

# Git 實作練習紀錄

以下操作環境為 Windows 作業系統

## Git 簡介

Git 是一種版本控制系統，可用來追蹤紀錄，並讓你與其他專案參與人員共同協作。

## 用途：版本控制

Git 會記錄「誰」在「哪個時間點」修改了「哪些內容」，方便參與專案的相關人員瞭解整個脈絡或回復到某一個版本。在沒有版本控制的協助下，我們便必須時時刻刻將專案儲存到新的資料夾，就好像以前用 Word 寫報告時，若我們要記錄每一個時期所做的變更，就必須一直「另存新檔」（不過通常應該不會有人這樣做就是了🤦），相當麻煩。

## 常用基本指令

- 建立目錄（資料夾）

```
mkdir [目錄名稱]
```

- 複製檔案

```
copy [要複製的檔案名稱] [複製出來的檔案名稱]
```

- 刪除檔案

```
del [要刪除的檔案名稱]
```

- 刪除同一副檔名的所有檔案

```
del *. [副檔名]
```

- 更改檔案名稱

```
move [要更名的檔案名稱] [更名後的檔案名稱]
```

- Git 相關指令

## Git Commands

- Clone -> Bring a repository that is hosted somewhere like Github into a folder on your local machine
- add -> Track your files and changes in Git
- commit -> Save your files in Git
- push -> Upload Git commits to a remote repo, like Github
- pull -> Download changes from remote repo to your local machine, the opposite of push

## 進行控管

- 將檔案交給 Git 控管

```
git add [檔案名稱]
```

- 一次將同一副檔名的所有檔案交給 Git 控管

```
git add *.[副檔名]
```

- 查看狀態

```
git status
```

- 進行 commit

```
math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$ git commit -m "Added index.html" -m "some description"
```

第一個「-m」為 commit 的主要標題，第二個「-m」則為補充說明

- 使用 git push 將資料上傳到 GitHub

```
math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$ git push
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 386 bytes | 386.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:HawkingHuang/git-exercise.git
   3bcfa99..02d5c95  main -> main
```

- 將 add 過的檔案重設

```
git reset [欲重設的檔案名稱]
```

- 若已經 commit，則請使用以下指令

```
git reset HEAD~[數字 (要回到之前的第幾個版本)]
```

沒有加「~數字」的話就是回復到最新提交版本

- 使用「git log」指令查看紀錄時，可以按下空白鍵滾動查看紀錄

```
math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$ git log
commit 7e8717b101e416778a5929626160e7219abe5e6b (HEAD -> main)
Merge: a329b07 335b63c
Author: HawkingHuang <math0915@gmail.com>
Date: Sun Sep 24 23:51:39 2023 +0800

    Merge branch 'main' of github.com:HawkingHuang/git-exercise

commit 335b63c25f1a9bb39f38ab09fddaf29fe934ab4f (origin/main, origin/HEAD)
Merge: 02d5c95 013b1f6
Author: HawkingHuang <126544105+HawkingHuang@users.noreply.github.com>
Date: Sun Sep 24 23:44:50 2023 +0800

    Merge pull request #1 from HawkingHuang/feature

    Updated README

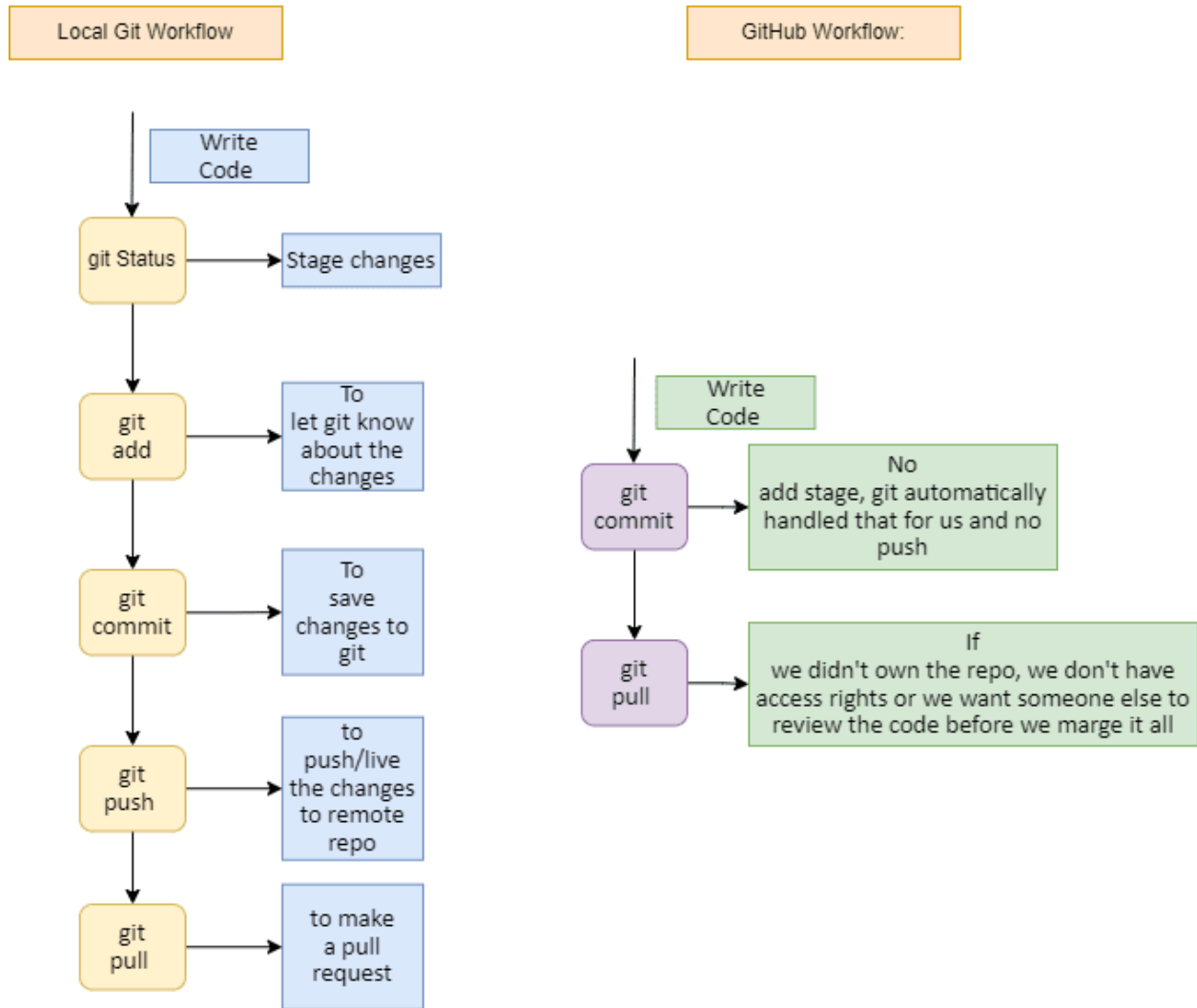
commit a329b072907fb008a94fa3bebf2ca132afac49a3
Author: HawkingHuang <math0915@gmail.com>
Date: Sun Sep 24 23:09:35 2023 +0800
```

- 若要回復版本同時重設檔案中的內容

```
git reset --hard [commit hash]
```

## 兩種建立專案的方式

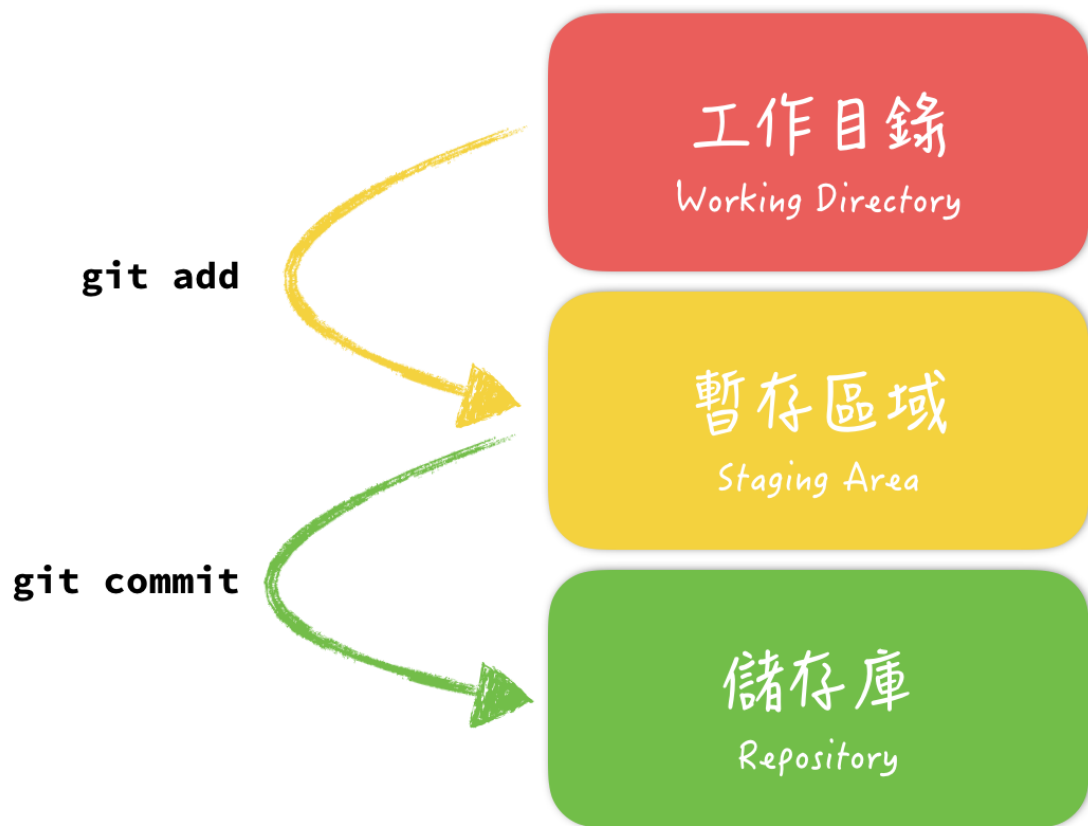
## GitHub VS Local Git Workflow



一種是先在本地建立，再透過 add、commit 及 push 等與線上同步；另一個方式則時直接在線上完成相關作業

## 工作區、暫存區與儲存庫

- 使用「git add」指令後，檔案會移往暫存區；使用「git commit」指令後，檔案則會移往儲存庫。



- 將兩個步驟合在一起

```
git commit -a -m "[內容]"
```

## Commit 的時機

- 基本上想 commit 就 commit。

## 刪除檔案與變更檔名

- 使檔案脫離 Git 控管

```
git rm [檔案名稱] --cached
```

- 變更檔案名稱

```
move [要更名的檔案名稱] [更名後的檔案名稱]
```

- 少做一步「add」動作的指令：

```
git mv [要更名的檔案名稱] [更名後的檔案名稱]
```

## 分支

- 建立分支可以避免尚未完成的程式碼污染到開放給終端使用者存取的服務或功能，讓開發人員可以在獨立的環境中作業，並在完成任務時合併到主要的分支
- 查詢目前分支

```
git branch
```

- 新增分支

```
git branch [欲新增分支的名稱]
```

- 刪除分支

```
git branch -d [欲刪除分支的名稱]
```

- 切換分支

```
git checkout [欲切換至分支的名稱]
```

- 輸入較長指令時如果前面的指令都一樣，可以輸入後方不一樣的部分並透過 Tab 鍵自動完成輸入
- 修改內容後，檔案右方會出現「M」，代表檔案被修改（modified）了

```

3  Hi there!
4
5  ## Subheader
6
7  Watch tutorial on YouTube.
8
9  ## Learning Git is painful
10
11 Hahaha
12

```

- 建立除了 main/master 以外的第二個分支後，若在原本的分支修改內容並 commit 後，切換到另一個分支時，剛剛修改的內容並不會出現

```

7  Watch tutorial on YouTube.
8
9  ## Learning Git is painful
10
11 Hahahaha
12

math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$ git checkout fe
error: pathspec 'fe' did not match any file(s) known to git

math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$ git checkout feature
Switched to branch 'feature'

math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (feature)
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)

math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$

```



這邊在 main 分支將「Hahaha」修改為「Hahahaha」，但是在以下 feature 分支中「Hahaha」還是不變

```
7 Watch tutorial on YouTube.
8
9 ## Learning Git is painful
10
11 Hahaha
12
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS MEMORY XRTOS SERIAL MONITOR CODEWHISPERER REFERENCE LOG

```
math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$ git checkout feature
Switched to branch 'feature'

math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (feature)
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)

math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$ git checkout feature
Switched to branch 'feature'

math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (feature)
$
```

> OUTLINE  
> TIMELINE

feature 0 0 0 0 ✓ AWS: AWS Builder ID ✓ CodeWhisperer Git Graph

- 查詢分支之間的差異

```
git diff [另一個分支的名稱]
```

```
math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$ git diff feature
diff --git a/README.md b/README.md
index a0a46d9..481af44 100644
--- a/README.md
+++ b/README.md
@@ -8,4 +8,4 @@ Watch tutorial on YouTube.

## Learning Git is painful

-Hahaha
+Hahahaha

math0@LAPTOP-4STP7PA4 MINGW64 ~/OneDrive/Monosparta/Git-exercise/git-exercise (main)
$
```

> OUTLINE  
> TIMELINE

main 01 21 0 0 0 0 ✓ AWS: AWS Builder ID ✓ CodeWhisperer Git Graph


- 修改舊檔（新檔案不適用）後要進行 commit 時，可使用以下指令：

```
git commit -am "[註解]"
```


## PR 是什麼？

Pull request（PR）是指請求將程式碼匯入到另一個分支。以下為範例成果：



### Updated README #1



 **Merged** HawkingHuang merged 1 commit into `main` from `feature` 3 minutes ago

Conversation 0 Commits 1 Checks 0 Files changed 1


 **HawkingHuang** commented 6 minutes ago Owner ...

Simple as the title.

  Updated README 013b1f6

  HawkingHuang merged commit `335b63c` into `main` 3 minutes ago Revert

---

 **Pull request successfully merged and closed** Delete branch

You're all set—the `feature` branch can be safely deleted.

## Git Flow

Git Flow 主要分為 master、develop、hotfix、release 及 feature 這五個分支。

- master：穩定版本
- develop：開發分支
- hotfix：應對緊急狀況的分支
- release：進行測試的分支
- feature：開發新功能時新增的分支（從 develop 延伸出來）

## .gitignore 配置

若要忽略某一個檔案，可使用以下指令建立 .gitignore 檔

```
touch .gitignore
```

在 .gitignore 加入欲忽略的檔案或資料夾，之後執行 `git status` 指令後，系統就會自動忽略相關檔案或資料夾的變更。