.' Below this, there are fields for 'Add a title' (containing 'util新增乘法') and 'Add a description' (containing '我新增乘法函數，我請求併到主線'). The modal also includes sections for 'Reviewers', 'Assignees', 'Labels', 'Projects', 'Milestone', and 'Development'. At the bottom, there is a 'Create pull request' button. A red arrow points from this button to the 'New pull request' button on the GitHub page.

5 分支與合併



● 管理者可以看到請求

The screenshot shows the GitHub interface for pull requests. At the top, there are navigation links: Issues, Pull requests (1), Actions, Projects, Wiki, Security, Insights, and Settings. Below the navigation bar, a modal window titled "Label issues and pull requests for new contributors" is displayed, stating: "Now, GitHub will help potential first-time contributors discover issues labeled with good first issue". A "Dismiss" button is located in the top right corner of the modal.

Below the modal, the main pull request list is shown. It includes filters: "Filters" (dropdown), "is:pr is:open" (search bar), "Labels 9", "Milestones 0", and a green "New pull request" button. The list shows one open pull request:

- 1 Open ✓ 0 Closed
- 🌟 utils新增乘法 #1 opened 3 minutes ago by SuSuYL

A red box highlights the second item in the list, which is the pull request from SuSuYL. At the bottom of the page, there is a "ProTip!" message: "Ears burning? Get @SuYLilian mentions with [mentions:SuYLilian](#)".

5 分支與合併



SuSuYL wants to merge 1 commit into `main` from `branch2`

Conversation 0 **Commits 1** **Checks 0** **Files changed 1**

Reviewers
No reviews
Still in progress? Convert to draft

Assignees
No one assign yourself

Labels

No conflicts with base branch
Merging can be performed automatically.

Merge pull request You can also merge this with the command line. [View command line instructions](#)

Add a comment

SuSuYL wants to merge 1 commit into `main` from `branch2`

Conversation 0 **Commits 1** **Checks 0** **Files changed 1**

Changes from all commits **File filter** **Conversations** **Jump to**

utils新增乘法

branch2 (#1)

SuYLilian committed 34 minutes ago

`commit e3171a7e4b7689717223a97c12783f3c37c7d098`

`utils.py`

```
diff --git a/utils.py b/utils.py
--- a/utils.py
+++ b/utils.py
@@ -1,2 +1,5 @@
 1 1 def add(a, b):
 2+> 2+ return a + b
 3+
 4+def mul(a, b):
```

Write **Preview** **Cancel** **Add single comment** **Start a review**

Leave a comment

Markdown is supported | Paste, drop, or click to add files

可以在上面給評論

- 可以查看修改的內容，並在上面給評論

可以在上面給評論

5 分支與合併



● 或是請求進行修改

The screenshot shows a GitHub pull request interface. At the top, there are tabs for Conversation (0), Commits (1), Checks (0), and Files changed (1). Below these are filters for Changes from all commits, File filter, Conversations, Jump to, and settings.

The main area displays a commit titled "utils新增乘法" by user "branch2 (#1)". The commit message is "SuYLilian committed 38 minutes ago". The diff shows additions to the "utils.py" file:

```
diff --git a/utils.py b/utils.py
@@ -1,2 +1,5 @@
 1   1 def add(a, b):
 2   2     return a + b
 3 +
 4 +def mul(a, b):
```

Below the commit, there is a "Write" section with a "Leave a comment" input field. The "Markdown is supported" and "Paste, drop, or click to add files" options are available.

On the right, a "Finish your review" modal is open. It includes a toolbar with Write, Preview, H, B, I, etc. buttons. The "Review changes" button is highlighted with a red box and labeled "1.". The modal also contains sections for "Comment", "Approve", and "Request changes". The "Request changes" option is selected and highlighted with a red box and labeled "2.". A "Submit review" button is at the bottom right of the modal.

5 分支與合併



- 如果覺得沒問題就可以合併分支，並刪除分支

The screenshot shows two side-by-side GitHub pull request pages. The left page is for a pull request from branch2 to main, with a comment from SuSuYL: "我新增乘法函數，我請求併到主線". The right page shows the result after merging: "Pull request successfully merged and closed". A red box highlights the "Merge pull request" button on the left, labeled "1.". Another red box highlights the "Delete branch" button on the right, labeled "2.".

utils新增乘法 #1

1. Open SuSuYL wants to merge 1 commit into `main` from `branch2`

Conversation 0 Commits 1 Checks 0 Files changed 1

SuSuYL commented 20 minutes ago
我新增乘法函數，我請求併到主線

No conflicts with base branch
Merging can be performed automatically.

Merge pull request

1.

2.

SuSuYL commented 21 minutes ago
我新增乘法函數，我請求併到主線

SuYLilian merged commit `32f807e` into `main` now

Pull request successfully merged and closed

Add a comment

Write Preview

Markdown is supported Paste, drop, or click to add files

Comment



- GitHub上刪除分支後，本地端也需同步，需手動刪除

步驟	指令類型	說明
1	git switch main	切換至主線
2	git pull	下載GitHub上最新的變更
3	git branch -d 分支名稱	刪除分支
4	git fetch -p	清掉不存在的遠端分支記錄

5 分支與合併



```
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git switch main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git pull
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (1/1), 920 bytes | 306.00 KiB/s, done.
From https://github.com/SuYLilian/GitHub_Git_Tutorial
  c374c48..32f807e  main      -> origin/main
Updating c374c48..32f807e
Fast-forward
  utils.py | 3 ++
  1 file changed, 3 insertions(+)
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git branch -d branch2
Deleted branch branch2 (was e3171a7).
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git fetch -p
From https://github.com/SuYLilian/GitHub_Git_Tutorial
  - [deleted]          (none)    -> origin/branch2
PS C:\Users\User\Desktop\Others\GitHub_Git_Tutorial> git branch
  branch3
* main
```

指令整理



指令	說明
git status	看狀態
git add 檔案、git add .、git add *.py	暫存變化
git commit -m "訊息"	建立版本
git log、git log --oneline	看歷史
git diff、git diff <ID> -- 檔案	比較差異
git checkout <ID> -- 檔案	檔案還原
git reset --hard <ID> (危險)	專案回溯
git remote add origin <儲存庫網址>	連接本地與遠端的儲存庫
git branch -M main	將主線分支改成main
git push -u origin main	推送到 GitHub (第一次推送)
git pull origin main --allow-unrelated-histories	把儲存庫原有的內容拉下來
git push	推送更改記錄到 GitHub

指令	說明
git clone <儲存庫網址>	將儲存庫的內容複製到本地端
cd 資料夾名稱	切換到想操作的資料夾
git pull	將儲存庫的內容同步至本地端
git switch -c 分支名稱	建立新分支並切換過去
git branch 分支名稱	建立新分支
git switch 分支名稱	切換分支
git branch	查看分支清單/目前所在分支
git push origin 分支名稱	分支所變更的紀錄推送到GitHub
git branch -d 分支名稱	刪除分支
git fetch -p	清掉不存在的遠端分支記錄



什麼是Git 的圖形使用者介面 (GUI) ？

Git GUI 是用視覺化操作來管理版本控制的工具。使用者可以用滑鼠點選按鈕來完成 Git 的動作 (如 commit、push) , 不需要輸入指令。

常見的 Git GUI 工具 ↓



GitHub Desktop



Sourcetree

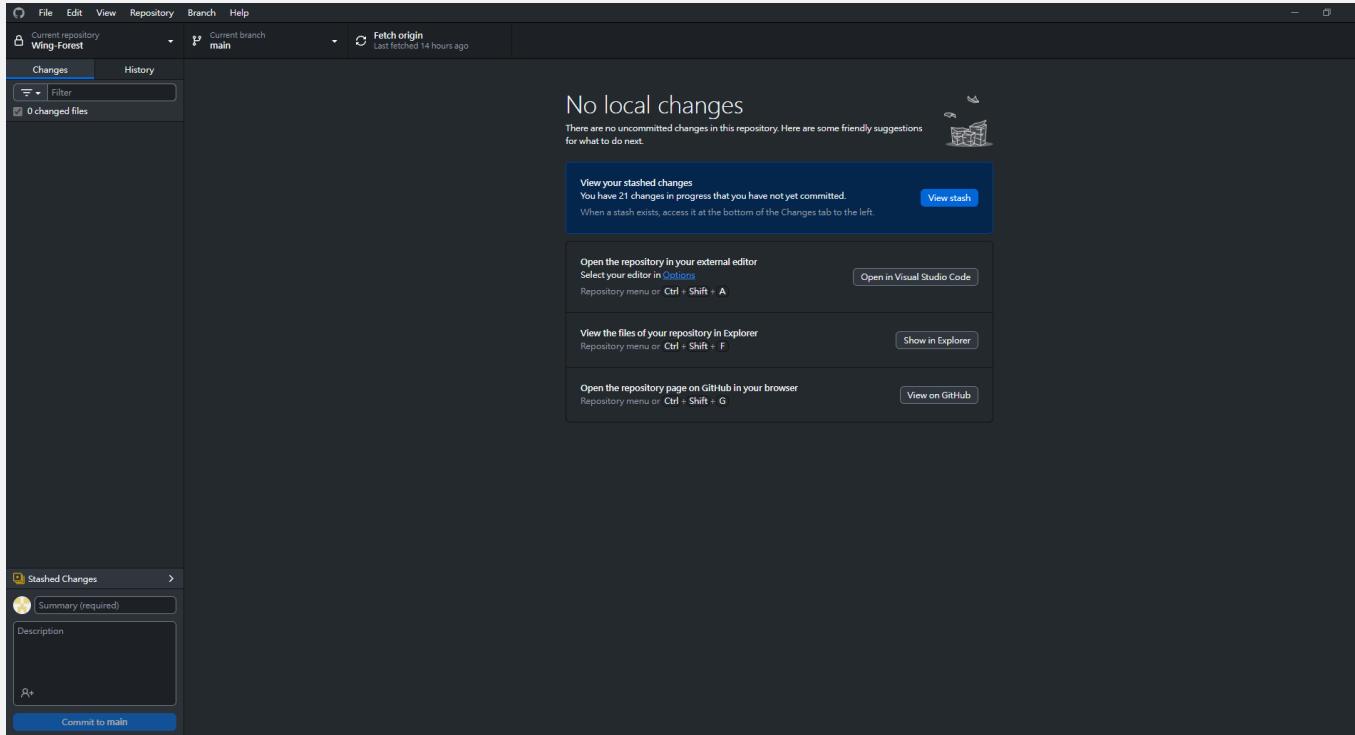


GitKraken

6 Git GUI (GitHub Desktop)



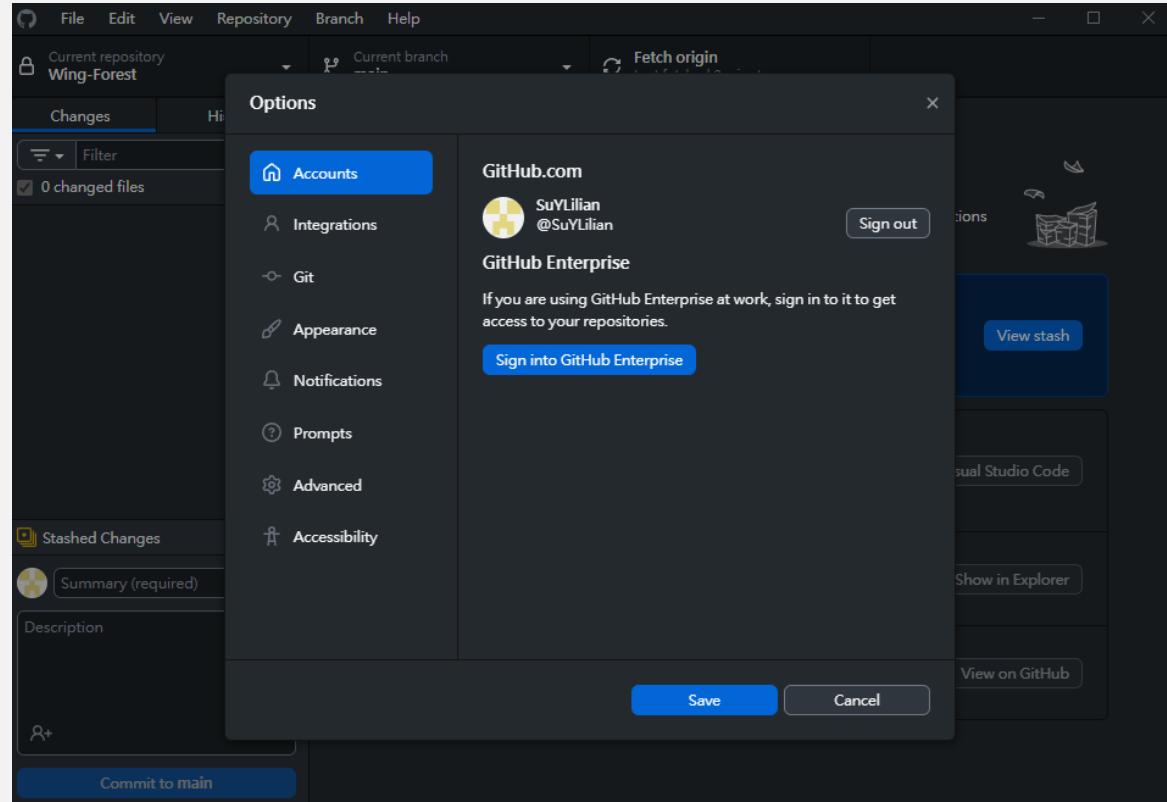
- GitHub Desktop
- 教學影片參考: <https://youtu.be/8Dd7KRpKeaE?si=y72qpKdGI8Yx1jAj>



6 Git GUI (GitHub Desktop)



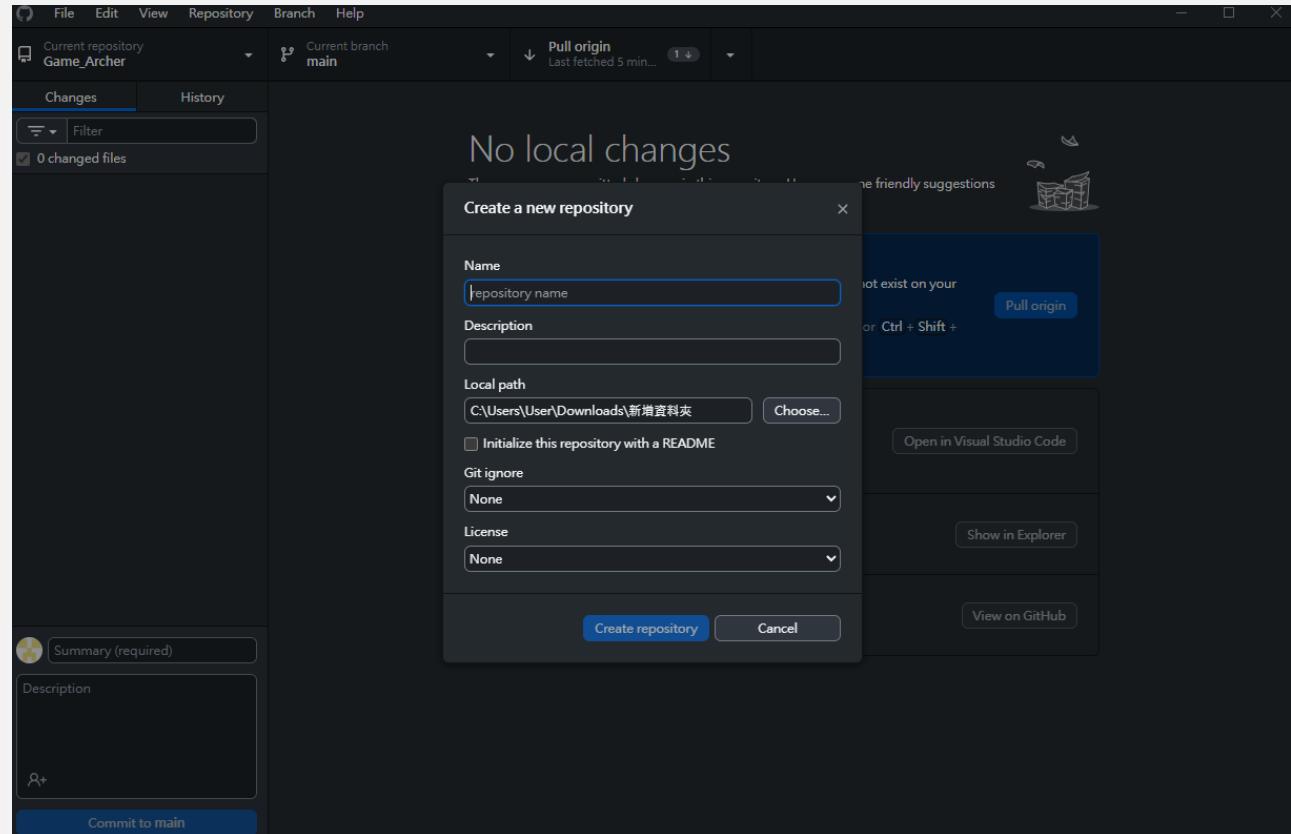
- 登入GitHub帳號，連結GitHub
- File → Options → Accounts



6 Git GUI (GitHub Desktop)



- 新建儲存庫
- File → New repository



6 Git GUI (GitHub Desktop)



● 介面介紹

The screenshot shows the GitHub Desktop application interface. At the top, there's a menu bar with File, Edit, View, Repository, Branch, and Help. Below the menu is a toolbar with icons for repository status, current branch (main), and publishing. A large central area displays a file viewer for 'main.py' with code for adding numbers. To the left, there's a sidebar with tabs for 'Changes' (highlighted with a red box) and 'History'. Under 'Changes', three files are listed: 'data.txt', 'main.py', and 'utils.py', all marked as changed. A red arrow points from the text '變更的檔案' to the 'Changes' tab. Another red arrow points from '歷史紀錄' to the 'History' tab. A third red arrow points from '切換/建立分支' to the 'Branch' tab in the toolbar. At the bottom, a 'Commit' dialog box is open, containing fields for a commit message ('FirstCommit') and a description, with a 'Commit 3 files to main' button. A red box surrounds this dialog. The overall background is dark-themed.

變更的檔案

歷史紀錄

切換/建立分支

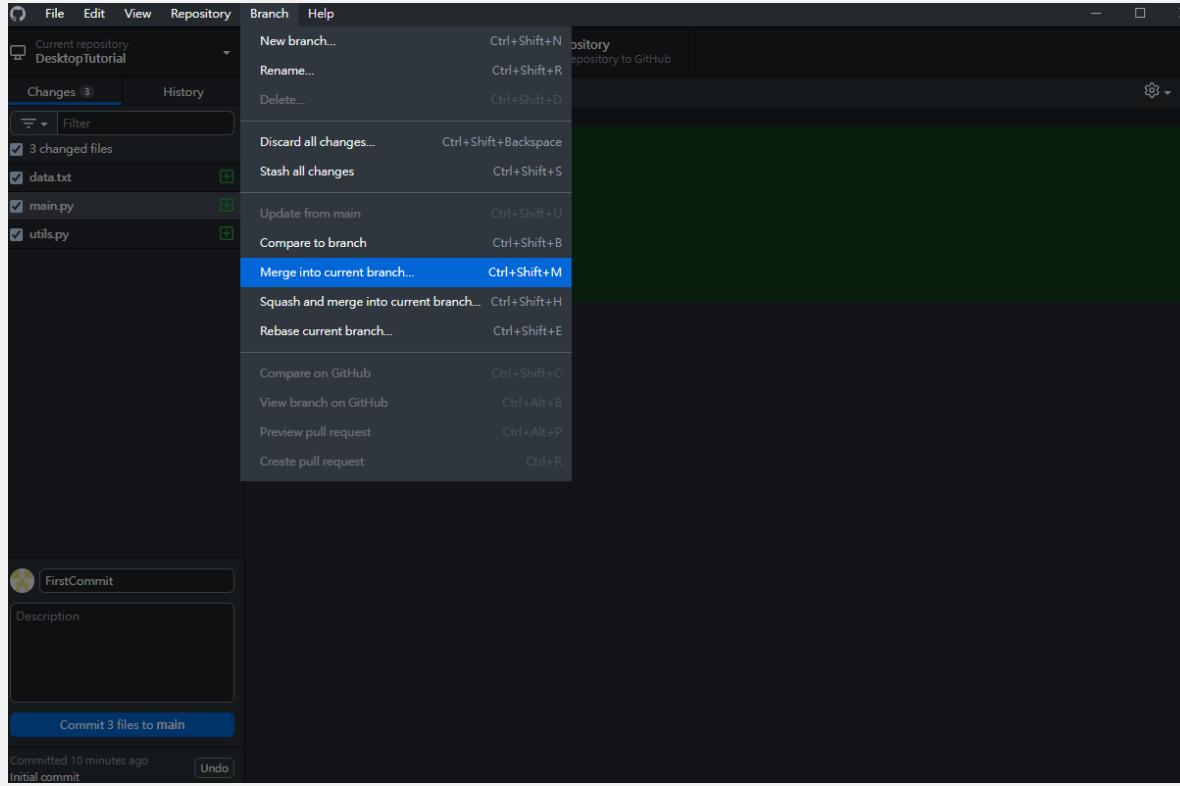
Commit

```
@@ -0,0 +1,9 @@
1 + from utils import add
2 +
3 + def main():
4 +     print("Hello Git!")
5 +     result = add(3, 5)
6 +     print("3 + 5 =", result)
7 +
8 + if __name__ == "__main__":
9 +     main()
```

6 Git GUI (GitHub Desktop)



● 合併分支



Thank you for listening

