**Git handout**

**參考git官方 https://git-scm.com/book/zh-tw/v2**

**設定識別資料：**

git config --global user.name “username” //設定使用者名稱

git config –global user.email “useremail” //設定使用者email

若你想指定不同的名字或電子郵件給特定的專案，只需要在該專案目錄內執行此命令，並確定未加上 --global—global

Git config –list //檢查設定擋

取得git repository：

方法一：在現有的資料夾中初始化repository

git init

方法二：克隆現有的repository

git clone [url]

ex:

git clone <https://github.com/libgit2/libgit2>

這指令將會建立名為「libgit2」的資料夾，並在這個資料夾下初始化一個 若你進入新建立的 若你想要將倉儲克隆到「libgit2」以外名字的資料夾，只需要再多指定一個參數即可

git clone https://github.com/libgit2/libgit2 mylibgit

這個命令做的事與上一個命令大致相同，只不過在本地創建的倉庫名字變為mylibgit

**Git基礎：**



圖為檔案狀態的生命週期

記錄變更到版本庫中：

git status

//用來偵測哪些檔案處在什麼樣得狀態下的主要工具

git add “filename” //追蹤一個新檔案

git add \* //追蹤所有檔案

git commit -m “message”//提交你的檔案

git commit -a //可略預存區,直接提交

git rm “filename”//移除檔案

要從Git 中刪除一個檔案，你需要將它從已追蹤檔案中移除（更準確地說，是從預存區中移除），然後再提交；git rm

git rm –cashed “filename”

保留工作目錄的檔案，但將它從預存區中移除； 換句話說，你或許想保留在磁碟機上的檔案但不希望Git 再繼續追蹤它

git mv file\_from file\_to //移動檔案

如果你在Git 中重新命名一個檔案，並不會有任何Git 後設資料記錄這個動作以辨別你曾經重新命名過檔案； 然而Git 可以在檔案移動後很聰明地將它們找出來

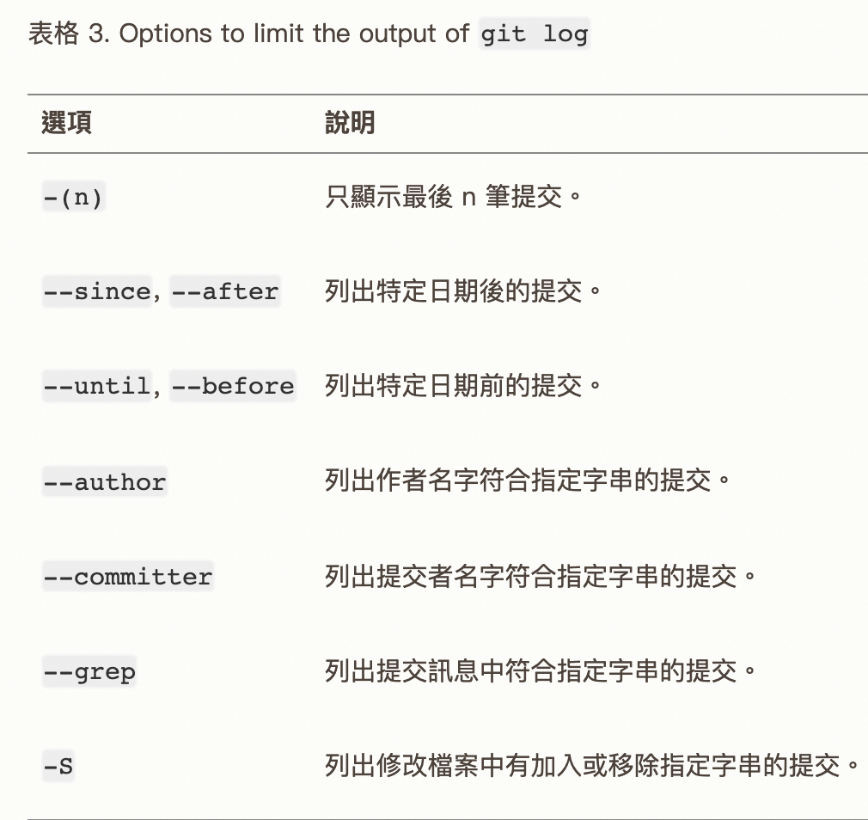
git log //檢視提交的歷史紀錄 (按q可離開log紀錄檔)

git log –decorate //可以看到分支指向何處

git log -p -2

最有用的選項之一是 -p，用來顯示每筆提交所做的修改內容； 你還可以加上 -2 選項，限制只輸出最後兩筆提交內容。

git log --stat //檢視每筆提交簡略的統計資訊



還有很多關於log的參數這裡就不多加贅述

git commit –amend

這個命令會再次把預存區（staging area）拿來提交， 如果自從上次提交以來你沒有做過任何檔案修改（例如：在上一次提交後，馬上執行此命令），那麼整個快照看起來會與上次提交的一模一樣，唯一有可能更動的是提交訊息。

git reset HEAD filename //將已預訂的檔案移出預存區

git remote -v //顯示你的遠端

git remote show [remote-name] //檢視更多遠端資訊

git remote add <簡稱> <url> //新增遠端版本庫

git fetch [remote-name] //從你的遠端專案中取得資料

git push [remote-name][branch-name] //推送資料到遠端

只有在你對克隆來源的伺服器有寫入權限，並且在這個當下還沒有其它人推送過，這個命令才會成功

git remote rename [origin-name] [new-name] //重新命名遠端

git remote rm [remote name] //移除一個遠端

**Git branch：**

為了具體說明，讓我們假設你有一個目錄包含了三個檔案，你預存（stage）並提交了它們； 檔案預存操作會對每一個檔案內容（譯註：請注意，只有檔案「內容」）計算雜湊值（即當使用 git commit你的Git 版本庫現在有五個物件：三個blob 物件用來儲存檔案內容、一個樹物件用來列出目錄的內容並紀錄各個檔案所對應的blob 物件、一個提交用來記錄根目錄的樹物件和其他提交資訊。

git branch [branch-name] //建立一個新分支

git checkout [branch-name] //在分支之間切換

git branch -b [branch-name]

//相當於執行git branch [branch-name], git checkout [branch-name]兩個指令

git merge [branch-name] //合併分支





執行merge hotfix指令的示意圖(HEAD指向master)

git branch -d [branch-name] //刪除branch





執行merge iss53的示意圖(HEAD指向master)