

Version Control

*How to use Git and GitHub (**Udacity Supplement**)*

Alan Peng

bash terminal 環境設定

- 下載以下兩個檔案並儲存於 *\$home* 目錄下
- <https://raw.githubusercontent.com/git/git/master/contrib/completion/git-completion.bash>
- <https://raw.githubusercontent.com/git/git/master/contrib/completion/git-completion.bash>

bash terminal 環境設定

- 從這裡下載bash_profile_course檔，並更名為***.bash_profile***，儲存於\$home目錄下
- 在.bash_profile裡設定alias如下：
- ***alias subl="/Applications/Sublime\ Text\ 2.app/Contents/SharedSupport/bin/subl"***

Git 環境設定

- *git config --global core.editor "/Applications/Sublime Text 2.app/Contents/SharedSupport/bin/subl' -n -w"*
- *git config --global push.default upstream*
- *git config --global merge.conflictstyle diff3*
- Restart terminal

git log

- 查看commit歷史紀錄

```
hcpeng (master) git_test $ git log
commit ec5a12ccc76bf8ff38e8377b398b74c4bee0b143
Author: Alan Peng <hcpeng1975@gmail.com>
Date:   Sat Mar 26 19:42:54 2016 +0800
```

add file7

```
commit d6a76422b7302d806b5f5eb033a5889387f96521
Author: Alan Peng <hcpeng1975@gmail.com>
Date:   Sat Mar 26 18:32:22 2016 +0800
```

Revert "file1 changed on branch issue2"

```
This reverts commit
0a5057ff014b1ef8b2d6a5ec66205ea70703b2f5.
```


git diff

- 比較兩個commit間的差異

```
hcxpeng (master) git_test $ git diff
ec5a12ccc76bf8ff38e8377b398b74c4bee0b143
d6a76422b7302d806b5f5eb033a5889387f96521
diff --git a/file7 b/file7
deleted file mode 100644
index bb1f06b..00000000
--- a/file7
+++ /dev/null
@@ -1,0 @@
-This is file 7
```


git checkout

- 幾個常混淆的用法：
- `git checkout filename` 回到filename上一個版本的狀態
- `git checkout commit` “detached HEAD” 實驗狀態，這個狀態下所做的任何變動在回到master後都會消失
- `git checkout branch` 切換已存在的分支
- `git checkout -b branch` 創建一個新的分支並切換到該分支上

git init

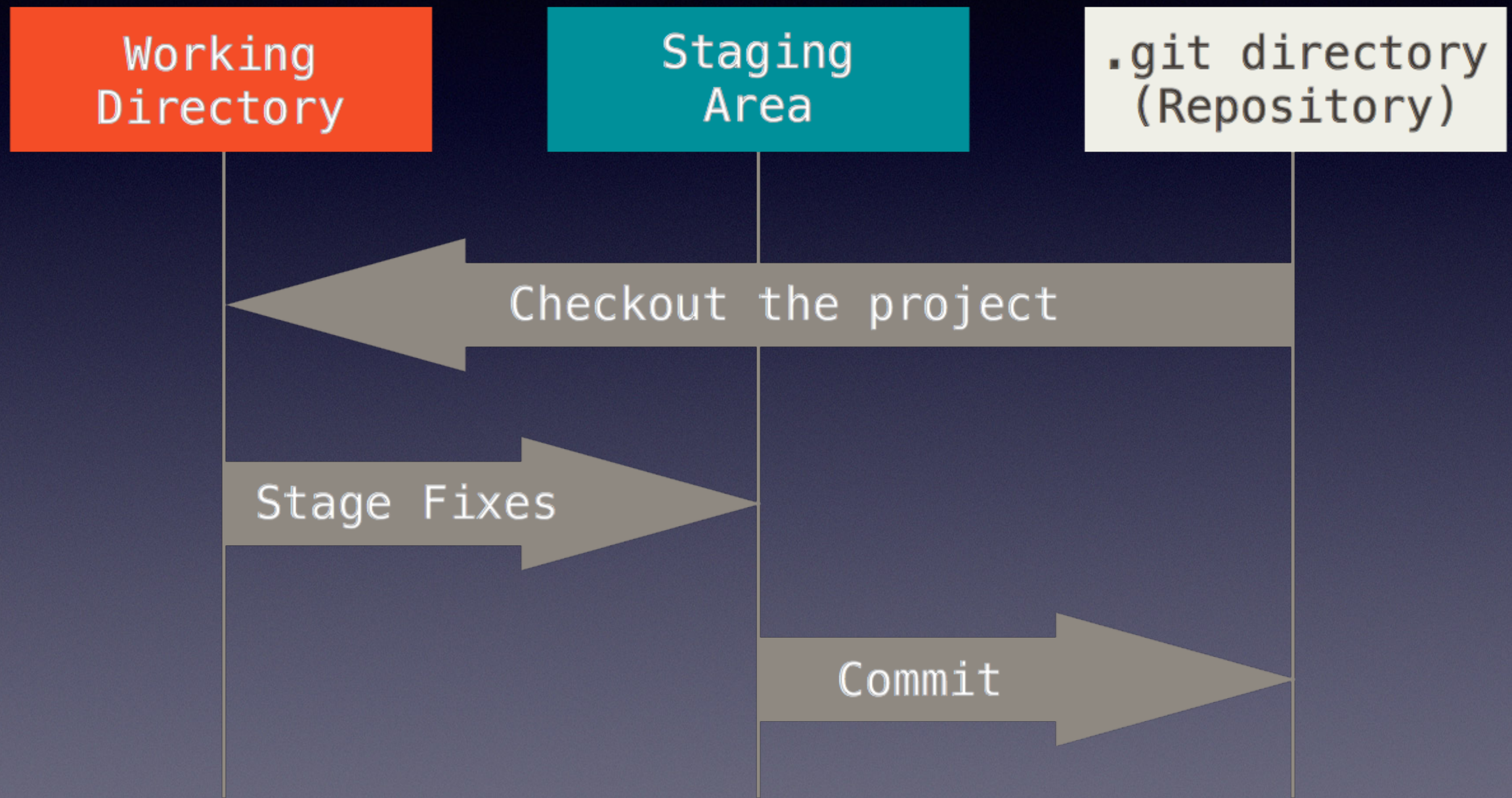
- 每個專案開始前，創建新的 *git repository* (簡稱 *repo*)

```
hcpeng git_test2 $ git init
```

```
Initialized empty Git repository in /Users/hcpeng/Intro to  
Git/git_test2/.git/
```

```
.git
├── HEAD
├── config
├── description
├── info
│   └── exclude
├── objects
│   ├── info
│   └── pack
└── refs
    ├── heads
    └── tags
```


git的三個states



git的三個states

- *Working Directory* 就是我們放專案的目錄(工作目錄)
- *Staging Area* 我們把希望讓git追蹤的檔案放在這裡
- *Git repository* Git的檔案目錄，commit的歷史紀錄會放在這裡

git add/rm

- `git add/rm filename` ◦ 把改過的檔案加入/移除 staging area

```
hcpeng (master #) git_test2 $ git add file1
hcpeng (master +) git_test2 $ git status
On branch master
```

Initial commit

Changes to be committed:
(use "git rm --cached <file>..." to unstage)

```
new file:   file1
```


git status

- 顯示目前工作目錄狀態，這裡顯示新增 *file1* 一個檔案

```
hcpeng (master +) git_test2 $ git status  
On branch master
```

```
Initial commit
```

```
Changes to be committed:  
  (use "git rm --cached <file>..." to unstage)
```

```
new file:   file1
```


git commit

- 紀錄repo的變動 *-m* 表示加上commit訊息

```
hcpeng (master +) git_test2 $ git commit -m  
"add file1"  
[master (root-commit) 78a3c51] add file1  
1 file changed, 1 insertion(+)  
create mode 100644 file1
```


git branch

- 顯示、建立或刪除分支
- *git branch, git branch feature or git branch -d feature*

```
hcpeng (master) git_test2 $ git branch feature
```

```
hcpeng (master) git_test2 $ git branch
```

```
feature
```

```
* master
```

```
hcpeng (master) git_test2 $ git branch -d feature
```

```
Deleted branch feature (was 78a3c51).
```

```
hcpeng (master) git_test2 $ git branch
```

```
* master
```


git remote add/rm

- 加入遠端的`repo`，例如GitHub上的專案

```
git remote add origin https://github.com/user/  
repo.git  
# Set a new remote
```

- 查看remote設定

```
git remote -v  
# Verify new remote  
origin https://github.com/user/repo.git (fetch)  
origin https://github.com/user/repo.git (push)
```


git clone/fetch/merge

- *git clone*是copy完整的repo；*git fetch*是只下載更新的部分；*git merge*是合併其他的分支

```
git clone https://github.com/USERNAME/REPOSITORY.git  
# Clones a repository to your computer
```

```
git fetch remotename (e.g. origin)  
# Fetches updates made to a remote repository
```

```
git merge remotename/branchname (e.g. origin master)  
# Merges updates made online with your local work
```


Merge Conflict

- 檔案的同一行被其他人修改過
- 無法進行自動合併，因為git不知道該保留哪個修改版本
- 找出該衝突檔案並修改保留其中一個版本
- 衝突解決後再進行合併

Merge Conflict

- *file1*的同一行在master及feature1被修改過
- 自動合併失敗

```
hcpeng (master) git_test2 $ git merge feature1
Auto-merging file1
CONFLICT (content): Merge conflict in file1
Automatic merge failed; fix conflicts and then
commit the result.
```


Merge Conflict

- This is file1
- <<<<<<< HEAD
- modified on master branch
- ||||| merged common ancestors
- =====
- *modified on feature1 branch (與master衝突的地方)*
- >>>>>>> feature1

Merge Conflict

- 保留 *feature1* 的修改，重新與master合併

This is file1
modified on feature1 branch

```
commit 0f47de7201105e90f257259ea82ef5d4278afaf5
```

```
Merge: b1ad477 931406e
```

```
Author: Alan Peng <hcpeng1975@gmail.com>
```

```
Date: Wed Mar 30 10:46:07 2016 +0800
```

Merge branch 'feature1'

```
hcpeng (master) git_test2 $ git tree (git log --graph --decorate --  
pretty=oneline --abbrev-commit)
```

```
* 0f47de7 (HEAD -> master) Merge branch 'feature1'  
|\   
| * 931406e (feature1) file1 modified on feature1 branch  
| * | b1ad477 file1 modified on master branch  
|/   
* 78a3c51 add file1
```


Merge Conflict

- 如果feature1從master分支出去後，所有檔案僅在feature1被修改
- 那麼我們可以說所有的變動是基於原master的code base之上
- 這時如果合併master & feature1就是一個fast-forward merge (僅往前移動master的HEAD到最新的commit); 合併線圖是一直線

```
hcpeng (master) git_test2 $ git merge feature1
```

```
Updating 78a3c51..931406e
```

```
Fast-forward
```

```
file1 | 1 +
```

```
1 file changed, 1 insertion(+)
```

```
hcpeng (master) git_test2 $ git tree
```

```
* 931406e (HEAD -> master, feature1) file1 modified on feature1  
branch
```

```
* 78a3c51 add file1
```


git push/pull

- *git push*是上傳local的repo到遠端的機器上；*git pull = git fetch + git merge*

```
git push remotename branchname  
e.g. git push origin master
```

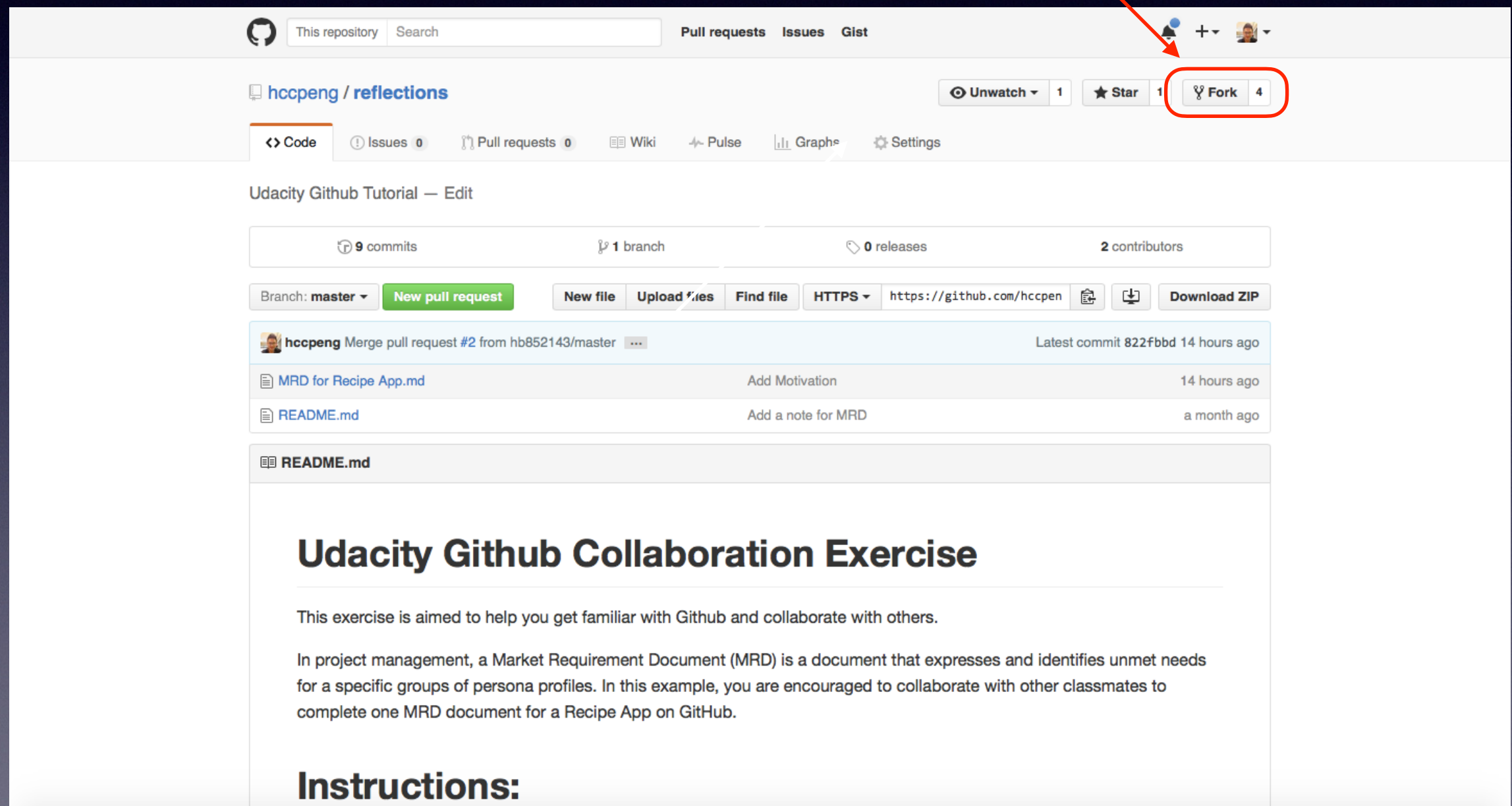
```
git pull remotename branchname  
# Grabs online updates and merges them with your local work  
e.g. git pull origin master
```


GitHub “fork”

- fork也是一種copy。在GitHub上fork是用來在其他人的專案的`code base`上做二次開發，尤其是一些開源的專案。或是希望能直接參與其他人的專案來提案`bug fix`或新功能。
- fork專案到自己repo下所做的任何變動都不會對原專案的code base產生任何影響。

GitHub “fork”

fork這個專案



The screenshot shows the GitHub interface for the repository 'hccpeng / reflections'. At the top, there's a navigation bar with 'Pull requests', 'Issues', and 'Gist'. Below this, the repository name 'hccpeng / reflections' is displayed. To the right of the repository name, there are buttons for 'Unwatch', 'Star' (1), and 'Fork' (4). A red arrow points to the 'Fork' button. Below the repository name, there are tabs for 'Code', 'Issues' (0), 'Pull requests' (0), 'Wiki', 'Pulse', 'Graphs', and 'Settings'. The main content area shows the repository details: 'Udacity Github Tutorial — Edit', '9 commits', '1 branch', '0 releases', and '2 contributors'. Below this, there are buttons for 'New pull request', 'New file', 'Upload files', 'Find file', 'HTTPS', and 'Download ZIP'. A commit history table is shown with columns for commit message, author, and time. The commit history includes a merge pull request #2 from hb852143/master and two other commits: 'MRD for Recipe App.md' and 'README.md'. The 'README.md' file is selected, and its content is displayed below. The content includes the title 'Udacity Github Collaboration Exercise' and a description of the exercise. The 'Instructions:' section is partially visible at the bottom.

hccpeng / reflections

Unwatch 1 Star 1 Fork 4

Code Issues 0 Pull requests 0 Wiki Pulse Graphs Settings

Udacity Github Tutorial — Edit

9 commits 1 branch 0 releases 2 contributors

Branch: master New pull request New file Upload files Find file HTTPS https://github.com/hccpen Download ZIP

Commit Message	Author	Time
hccpeng Merge pull request #2 from hb852143/master	hccpeng	Latest commit 822fbdd 14 hours ago
MRD for Recipe App.md	hccpeng	14 hours ago
README.md	hccpeng	a month ago

README.md

Udacity Github Collaboration Exercise

This exercise is aimed to help you get familiar with Github and collaborate with others.

In project management, a Market Requirement Document (MRD) is a document that expresses and identifies unmet needs for a specific groups of persona profiles. In this example, you are encouraged to collaborate with other classmates to complete one MRD document for a Recipe App on GitHub.

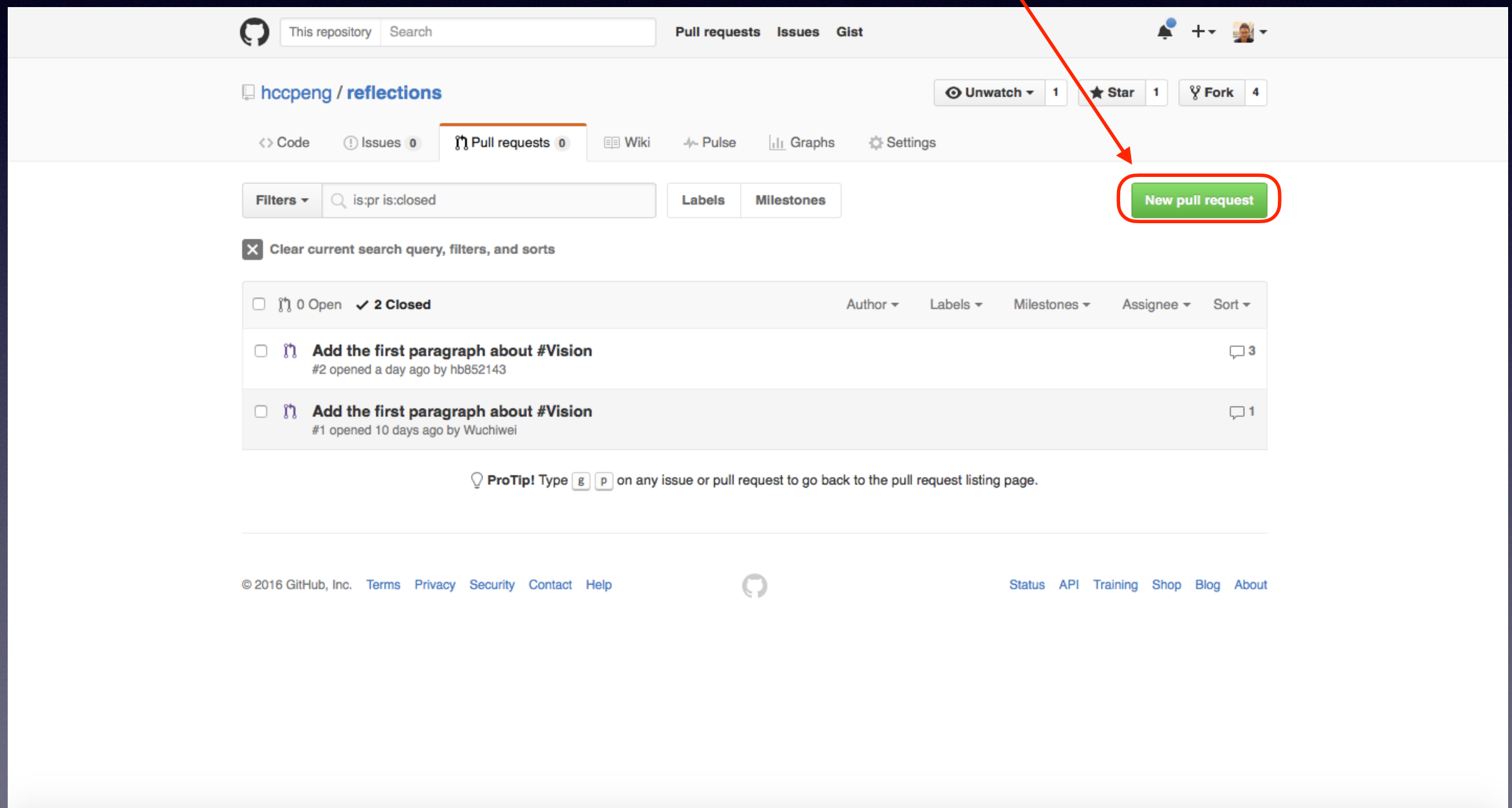
Instructions:

GitHub “pull request”

- *Pull request*是一個合併請求。在fork他人專案並在自己的機器上做修改後，需要提出請求讓原作者把自己修改的部分合併回原專案。
- 原專案作者就如同專案管理員，有權控管哪些修改可以併入到該專案裡。

GitHub “pull request”

提交新的pull request



GitHub “pull request”

新的pull request在經原專案作者同意後合併到原專案裡

The screenshot displays a GitHub pull request interface. At the top, navigation tabs include Code, Issues (0), Pull requests (0), Wiki, Pulse, Graphs, and Settings. The main title is "Add the first paragraph about #Vision #1" with an "Edit" button. Below the title, a status bar indicates "Merged" by hccpeng, merging 1 commit into hccpeng:master from Wuchiwei:Lucas.Wu, 9 days ago. A summary bar shows 1 conversation, 1 commit, and 1 file changed, with a net change of +3 lines and -1 line. The comment history shows Wuchiwei's comment from 10 days ago and hccpeng's response from 9 days ago, stating the pull request will be merged. A merge event is also shown. A sidebar on the right contains metadata: Labels (None yet), Milestone (No milestone), Assignee (No one—assign yourself), and Notifications (Unsubscribe button). At the bottom, there is a "Write" comment box with a "Preview" tab and a rich text editor toolbar.

中文教材參考

- <https://ihower.tw/git/> (*Git* 版本控制系統)
- <https://git-scm.com/book/zh-tw/v1/開始-關於版本控制> (*Pro Git* 中文版)
- <https://backlogtool.com/git-guide/tw/> (連猴子都能懂的*git*入門指南)