目錄

[簡介 2](#_Toc491537488)

[檔案的四種狀態 3](#_Toc491537489)

[回復GIT不同區域修改 4](#_Toc491537490)

[git reset 4](#_Toc491537491)

[git rm --cached 4](#_Toc491537492)

[git reset --hard 4](#_Toc491537493)

[分支branch 5](#_Toc491537494)

[建立分支 5](#_Toc491537495)

[查詢目前分支 5](#_Toc491537496)

[開新分支 5](#_Toc491537497)

[開新分支更快指令 5](#_Toc491537498)

[刪除分支 5](#_Toc491537499)

[刪除分支造成部分分支消失 5](#_Toc491537500)

[分支合併 6](#_Toc491537501)

[回復合併 6](#_Toc491537502)

[快速合併 6](#_Toc491537503)

[取消合併(合併遇到衝突時) 6](#_Toc491537504)

[Stash暫存 7](#_Toc491537505)

[參考網站 8](#_Toc491537506)

# 簡介

master(主幹):穩定版本

branch(分支):變動、開發中版本

merge(合併)

主幹開發主要功能，分支開發副功能

git提供視覺版本線圖(commit graph)

clone:遠端儲存庫上的內容複製到本地

Fork:把別人專案的遠端儲存庫複製一分到自己的遠端儲存庫

Push:把目前本地端儲存庫的相關檔案，上傳到遠端儲存庫

Pull:把遠端最新版本下載至自己本機端

Pull不像Clone會下載完整專案各版本內容

Pull Request:主動要求第三方開發者納入自己設計程式

A成員可以透過Pull Request要求B成員，將A成員修改過後程式碼併入B成員開發專案中

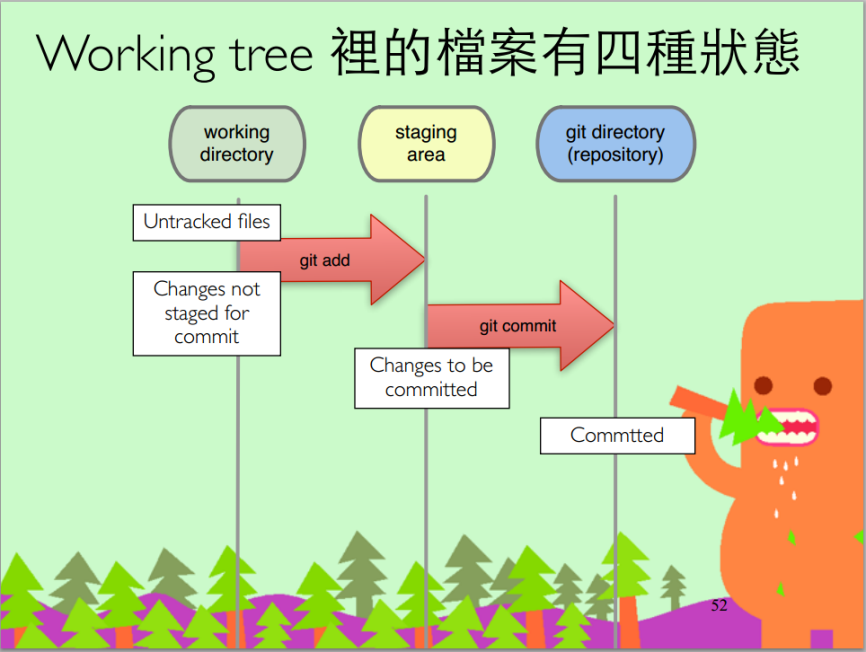
commit message說明此版本做了甚麼修正，能多寫就不要少寫，能詳細不要簡易

git status:查看目前檔案狀態

git log:查看提交詳細log檔(資料太長，按Q離開)

git log --oneline:簡易log檔

# 檔案的四種狀態



未追蹤(untracked files):版本提交後，才又加進來檔案

已更改(changes not staged for commit):已提交後，卻又再次修改，這些檔案會被

丟回工作目錄

等待提交(changes to be commit):git . add之後存放在暫存區(stage)

已提交(committed):執行git commit後，檔案便置於儲存區(repo)

# 回復GIT不同區域修改

## git reset

將等待提交(changes to be commit)改成，未追蹤(untracked files)

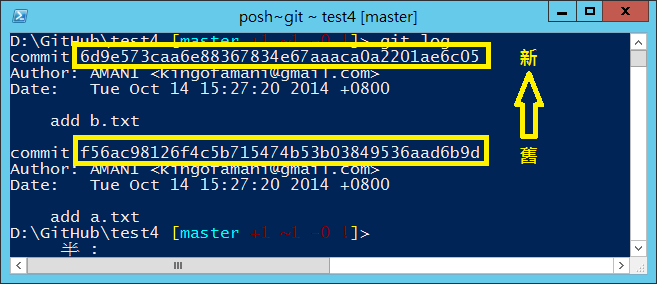
git reset 或是 git rm --cached c.txt

## git rm --cached

要將未追蹤(untracked files)的檔案d.txt移除(不用加git)

rm d.txt

git status



## git reset --hard

回去先前版本git log後版本由新到舊分別是上到下，輸入後可以看到一堆SHA1碼，可以想成是每一個版本的ID，打前六碼就好，除非專案太大，才要增加碼數(範例為f56ac9)

git reset --hard f56ac9

--hard表示強制轉換成f56ac9版本檔案

# 分支branch

## 建立分支

.git branch 名稱

## 查詢目前分支

git branch

## 開新分支

git branch 名稱

若在master上開新分支，例如開fix分支

git branch fix

切換到新支上

git checkout fix

在下一次查看分支指令

git branch

## 開新分支更快指令

git checkout -b 名稱

以一個fix分支來說

【git checkout -b fix】=【git branch fix】+【git checkout fix】

## 刪除分支

branch –d 名稱

刪除分支，記得需跳出此分支

git checkout master

git branch -d fix

註:若此分支未和其他分支合併，無法直接誤刪，必須透過branch -D來強制  
刪除

## 刪除分支造成部分分支消失

git --online

git reflog

git checkout [分支SHA1值]

git checkout –b fix

## 分支合併

將fix分支合併到master分支上

切換到master分支上

git checkout master

將fix分支合併

git merge fix

註:合併後，代表fix分支完成階段任務，通常會刪除該分支

## 回復合併

git checkout master

git log --online

git reset --hard[commit ID]

註:更簡單方法，直接回到上一個版本

git reset --hard ORIG\_HEAD

若想返回前一個版本，及保留目前檔案可用

--soft

## 快速合併

git checkout master

git merge fix

註:分支不會長耳朵，不用多交commit

## 取消合併(合併遇到衝突時)

git reset --hard HEAD

## Stash暫存

git stash -u

or

git stash save -u “註解”

註:可將目前WD(Work Direction)與SA(Stage Area)的檔案暫存，儲存完畢後可切換分支

回到暫存的分支，檢查站存是否還在

git stash list

暫存叫回來

git stash pop

註:刪除暫存指令

git stash clear

# 參考網站

[1]<https://kingofamani.gitbooks.io/git-teach/content/chapter1\_install\_git/>