目錄

[git基礎指令 1](#_Toc139525552)

[fast-forward機制 3](#_Toc139525553)

[將內容推送到遠端的儲存庫 3](#_Toc139525554)

[merge機制類型 3](#_Toc139525555)

[ff -- no-commit先測試再commit 3](#_Toc139525556)

[squash merge 壓縮被合併的分支commit 4](#_Toc139525557)

[版本控制 4](#_Toc139525558)

[版本線圖 4](#_Toc139525559)

[版本衝突 5](#_Toc139525560)

[vi文字編輯器 : 包含開啟/編輯/儲存/取消/檢視/搜尋…功能 6](#_Toc139525561)

[設定vscode為git的文字編輯器 7](#_Toc139525562)

git基礎指令

建立project 空間 : mkdir

初始化git : cd , git init

基本資訊 : git config --global

user.name

user.email

core.editor <想設定的文字編輯器exe檔案位置>

查詢 git config user.name

建立檔案 : cat > file.txt

類似指令touch, echo, cat

查看狀態 git status

版本追蹤 : git add

加入當前所有檔案 : git add .

建立版本 : git commit -m

查詢版本 : git log

將變更的提交撤銷 : git reset

在add後、commit前 : git reset HEAD <檔案名稱>

在commit後 : git reset HEAD~

* 具有不同程度的撤銷

- 若要撤銷最後一次提交但保留變更在暫存區：

`git reset --soft HEAD~1`

- 若要撤銷最後一次提交並取消暫存區的變更：

`git reset --mixed HEAD~1`

- 若要撤銷最後一次提交並刪除暫存區和工作目錄的變更：

`git reset --hard HEAD~1`

換分支/回復提交/檔案 : git checkout

取出指定提交、檔案且覆蓋當前的現狀

git checkout -- branch name

git checkout -- commit\_hash

git checkout -- file name

添加/刪除branch

添加 git branch <添加的branch名>

指定添加 git branch <添加的branch名> <要指向的commit版本號>

刪除 git branch -d <移除的branch名>

復原 git branch <添加的branch 名 branch hash 版本代碼>

刪除已合併的分支

git branch –merged | agrep -v “(^\\*|master|dev)” xargs git branch -d

列出已merge的branch、

反向搜尋\*當前目錄、master、dev、(不想刪除的分支)

轉換成git 指令並刪除

移除git 版控

刪除版控資訊的資料夾 (.git) 則可解除該資料夾的版控

[fast-forward機制](https://ithelp.ithome.com.tw/articles/10222637)

git預設分支整併使用fast-forward機制，在整併主線上若尚未做出變更，會把已更新的、被整併的支線版本直接指定給目前整併主線，而不保留原始變更紀錄。

若想留存原始紀錄，不想使用fast-forward，於merge時添加

git merge <要整併之分支> –no-ff

將內容推送到遠端的儲存庫

定義遠端路徑

git remote add <路徑別名 , 例如origin> <http url>

將當前編輯內容push上去

git push -u <遠端路徑別名, origin> <當前編輯內容, master>

將遠端儲存庫內容複製到本地

git clone <http url>

更新內容

cd <遠端資料於本地位置>

echo <添加的內容> > <指定文件>

把內容發到遠端

git push

**若想查看最新的遠端資料庫版本，需將遠端資料下載回本地，本地機無法自動及時更新。**

**可使用git pull 將最新資料拉回，**

**其中 git pull = git fetch(更新下載回來) + git merge(整併)**

**若兩條支線的定義不同，建議不使用fast-forward, 保留原始的線圖 (**–no-ff)

merge機制類型

ff -- no-commit先測試再commit

有時程式碼需要先測試再commit，(先merge測試確認沒問題再做commit)

可使用--no-ff --no-commit

則合併過程可以修改程式，修改完再做add與commit

此時狀態為合併中、修改中。

squash merge 壓縮被合併的分支commit

squash merge與一般merge的差異

有時在dev上做些微測試並不需要列入commit紀錄中，因此採用squash





版本控制

若master為主要分支、dev開發中分支，要將dev變動部分merge回master :

1. 切回master

2. git merge <要merge的分支>

若要回復合併前的狀態

git reset --hard ORIG\_HEAD

版本線圖

只顯示第一行的資訊

git log --oneline

以圖的方式顯示

git log --oneline --graph

版本衝突

取消合併

git merge --abort

檢視衝突

vi <衝突的文字文件>、編輯要留下的邏輯、儲存

git add <解衝突完畢的文件>

git commit

vi文字編輯器 : 包含開啟/編輯/儲存/取消/檢視/搜尋…功能

1. 開啟檔案：輸入 `vi 檔案名稱`，例如 `vi README.md`。如果檔案不存在，`vi` 會自動建立一個新檔案。

2. 編輯模式：進入 `vi` 編輯模式，按下 `i` 鍵。現在您可以自由編輯檔案的內容。

3. 儲存變更：在編輯完成後，按下 `Esc` 鍵，然後輸入 `:wq` (write and quit) 儲存並離開編輯器。

4. 取消編輯：在編輯模式下按下 `Esc` 鍵，然後輸入 `:q!` 來取消編輯，並丟棄所有變更。

5. 檢視檔案內容：在命令模式下，輸入 `:sp 檔案名稱` 來分割視窗並檢視其他檔案的內容。同樣地，您可以用 `:q` 來關閉分割視窗。

6. 搜尋文字：在命令模式下，輸入 `/搜尋內容` 來搜尋指定的文字。

Vim cheat sheet <https://vim.rtorr.com/>

設定vscode為git的文字編輯器

對於windows

* 安裝vscode時需要選擇 add to path的選項
* 在git bash介面 git config --global core.editor "code --wait"
* 文字編輯器即被設定為vscode

<https://code.visualstudio.com/docs/sourcecontrol/overview#_vs-code-as-git-editor> 