

# Git 與 Github

TYIC 桃高資訊社

# Git

Git 是個版本控制系統(Version Control System，簡稱 VCS)

Git 是由 Linux 之父林納斯·托瓦茲(Linus Torvalds)開發  
開發的目的就是用來管理 Linux 的程式碼

版本控制系統，顧名思義，就是能控制版本  
每筆提交(commit)都會被紀錄下來，隨時可檢視和復原(revert)  
也可以建立分支(branch)，進行獨立修改  
並在必要時將分支合併(merge)

# Git

Git 的第一版 **README (讀我)** 中寫道：

 Git 的第一版 README

*GIT - the stupid content tracker*

*"git" can mean anything, depending on your mood.*

- random three-letter combination that is pronounceable, and not actually used by any common UNIX command.*

*The fact that it is a mispronunciation of "get" may or may not be relevant.*

- stupid. contemptible and despicable. simple. Take your pick from the dictionary of slang.*
- "global information tracker": you're in a good mood, and it actually works for you.*

*Angels sing, and a light suddenly fills the room.*

- "goddamn idiotic truckload of sh\*t": when it breaks*

*This is a stupid (but extremely fast) directory content manager.*

*It doesn't do a whole lot, but what it does do is track directory contents efficiently.*

*- Linus Torvalds*

# Github

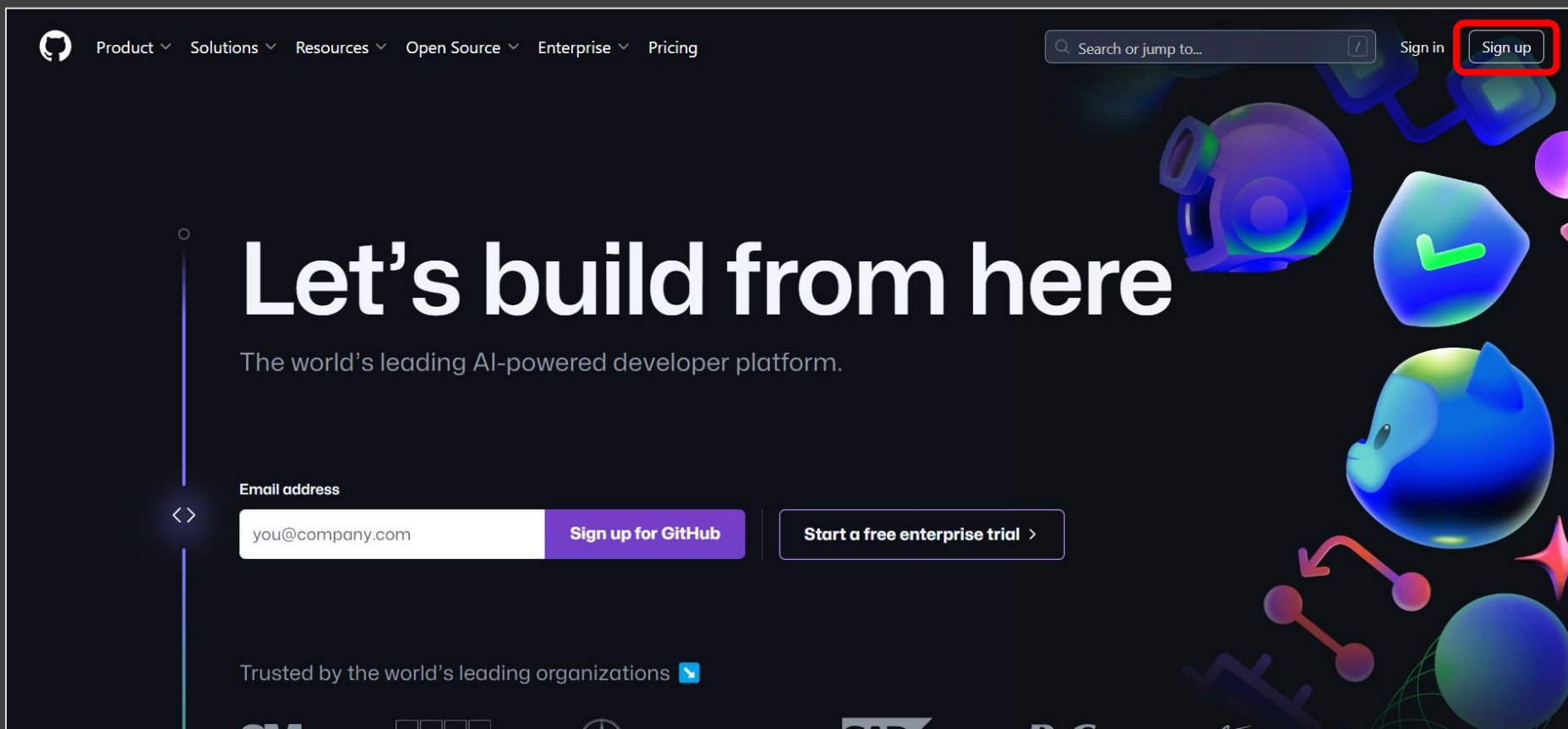
使用 **Git** 時，需要有地方可以存放檔案  
而當今最常見的就是 **Github**，網站為 <https://github.com>  
**Github** 除了提供數據庫(repository，簡稱 repo)供檔案存取  
還提供了問題(issue)、動作(action)、維基(wiki)  
分叉(fork)、拉取請求(pull request，簡稱 PR)等服務  
使用 **Github** 相比直接使用 **Git**  
更著重在一個網路上的第三者參與或貢獻(contribute)該數據庫  
而數據庫擁有者(owner)則可以很方便的管理他人貢獻

 <https://github.com/>

# Github 帳號註冊

Github 帳號註冊非常簡單

只要在 <https://github.com> 首頁右上方選擇 "Sign up"  
接著依照指示即可完成註冊



# 數據庫

點選右上方頭像即會出現右方導覽欄

從右方導覽欄點選 **"Your repositories"**

即會顯示自己的所有數據庫

點擊右上方 **"New"** 即可開始創建新數據庫

The screenshot shows the GitHub profile of user 'Myster7494'. The profile includes a bio, a profile picture, and a list of repositories. The 'New' button in the repository list is highlighted with a red box. The user's bio states: '我是個16歲的高中生，熱愛寫程式，以及聊天，也許。 I am a 16-year-old middle school student who loves programming and chatting, maybe.' The repositories listed are 'PolynomialFactorization', 'tao-bus-app', 'cap-countdown', and 'Myster7494'.

The screenshot shows the GitHub sidebar navigation menu for user 'Myster7494'. The menu items are: 'Set status', 'Your profile', 'Your repositories' (highlighted with a red box), 'Your Copilot', 'Your projects', 'Your stars', 'Your gists', 'Your organizations', 'Your enterprises', 'Your sponsors', 'Try Enterprise' (with a 'Free' badge), 'Feature preview', 'Settings', 'GitHub Docs', 'GitHub Support', 'GitHub Community', and 'Sign out'.

# 創建數據庫

按照說明填寫即可  
星號(\*) 表示必填

這些欄位或選項  
在創建數據庫後也可以更改  
唯獨數據庫名稱須謹慎填寫  
創建後再更改數據庫名稱  
可能會造成一些問題

## Create a new repository

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

Required fields are marked with an asterisk (\*).

### Repository template

No template ▾

Start your repository with a template repository's contents.

Owner \*

 Myster7494 ▾

 / 

Repository name \*

Great repository names are short and memorable. Need inspiration? How about [probable-computing-machine](#) ?

Description (optional)

☒  **Public** 公開(大家都看得到)

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private** 私人(自己才看得到)

You choose who can see and commit to this repository.

數據庫名稱

數據庫說明

# 創建數據庫

同樣按照說明填寫即可

內容填寫完成後

點擊右下角

"Create repository"

即完成創建數據庫

The screenshot shows the GitHub 'Create repository' form with several annotations:

- Initialize this repository with:**
  - ☐ **Add a README file** (Annotated with a blue box and the text "新增 README")  
This is where you can write a long description for your project. [Learn more about READMEs.](#)
- Add .gitignore** (Annotated with a yellow box and the text "新增 .gitignore")
  - .gitignore template: **None** (dropdown menu)
  - Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)
- Choose a license** (Annotated with a green box and the text "選擇開源協議")
  - License: **None** (dropdown menu)
  - A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

At the bottom, there is a note: "You are creating a public repository in your personal account." and a large green button labeled "Create repository" (Annotated with a yellow box and the text "創建數據庫").



# 創建數據庫

創建好後  
會跳轉到  
該頁面

TYSHIC / ExampleRepository

Repository not found

Type  to search

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

ExampleRepository Public

Edit Pins Watch 0 Fork 0 Star 0

**Start coding with Codespaces**  
Add a README file and start coding in a secure, configurable, and dedicated development environment.  
[Create a codespace](#)

**Give access to the people you work with**  
Ensure the right people and teams have access to this repository.  
[Manage access](#)

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

[Set up in Desktop](#) or **HTTPS** [SSH](#) <https://github.com/TYSHIC/ExampleRepository.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 "# ExampleRepository" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/TYSHIC/ExampleRepository.git
git push -u origin main
```

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

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

# Git 登入

在第一次使用 **Git** 之前，我們必須先告訴 **Git** 我們是誰  
使用下方 **Git** 指令用 **Github** 登入 **Git**：

```
git credential-manager github login
```

bash

接著會彈出一個視窗  
選擇 "**Sign in with your browser**"  
然後在瀏覽器中登入 **Github**  
或進行 **Github** 帳號授權



# 使用 Git

有了數據庫之後，我們便可以開始上傳檔案

但要使用 **Git** 上傳檔案之前

電腦需要先與

在 **Github** 上的遠端數據庫(remote repo)建立連線

使用 **Git** 指令建立連線最簡單的方式就是：

1. 複製(clone)遠端數據庫到本地(local)
2. 進入複製下來的本地數據庫(local repo)
3. 直接連線遠端數據庫

# 複製遠端數據庫

欲使用 **Git** 複製遠端數據庫，須使用該 **Git** 指令：

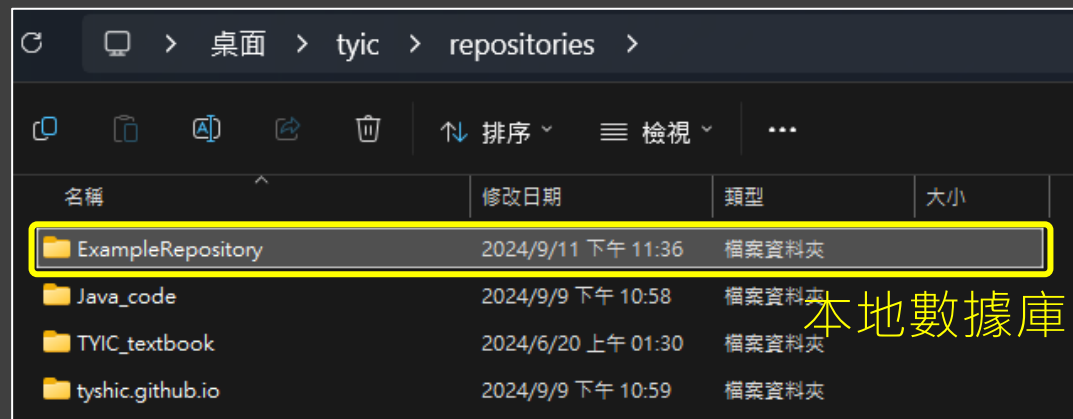
```
git clone 數據庫url bash
```



Windows PowerShell 窗口顯示了以下命令執行過程：

```
PS C:\Users\jacky\Desktop\tyic\repositories> git clone https://github.com/TYSHIC/ExampleRepository.git
Cloning into 'ExampleRepository'...
warning: You appear to have cloned an empty repository.
PS C:\Users\jacky\Desktop\tyic\repositories> |
```

使用該指令後會在  
工作目錄(**working directory**)  
出現一個新資料夾(**folder**)  
名稱與數據庫相同  
即為本地數據庫

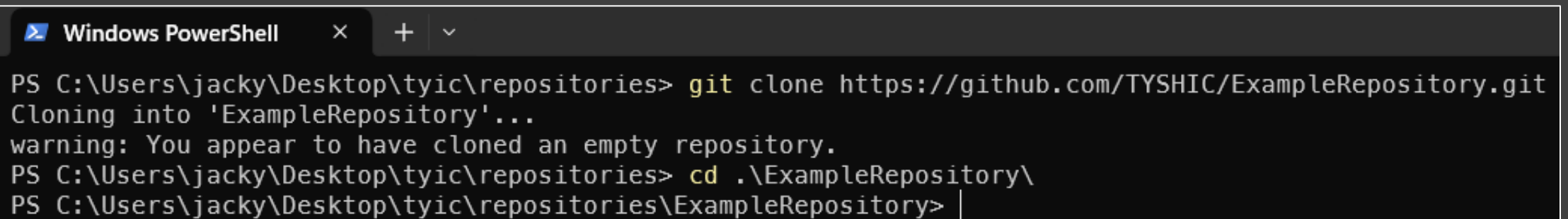


# 進入本地數據庫

欲進入本地數據庫，須在本地數據庫內開啟新的 `bash` 或在剛剛的 `bash` 中使用該 `Windows / Unix` 指令  
該指令會切換當前 `bash` 的工作目錄至指定資料夾

`cd` 資料夾名稱

`bash`



```
Windows PowerShell  x  +  v
PS C:\Users\jacky\Desktop\tyic\repositories> git clone https://github.com/TYSHIC/ExampleRepository.git
Cloning into 'ExampleRepository'...
warning: You appear to have cloned an empty repository.
PS C:\Users\jacky\Desktop\tyic\repositories> cd .\ExampleRepository\
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> |
```

# 連線遠端數據庫

在複製的本地數據庫中

輸入下方 **Git** 指令即可直接連線遠端數據庫：

```
git remote
```

bash

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git remote  
origin  
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> |
```

# Git 檔案狀態

Git 有四種檔案狀態：未追蹤(untrack)、未修改(unmodified)、未暫存(unstaged、Changes not staged for commit)、已暫存(staged、Changes to be committed)

可通過下方 Git 指令來查看當前的檔案狀態：

```
git status
```

bash

提交時只會提交已暫存的檔案，且已暫存檔案提交後會變回未修改而欲將未追蹤或未暫存的檔案變為已暫存，須使用下方 Git 指令：

特定資料夾或檔案：`git add 資料夾或檔案`

bash

所有檔案：`git add -A`      所有已修改檔案：`git add -u`

bash

bash

# Git 檔案狀態

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git status
On branch main

No commits yet

nothing to commit (create/copy files and use "git add" to track)
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> echo "# 範例倉庫" > README.md
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git status
On branch main

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git add .\README.md
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git status
On branch main

No commits yet

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

PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> █
```



# 提交與推送

提交就是將本地數據庫的變更變為新版本

每個提交都有一個編號

使用下方 **Git** 指令進行提交：

```
git commit -m "提交訊息"
```

bash

提交只會儲存在本地數據庫，不會自動推送(push)到遠端數據庫

要推送提交到遠端數據庫，須使用下方 **Git** 指令：

```
git push
```

bash

Git 小遊戲 

# 提交與推送

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git commit -m "init"
[main (root-commit) d57cc20] init
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 README.md
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 232 bytes | 232.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/TYSHIC/ExampleRepository.git
* [new branch]      main -> main
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> |
```

# 提交訊息

提交訊息最好要寫清楚該提交到底做了什麼  
但也必須保持精簡

常見的寫法是 "提交種類: 簡單說明"

提交種類有: **feat**(feature, 新功能)、**fix**(修復錯誤)  
**docs**(documentation, 更新說明)、**style**(格式變更)  
**refactor**(重構)、**chore**(雜項)、**revert**(撤回提交) 等  
但在首次提交, 常常使用 "**init**" 或 "**initial commit**"

# 提交與推送

The screenshot shows the GitHub interface for a repository named 'ExampleRepository' by user 'TYSHIC'. The repository is public and has 1 star, 0 forks, and 0 watches. The main branch is 'main', and there is 1 branch and 0 tags. The repository contains a single commit by 'Myster7494' titled 'init', which includes a 'README.md' file. The 'README.md' file content is visible, showing a license header. The right sidebar shows the 'About' section with a description, 'Readme', 'Custom properties', '1 star', '0 watching', '0 forks', and 'Report repository'. The 'Releases' section shows 'No releases published' and a link to 'Create a new release'. The 'Packages' section shows 'No packages published' and a link to 'Publish your first package'.

Annotations on the screenshot:

- A blue box highlights the branch selection area (main, 1 Branch, 0 Tags) with the label "分支" (Branch).
- A green box highlights the commit history table with the label "最近一筆提交" (Latest commit).
- A yellow box highlights the file list (README.md) and the README content area with the label "數據庫內容" (Database content).
- A yellow box highlights the README content area with the label "README、LICENSE 等" (README, LICENSE, etc.).

# 提交與推送

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> 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")
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git add -u
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> 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

PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git commit -m
"docs: changed README.md file encoding to UTF-8"
[main b4947d8] docs: changed README.md file encoding to UTF-8
 1 file changed, 0 insertions(+), 0 deletions(-)
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 296 bytes | 296.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/TYSHIC/ExampleRepository.git
 d57cc20..b4947d8  main -> main
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> |
```

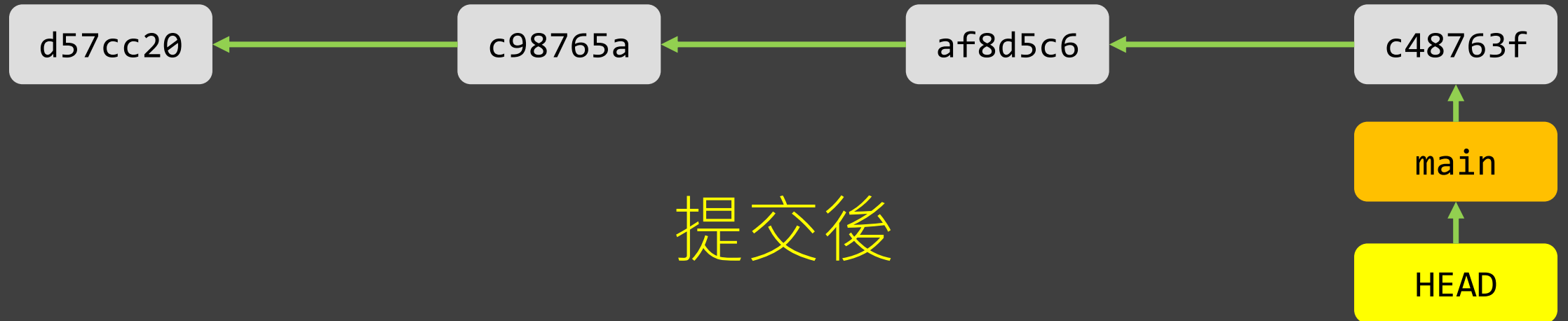
The screenshot shows the GitHub interface for a repository named 'ExampleRepository'. At the top, it indicates the repository is 'Public'. Below this, there are buttons for 'Edit Pins' and 'Watch' (with a count of 0). The main section shows the 'main' branch with 1 branch and 0 tags. A search bar and buttons for 'Add file' and 'Code' are present. A commit by 'Myster7494' is displayed, titled 'docs: changed README.md file encoding to UTF-8', with a commit hash of 'b4947d8' and a timestamp of '6 minutes ago'. Below the commit, the 'README.md' file is shown, containing the text '測試倉庫' (Test Repository).

The screenshot shows the 'Commits' page for the 'ExampleRepository'. It lists the commit history for the 'main' branch. The most recent commit is by 'Myster7494' with the message 'docs: changed README.md file encoding to UTF-8' and hash 'b4947d8', committed 'yesterday'. Below it is the 'init' commit by 'Myster7494' with hash 'd57cc20', also committed 'yesterday'. The page includes filters for 'All users' and 'All time'.

# 分支

分支是一個指標，指向某個提交(版本)  
在某個分支上做的修改和提交不會影響到其他分支  
每個數據庫至少要一個分支  
通常名稱為 "main" 或 "master"  
在分支出現新提交後，分支會指向新提交  
"HEAD" 是一個指標，指向某個分支  
指向的分支即成為當前分支  
提交只會提交到當前分支上

# 單分支



# 多分支

提交前



提交後





# 新建與切換分支

欲新建分支，須使用下方 **Git** 指令：

```
git branch 新分支名稱
```

bash

該指令會新建一個分支，且指向當前分支指向的提交

**HEAD** 不會自動指向新分支，即新分支不會自動成為當前分支

欲使 **HEAD** 指向某個分支，即讓某個分支成為當前分支

也就是切換分支，須使用下方 **Git** 指令：

```
git checkout 目標分支名稱
```

bash

或

```
git switch 目標分支名稱
```

bash

切換分支之後，資料夾內的檔案都會被替換成該分支的檔案

# 推送與刪除分支

在推送本地數據庫的未追蹤的分支或新分支到遠端數據庫時  
需在推送時加上 "-u" 引數：

```
git push -u origin 分支名稱
```

bash

這會使得本地數據庫的分支追蹤遠端數據庫的分支  
若在遠端數據庫上沒有此分支，則會在遠端數據庫上創建該分支

若要刪除分支，須切換到其他分支，並使用下方 Git 指令：

```
git branch -d 分支名稱
```

bash

# 分支

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git status
On branch main
Your branch is up to date with 'origin/main'.
```

```
nothing to commit, working tree clean
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> dir
```

目錄: C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository


| Mode   | LastWriteTime      | Length | Name      |
|--------|--------------------|--------|-----------|
| -a---- | 2024/9/12 下午 11:32 | 16     | README.md |

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git branch feature
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git switch feature
Switched to branch 'feature'
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> dir
```


目錄: C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository


| Mode   | LastWriteTime      | Length | Name      |
|--------|--------------------|--------|-----------|
| -a---- | 2024/8/15 下午 02:05 | 326    | Main.java |
| -a---- | 2024/9/12 下午 11:32 | 16     | README.md |

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git add Main.java
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git commit -m "feat: add Main.java"
[feature 5c95fd8] feat: add Main.java
1 file changed, 11 insertions(+)
create mode 100644 Main.java
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git push -u origin feature
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 20 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 538 bytes | 134.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote: https://github.com/TYSHIC/ExampleRepository/pull/new/feature
remote:
To https://github.com/TYSHIC/ExampleRepository.git
 * [new branch] feature -> feature
branch 'feature' set up to track 'origin/feature'.
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> |
```


 **feature** had recent pushes 5 minutes ago

[Compare & pull request](#)

 **feature**

 **2 Branches**

 **0 Tags**

 Go to file

 **t**


[Add file](#)


[Code](#)

This branch is **1 commit ahead of** **main**.

[Contribute](#)


 **Myster7494** feat: add Main.java

5c95fd8 · 5 minutes ago  **3 Commits**

 **Main.java**

feat: add Main.java

5 minutes ago

 **README.md**

docs: changed README.md file encoding to UTF-8

yesterday

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git switch main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> dir
```

目錄: C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository

| Mode   | LastWriteTime      | Length | Name      |
|--------|--------------------|--------|-----------|
| -a---- | 2024/9/12 下午 11:32 | 16     | README.md |

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> |
```

# 分支

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> dir




    目錄：C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository


Mode                LastWriteTime         Length Name
----                -
-a-----          2024/9/12 下午 11:32             16 README.md


PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git add -A
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> 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)
        new file:   LICENSE


PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git commit -m "docs: add LICENSE"
[main c083eae] docs: add LICENSE
1 file changed, 32 insertions(+)
create mode 100644 LICENSE
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 20 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.15 KiB | 1.15 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/TYSHIC/ExampleRepository.git
 b4947d8..c083eae  main -> main
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> |
```


 **main**  2 Branches  0 Tags




 Add file


 Code


 **Myster7494** docs: add LICENSE c083eae · 1 minute ago  3 Commits



 LICENSE docs: add LICENSE 1 minute ago



 README.md docs: changed README.md file encoding to UTF-8 2 days ago


 **feature**  2 Branches  0 Tags


 Add file

 Code

This branch is 1 commit ahead of, 1 commit behind  main .  Contribute

 **Myster7494** feat: add Main.java 5c95fd8 · 11 hours ago  3 Commits

 Main.java feat: add Main.java 11 hours ago

 README.md docs: changed README.md file encoding to UTF-8 2 days ago

# 合併

若想將兩個分支合在一起，可以考慮合併(merge)

使用下方 Git 指令進行合併：

```
git merge 合併分支名稱 -m "提交訊息"
```

bash

這會將合併分支的修改添加到當前分支，並自動提交

若當前分支為合併分支的親代提交

則只會將當前分支指向合併分支，而不會進行產生新提交

但若合併的兩個分支對同一處都進行了修改

就會發生合併衝突(merge conflict)，合併中止

需要手動修改有衝突的地方並手動提交，才能順利完成合併

# 補充：合併衝突解決

在發生衝突時，打開發生衝突的檔案，會發現下方區段：

```
<<<<<<< HEAD:衝突檔案  
當前分支的衝突內容
```

```
=====
```

```
合併分支的衝突內容
```

```
>>>>>>> 合併分支:衝突檔案
```

text

將這整段文字刪除，並改為想要的內容

編輯完成後，再使用下方 **Git** 指令標記解決：

```
git add 衝突檔案
```

bash

解決衝突後再手動提交即可完成合併

# 合併

合併 feature 到 main

合併前



合併後



# 合併

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git merge feature -m "Merge branch 'feature' into main"
Merge made by the 'ort' strategy.
 Main.java | 11 ++++++++
 1 file changed, 11 insertions(+)
 create mode 100644 Main.java
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 20 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 343 bytes | 343.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/TYSHIC/ExampleRepository.git
 c083eae..7f70e34  main -> main
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> |
```



# 拉取

如果當前分支在遠端數據庫有了新提交  
本地數據庫就必須拉取(**pull**)當前分支的新提交  
使用下方 **Git** 指令來拉取新提交：

```
git pull
```

bash

在當前分支未拉取前  
無法將本地數據庫變更提交到遠端數據庫  
若本地數據庫已有新提交才進行拉取  
則會將本地數據庫新提交的親代提交變為遠端數據庫的新提交  
並且可能會出現合併衝突

# 拉取

遠端數據庫

更改 README.md

本地數據庫

拉取該提交

main 2 Branches 0 Tags  Add file Code

**Myster7494** docs: update README.md 462706e · 1 minute ago 6 Commits

|           |                        |              |
|-----------|------------------------|--------------|
| LICENSE   | docs: add LICENSE      | 2 hours ago  |
| Main.java | feat: add Main.java    | 14 hours ago |
| README.md | docs: update README.md | 1 minute ago |

README BSD-3-Clause-Clear license

## 測試數據庫

用於教學演示

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 983 bytes | 245.00 KiB/s, done.
From https://github.com/TYSHIC/ExampleRepository
   7f70e34..462706e  main       -> origin/main
Updating 7f70e34..462706e
Fast-forward
 README.md | 4 +++-
 1 file changed, 3 insertions(+), 1 deletion(-)
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> |
```

# Github 功能

The screenshot displays the GitHub interface for a repository named 'sampleRepository'. The top navigation bar includes icons for menu, GitHub logo, and a search bar. Below this, a row of tabs is visible: '問題' (Issues), '動作' (Actions), '維基' (Wiki), and '設定' (Settings). A secondary row of tabs includes '<> Code', 'Issues 1', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. On the right side of the repository header, there are buttons for '分叉' (Fork) and '星星' (Stars). Below the repository name, there are buttons for '程式碼' (Code), '拉取請求' (Pull requests), and '洞察' (Insights). The main content area shows a list of commits, with the most recent one by 'Myster7494' updating the README.md file. The right sidebar contains the 'About' section, which is highlighted with a red border, and a '關於' (About) button. The 'About' section lists repository details such as 'Readme', 'BSD-3-Clause-Clear license', 'Activity', 'Custom properties', '1 star', '0 watching', and '0 forks'.

問題 sampleRepository 動作 維基 設定

<> Code Issues 1 Pull requests Actions Projects Wiki Security Insights Settings

分叉 星星

程式碼 拉取請求 Public 洞察 Edit Pins Watch 0 Fork 0 Starred 1

main 2 Branches 0 Tags Go to file Add file <> Code

Myster7494 docs: update README.md 462706e · 37 minutes ago 6 Commits

LICENSE docs: add LICENSE 3 hours ago

Main.java feat: add Main.java 14 hours ago

README.md docs: update README.md 37 minutes ago

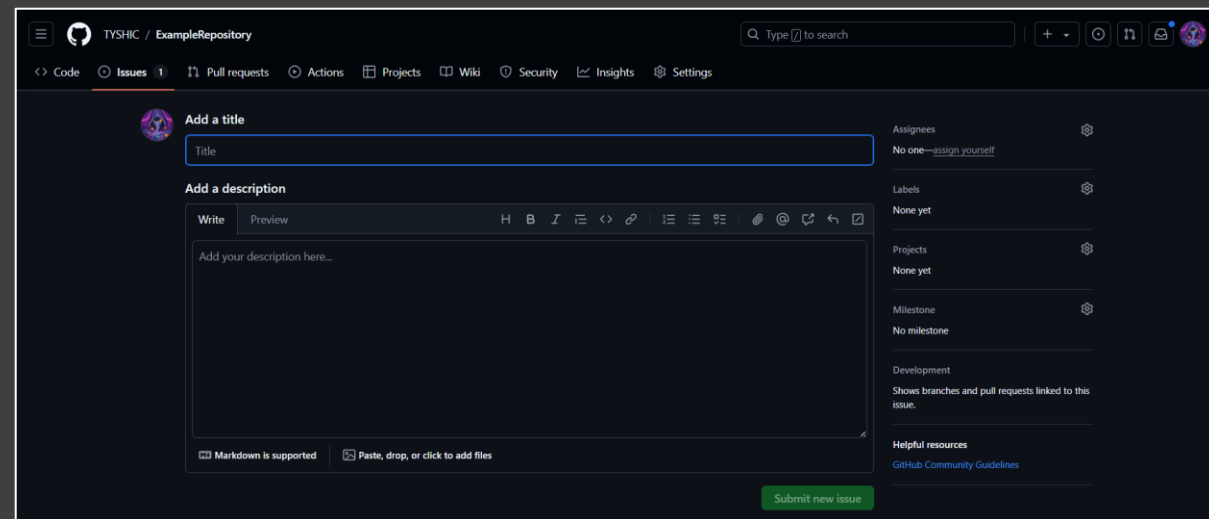
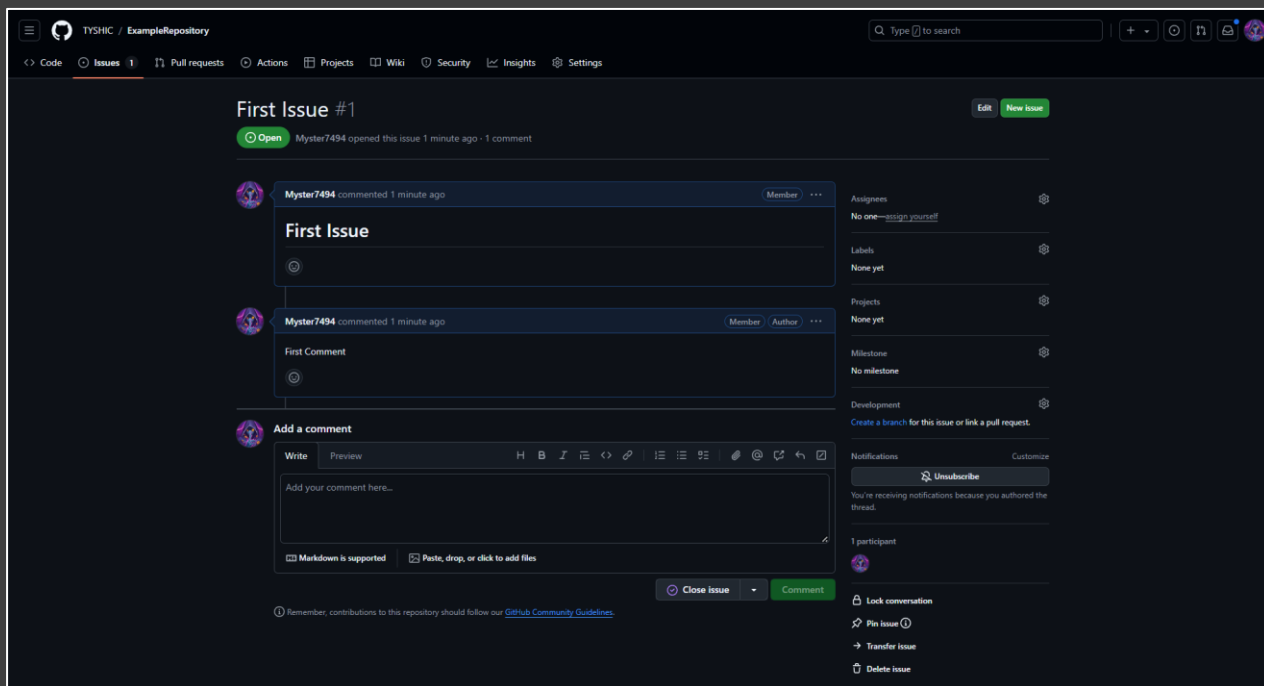
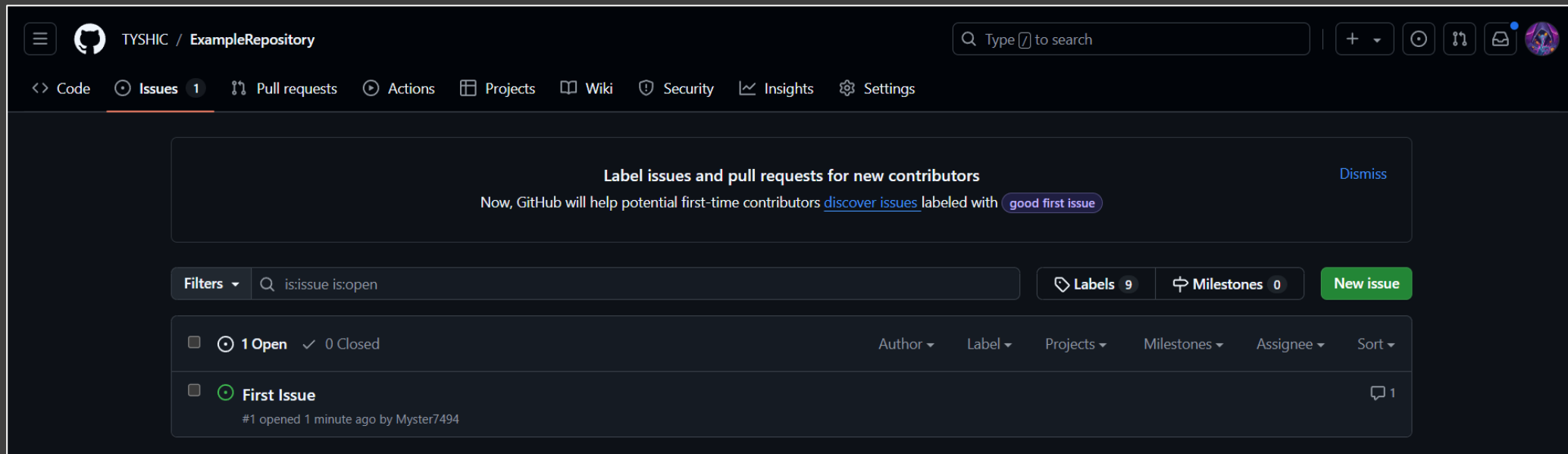
README BSD-3-Clause-Clear license

About 關於

No description, website, or topics provided.

Readme BSD-3-Clause-Clear license Activity Custom properties 1 star 0 watching 0 forks Report repository

# 問題



# 分叉

ExampleRepository Public

Edit Pins

Watch 0

Fork 0

Starred 1

Fork your own copy of TYSHIC/ExampleRepository

分叉(fork)就是把別人的數據庫複製一份變成自己的  
點擊右上角  
"Fork" 按鈕  
即可分叉該數據庫

## Create a new fork

A fork is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project.

Required fields are marked with an asterisk (\*).

Owner \*

Myster7494

Repository name \*

ExampleRepository

ExampleRepository is available.

By default, forks are named the same as their upstream repository. You can customize the name to distinguish it further.

Description (optional)

是否只複製預設分支

☒ Copy the main branch only

Contribute back to TYSHIC/ExampleRepository by adding your own branch. [Learn more.](#)

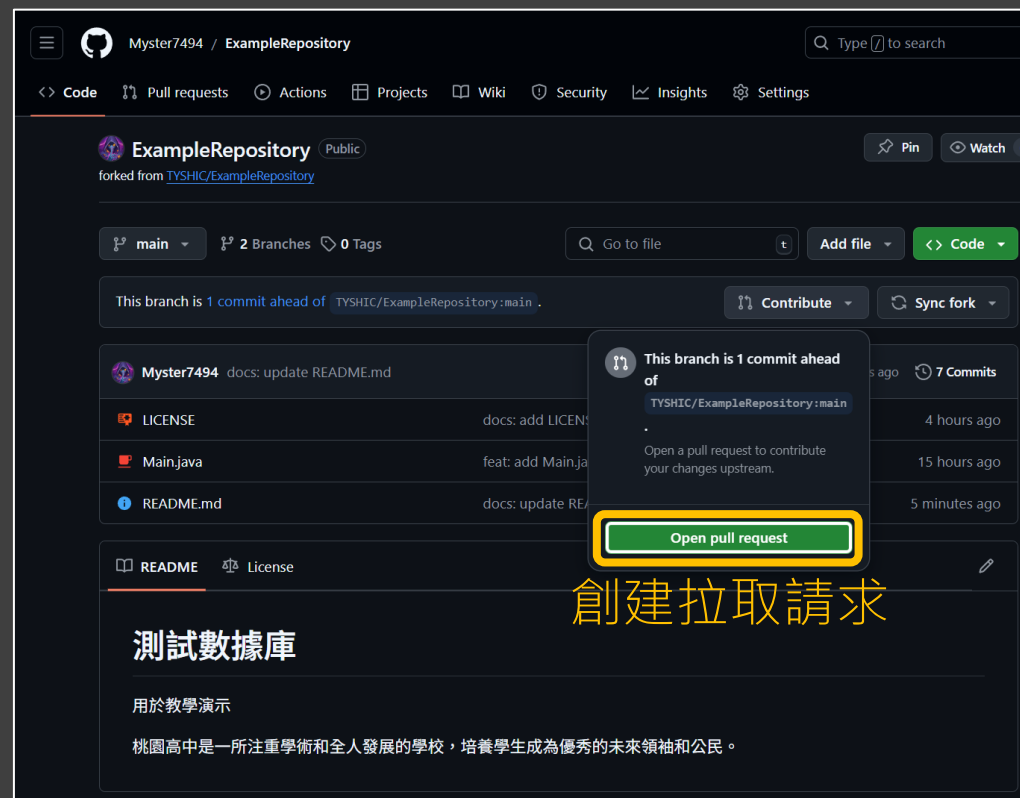
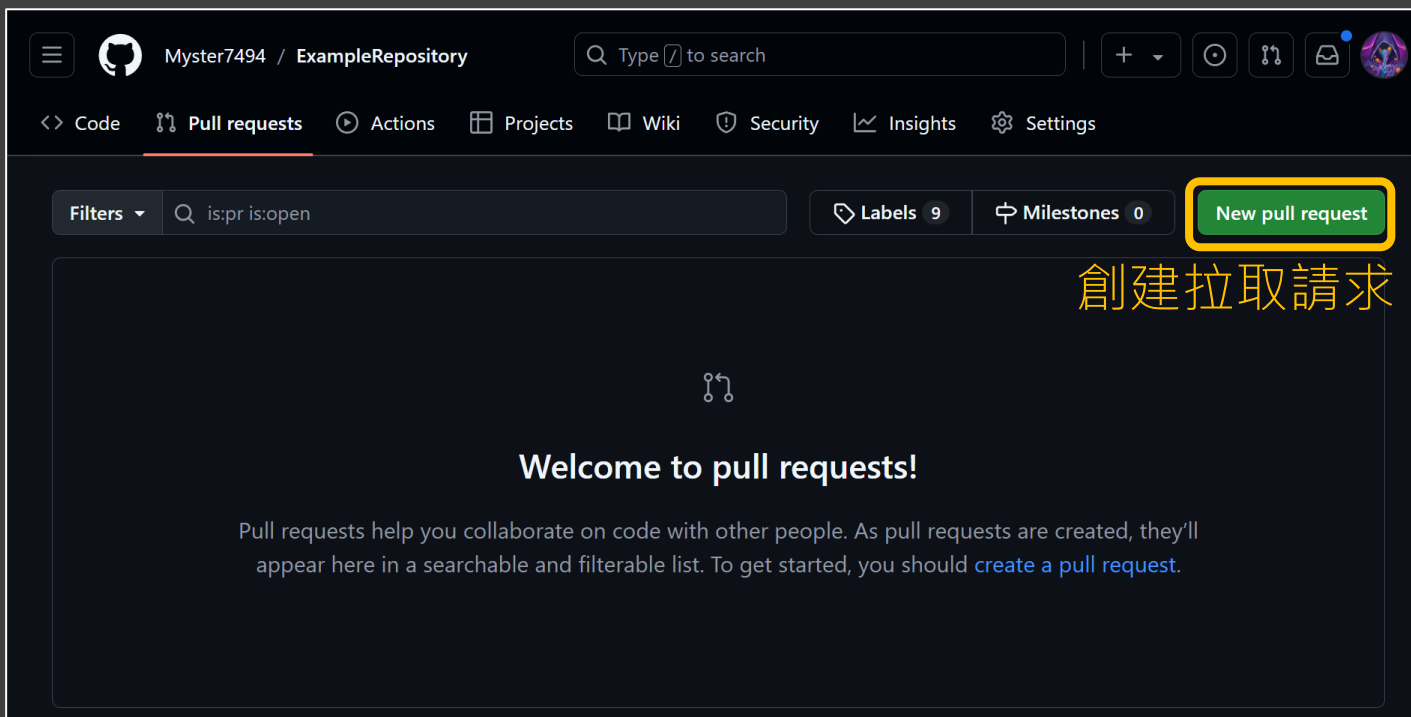
*i* You are creating a fork in your personal account.

創建分叉

Create fork

# 拉取請求

拉取請求(**pull request**，簡稱 **PR**)與拉取沒什麼大關係  
實際上就是請求(**request**)數據庫擁有者合併某個分支  
包括該數據庫的分支或是分叉數據庫的分支  
可在拉取請求頁面或分支創建拉取請求



# 創建拉取請求

## Comparing changes

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](#).

base repository: TYSHIC/ExampleRepository base: main head repository: Myster7494/ExampleRepository compare: main

✓ **Able to merge.** These branches can be automatically merged.

**Add a title**

docs: update README.md

**Add a description**

Write Preview H B I

Add your description here...

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

☒ Allow edits by maintainers **Create pull request**

**Reviewers** No reviews

**Assignees** No one—assign yourself

**Labels** None yet

**Projects** None yet

**Milestone** No milestone

**Development** Use [closing keywords](#) in the description to automatically close issues

[Helpful resources](#)

1 commit 1 file changed 1 contributor

Commits on Sep 14, 2024

docs: update README.md  
Myster7494 committed 13 minutes ago

Showing 1 changed file with 2 additions and 0 deletions. Split Unified

2 README.md

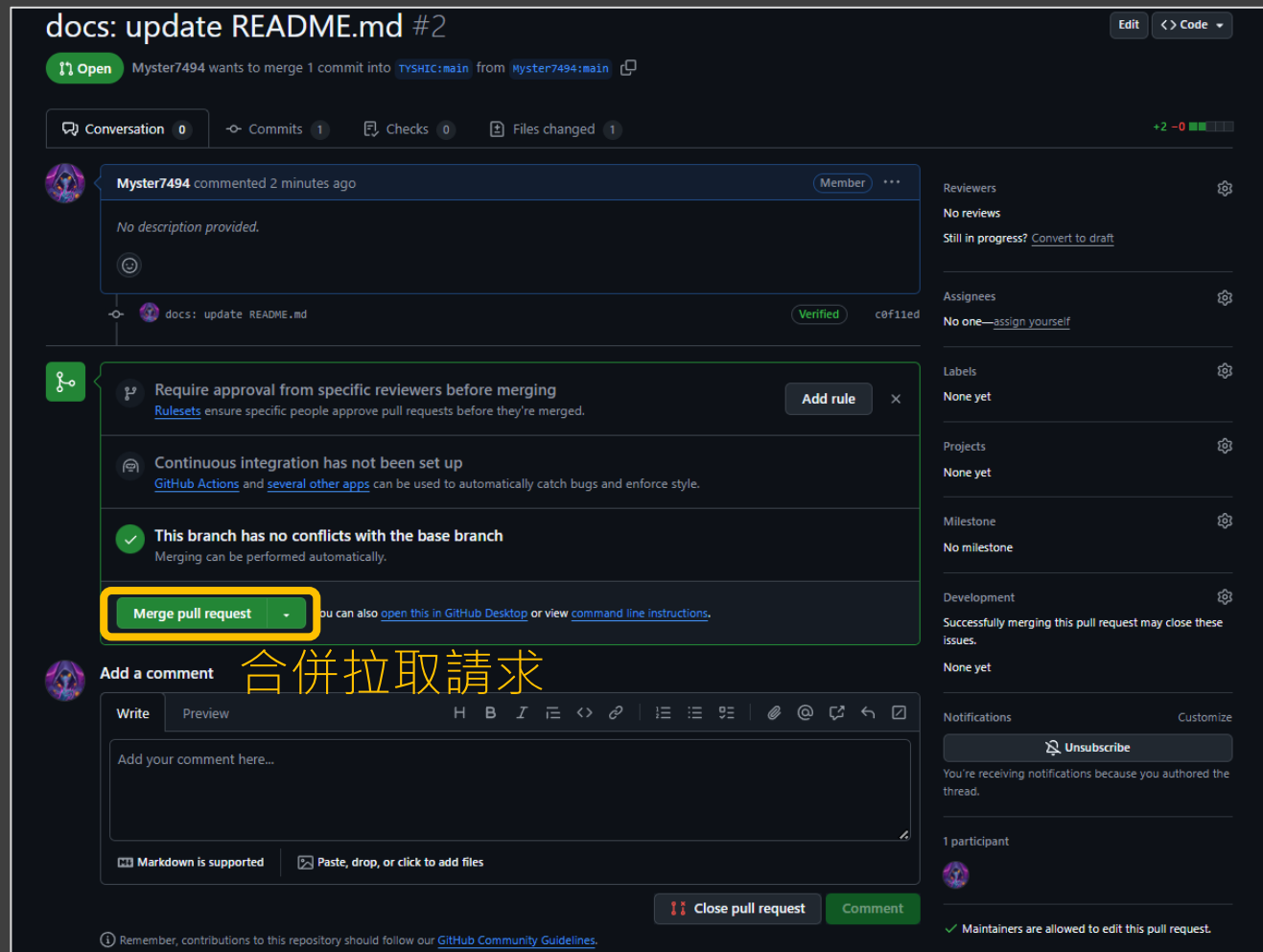
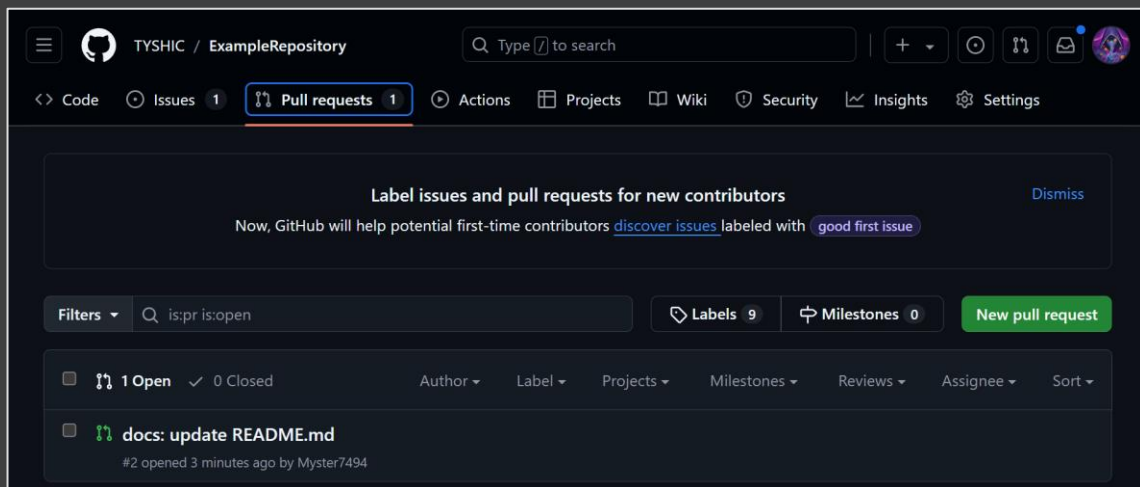
|     |                 |   |   |
|-----|-----------------|---|---|
| ... | @@ -1,3 +1,5 @@ | 1 | # 測試數據庫                                 |
| 1   | # 測試數據庫         | 2 |   |
| 2   |                 | 3 | 用於教學演示                                  |
| 3   | 用於教學演示          | 4 | +                                       |
|     |                 | 5 | + 桃園高中是一所注重學術和全人發展的學校，培養學生成為優秀的未來領袖和公民。 |

# 同意拉取請求

創建拉取請求後

數據庫擁有者便可在拉取請求頁面  
看到該拉取請求

並可選擇合併拉取請求或其他動作





# .gitignore

若有些檔案只想留在本地數據庫  
而不想被上傳到遠端數據庫  
可以創建 **.gitignore** 檔案  
其就是一個名稱為 **".gitignore"** 的文字文件  
可以在其中定義排除的路徑，路徑可為檔案或目錄  
但須注意，**.gitignore** 檔案只能排除未追蹤的檔案  
已追蹤的檔案是無法被排除的

# .gitignore

**.gitignore** 檔案的格式如下：

1. 以 '#' 開頭行為**註解**，否則為**排除路徑**
2. **路徑**以 '/' 開頭表示**路徑**相對於此 **.gitignore** 檔案  
否則該**路徑**可以相對於任何位置
4. **路徑**中間的 '/' 用於分隔**目錄**
5. **路徑**以 '/' 結尾表示只會排除**目錄**
6. '\*' 可以表示任何不包含 '/' 的文字
7. '?' 可以表示任何不包含 '/' 的字元(一個字)
8. "[]" 可以表示中括號內的任一字元(一個字)
9. **路徑**以 "\*\*/" 開頭表示該**路徑**可以相對於任何位置
10. **路徑**以 "/\*" 結尾表示該**路徑**指向的**目錄**內的任何東西
11. **路徑**中間的 "\*\*/" 表示中間可以有任意多層任意**目錄**

```
# gradle
.gradle/
build/
out/
classes/

# eclipse
*.launch

# idea
.idea/
*.iml
*.ipr
*.iws

# vscode
.settings/
.vscode/
bin/
.classpath
.project

# macos
*.DS_Store

# fabric
run/

# java
hs_err_*.log
replay_*.log
*.hprof
*.jfr

.gitignore
```

# .gitignore

```
# .gitignore file
/hi.txt          .gitignore
```

```
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> dir

Directory: C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository

Mode                LastWriteTime         Length Name
----                -
-a-----          2024/9/9   下午 10:58        1722 LICENSE
-a-----          2024/9/14   下午 12:28         326 Main.java
-a-----          2024/9/14   下午 03:05          41 README.md

PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> dir

Directory: C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository

Mode                LastWriteTime         Length Name
----                -
-a-----          2024/9/15   下午 05:45           7 .gitignore
-a-----          2024/9/15   下午 05:45           2 hi.txt
-a-----          2024/9/9   下午 10:58        1722 LICENSE
-a-----          2024/9/14   下午 12:28         326 Main.java
-a-----          2024/9/14   下午 03:05          41 README.md

PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository> git status
On branch main
Your branch is up to date with 'origin/main'.

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

nothing added to commit but untracked files present (use "git add" to track)
PS C:\Users\jacky\Desktop\tyic\repositories\ExampleRepository>
```