```
56  ```
57  > 「add .」是將所有資料加入追蹤，也可以使用「add 檔案名稱」
58
59  加入儲存庫
60  ```bash=
61  git commit -m "message"
62  # 單引號或雙引號都可以寫message，但寫英文message時單引號會誤判報錯
63  ```
64  > -m 指的是message。在commit的時候一定要有-m，否則無法commit
65  ## 紀錄
66  Git 紀錄
67  ```bash=
68  git log
69  ```
70  Git紀錄用一行顯示
71  ```bash=
72  git log --oneline
73  ```
74  查看所有紀錄
75  ```bash=
```

空白寬度： 4 SUBLIME 長度：2327

Git 版本控制

🔒

什麼是Git?

- Linus Torvalds 2005
- 分散式
- 開源
- 免費

安裝Git

https://git-scm.com/ (https://git-scm.com/)

設定-git config

檢查設定

```
1  git config --list
```

設定/修改使用者

```
1  git config --global user.name "你的名子
```

設定/修改EMAIL

```
1  git config --global user.email "你的EMA
```

檢查使用者

```
1  git config --global user.name
```

Git初始化

先移到要版本控制的資料夾內並且執行:

```
git init
```

常用指令

Windows	Mac / Linux	說明
cd	cd	切換目錄
mkdir	mkdir	新增資料夾
rmdir	rmdir	移除資料夾
dir	ls	查看資料夾
echo >>	touch	新增檔案
del	rm	刪除檔案
cls	clear	清除畫面
cd	pwd	列出目前路徑

Working、Staging & Repository

查看檔案狀態

```
git status
```

加入追蹤(tracked)

```
git add .
```

> 「add .」是將所有資料加入追蹤，也可以使用「add 檔案

加入儲存庫

```
git commit -m "message"
# 單引號或雙引號都可以寫message，但寫英文message時單
```

> -m 指的是message。在commit的時候一定要有-m，否則

紀錄

Git 紀錄

```
git log
```

Git紀錄用一行顯示

```
git log --oneline
```

查看所有紀錄

```
git reflog
```

or

```
git log -g
```

分支

查詢分支

```
git branch

git branch -a
#近端與遠端
```

新增分支

```
git branch 分支名稱
```

移除分支

```
git branch 分支名稱 -d
```

切換分支

```
git checkout 分支名稱
```

合併分支

```
git merge 分支名稱
```

標籤

查詢標籤

```
git tag
```

新增標籤

```
git tag 標籤名稱
```

移除標籤

```
git tag 標籤名稱 -d
```

上傳標籤

```
git push origin master --tags
```

從暫存區移除檔案

將myTxt.txt檔從暫存區移除檔案

```
git rm --cached myTxt.txt
```

移除所有暫存區的檔案

```
git rm --cached -r .
```

還原

在還沒追蹤之前，將內容還原成上一次修改的狀態

```
git checkout index.html
```

回復上一次的commit

- 使用cmd

```
git reset HEAD~1 --hard
```

~1 代表回復上一個版本， ~2 代表回復上兩個版本

- 使用git bash

```
git reset HEAD^
```

回到特定版本

```
# 硬回復 回到該版本commit之後的狀態
git reset 版本號 --hard

# 軟回復 回到該版本尚未commit之後的狀態
git reset 版本號 --soft

# 回復 回到該版本尚未add之後的狀態
git reset 版本號
```

推送至遠端

連線

```
git remote add origin 連線位置
# origin為連線的名稱，可以自訂

```

推送

```
git push origin master
# master為分支名稱


```

如果連線名稱origin與分支名稱master都是固定的情況下，可以省略origin 與 master

```
git push origin master -u
# 串流推送

git push
```

下載

當遠端專案版本較新，本地端較舊，可以用pull來下載更新

```
git pull origin master
```

取得儲存庫

本地若沒有專案，可用clone取得

```
git clone 儲存庫位置
```