

人生不能重來

但 Git 可以

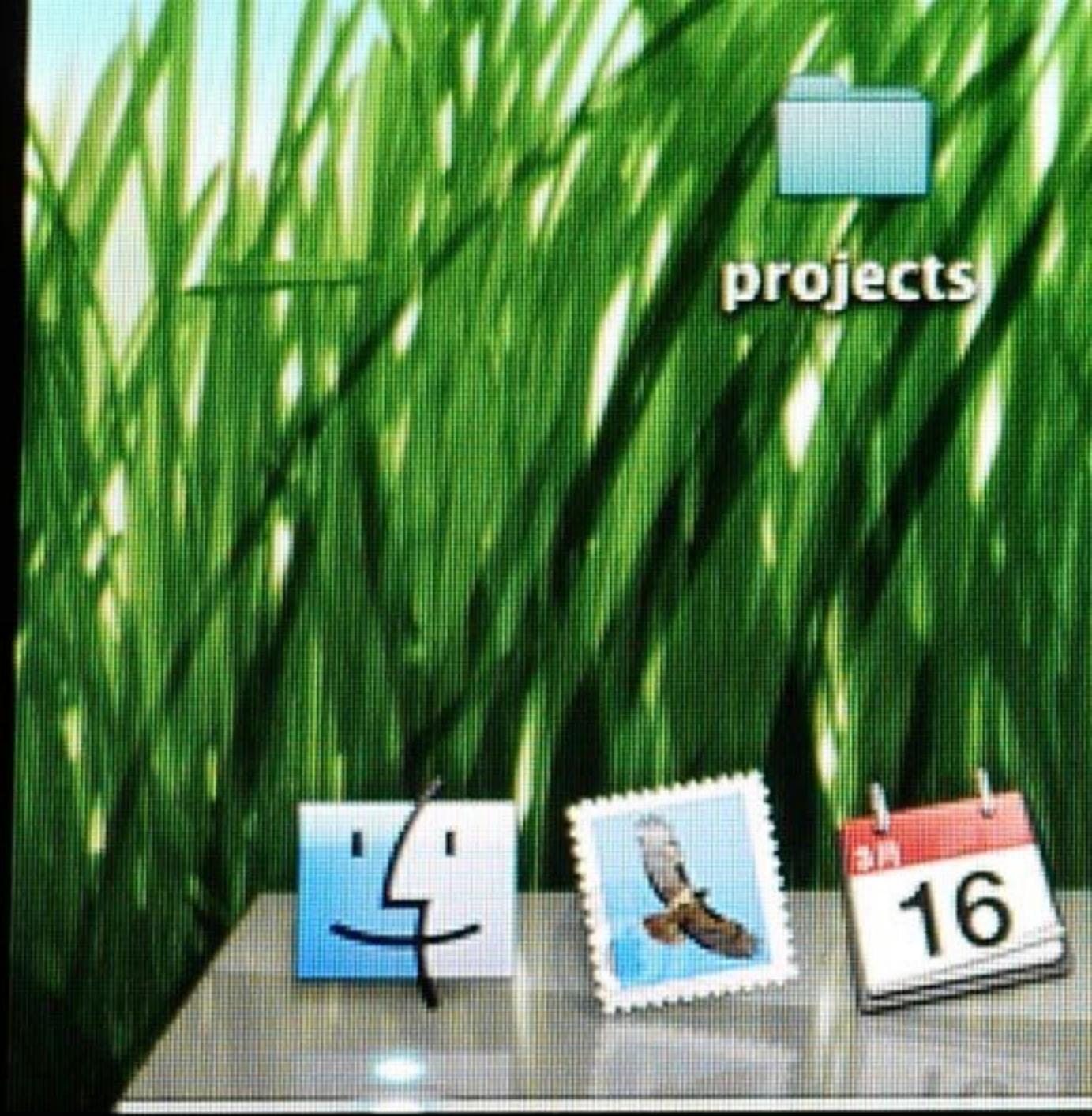
高見龍

自 我 介 紹

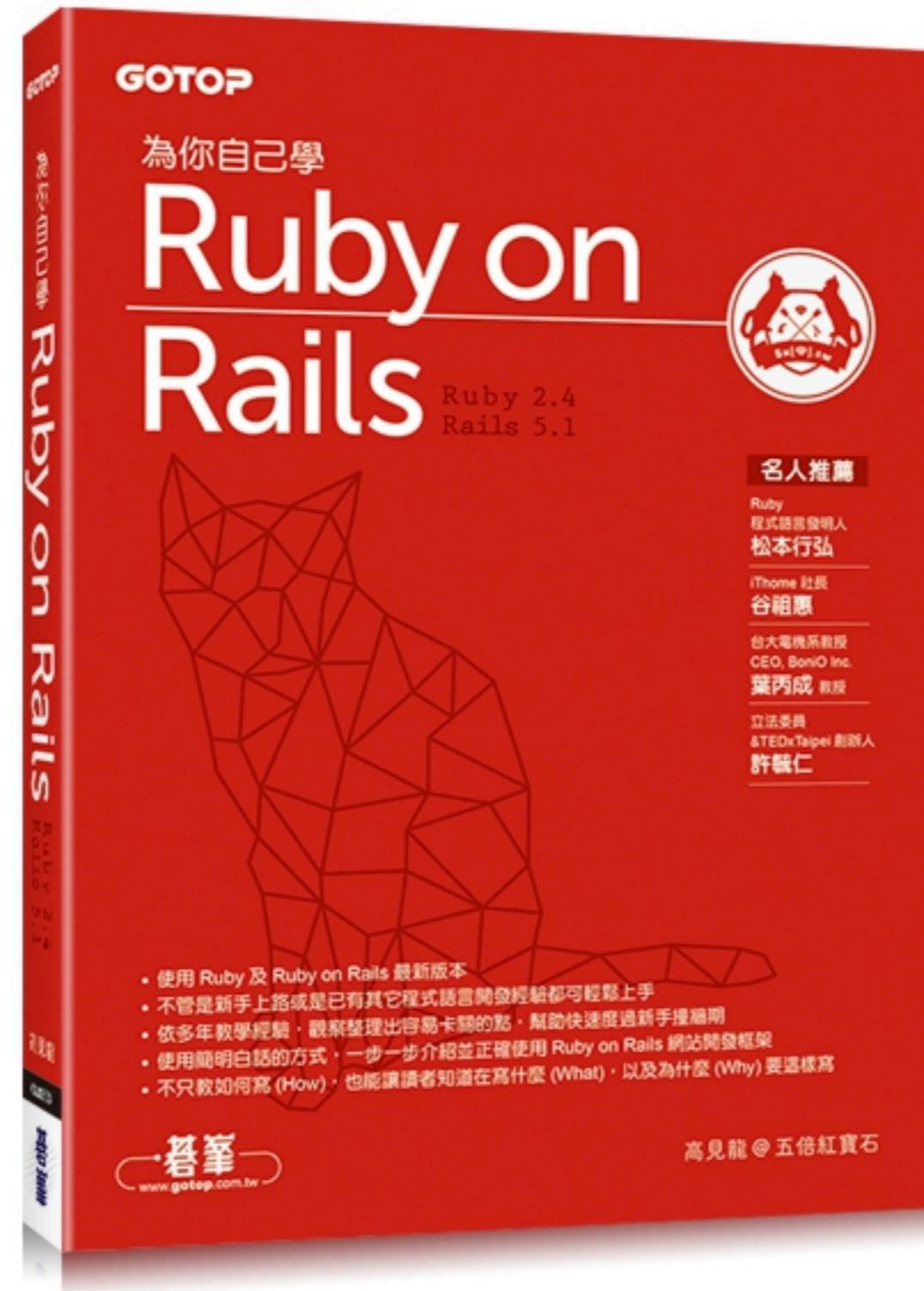
高見龍 @eddiekao

a.k.a Eddie

- 愛現！喜歡冷門的玩具
- Ruby/Rails/iOS app 開發者、講師
- Ruby 技術推廣、教育、諮詢
- 台灣、日本等國內外 Ruby 技術研討會講者
- 目前於五倍紅寶石擔任紅寶石鑑定商職務
- 部落格：<https://kaochenlong.com>

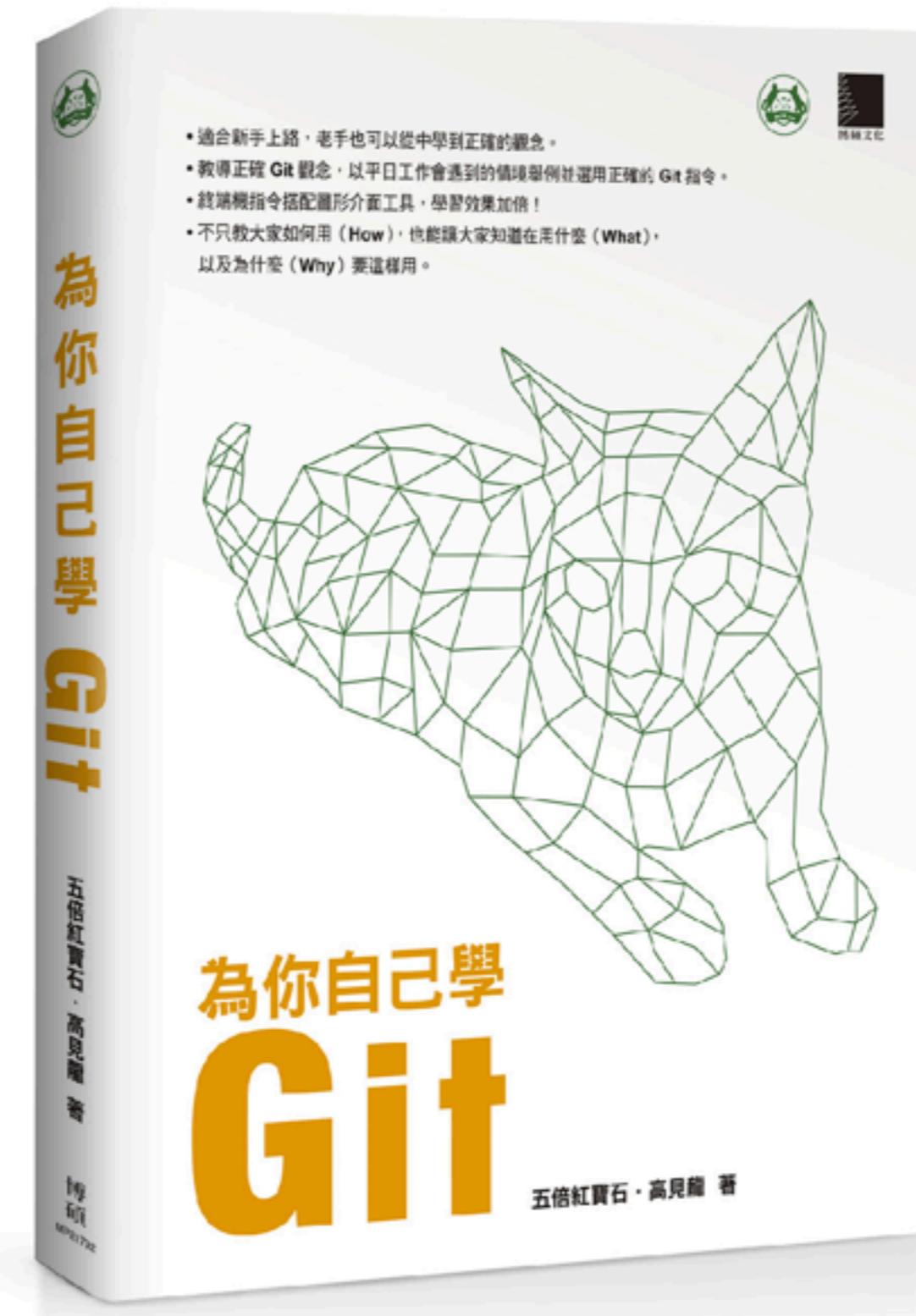


發售中！



<https://railsbook.tw/>

發售中！



<https://gitbook.tw/>



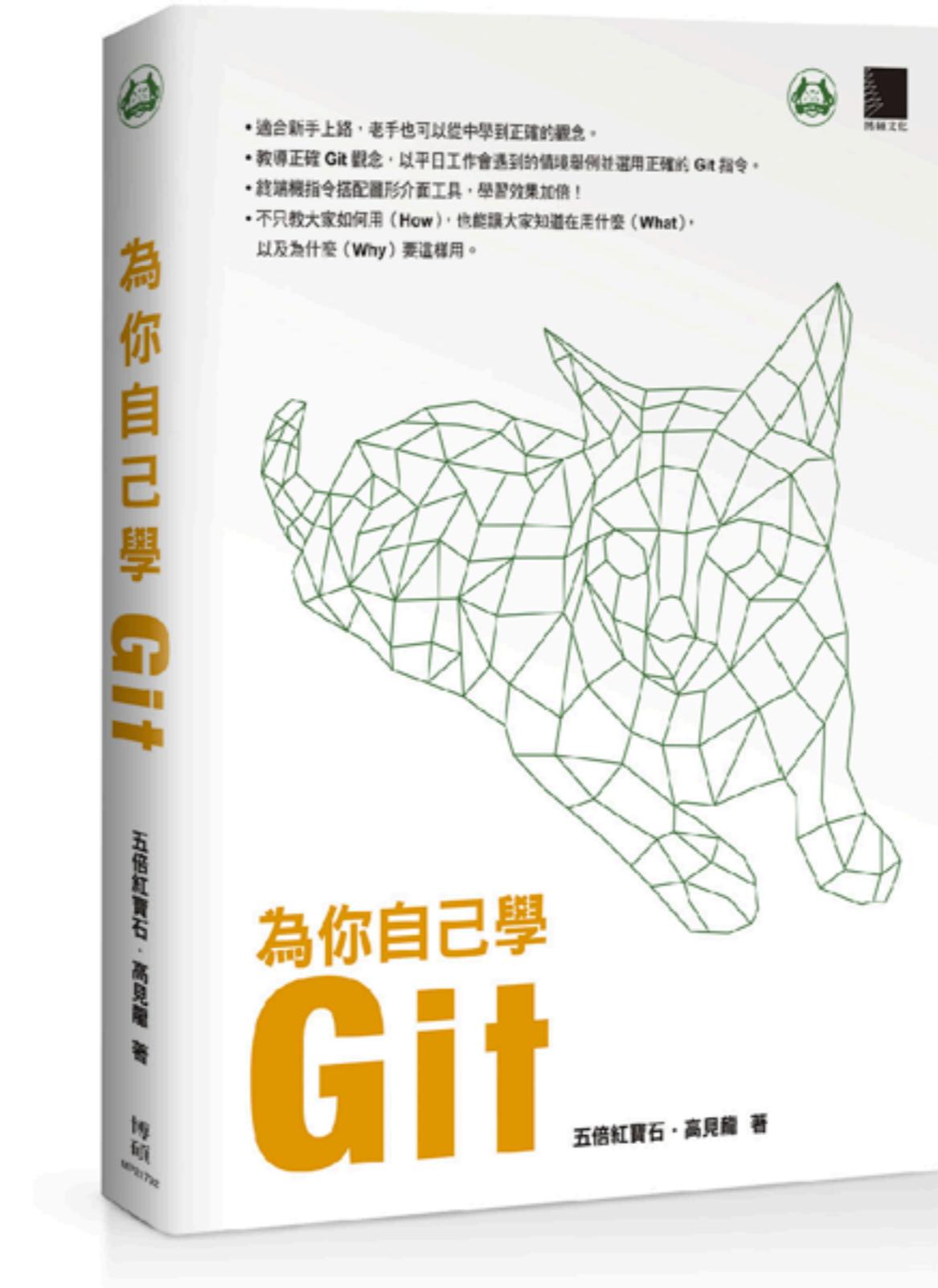
Line 群組

開始之前...

問題

有沒有第一次聽過

Git 這三個字母的人..?



<https://gitbook.tw/>

問題

工作上有在用 Git, 但只有把 Git 當 FTP 在用？

THIS IS GIT. IT TRACKS COLLABORATIVE WORK
ON PROJECTS THROUGH A BEAUTIFUL
DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZIZE THESE SHELL
COMMANDS AND TYPE THEM TO SYNC UP.
IF YOU GET ERRORS, SAVE YOUR WORK
ELSEWHERE, DELETE THE PROJECT,
AND DOWNLOAD A FRESH COPY.



「這就是 Git，它使用一種很漂亮的分散式圖型理論模型的技術來追蹤我們專案的內容」

「酷！這要怎麼用？」

「不知道，大概就是記一些終端機指令 指令敲一敲檔案就會同步了。如果發生問題，先把檔案先備份一份到別的地方，把專案砍掉，然後重新下載一份全新的專案就行了。」

本文開始！

你每天的工作，

可能都會建立或修改很多的檔案...

編輯、存檔、編輯、存檔.. × N

問題

你怎麼做檔案備份？

最簡便也最無敵的

複製、貼上

Ctrl-C + Ctrl-V

online

Git advanced v1.key

Git advanced-v1-out.key

Git and Github v2.key

Git and Github v3.key

Git and Github v4.key

Git and Github v5.key

Git and Github v6.key

Git and Github v7.key

Git and Github v8.key

Git and Github v9.key

Git and Github v10.key

Git and Github v11.key

Git and Github v12.key

Git and Github.key

Git v3.key

Git v4.key

其實這份簡
報就是 v12

- ▼  resumes
 - >  resume-2016-02-08
 - >  resume-2016-02-10
 - >  resume-2016-05-08
 - >  resume-2016-08-22
 - ▼  resume-2016-11-28
 -  eddie.md
 -  john.md
 -  kao.md
 -  mary.md
 -  sherly.md
 -  tracy.md
 - >  resume-bak
 - >  resume-for-5xruby



怎麼少了一個人？

怕本機硬碟壞掉？

備份到 NAS 或雲端硬碟就好啦

這有什麼問題嗎？

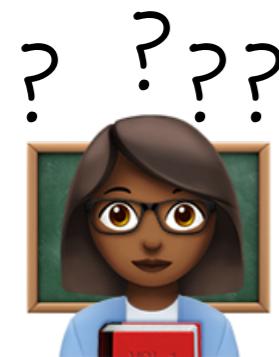
版本控制

Version Control



那個...你有聽過 Git 嗎？那是什麼？

啊就一種分散式的版本控制系統啊！



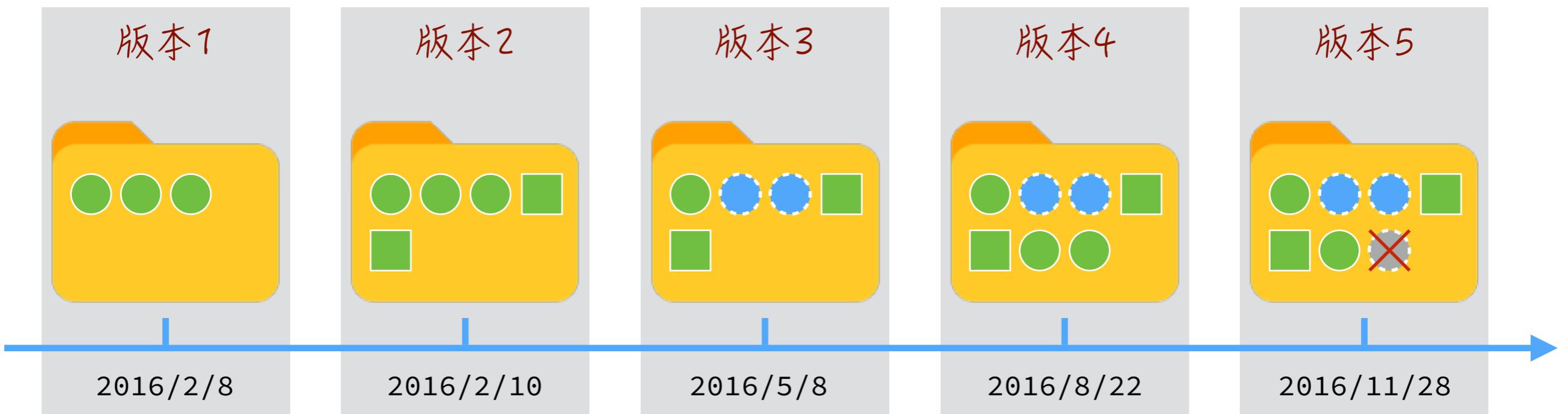
分散？版本？

問題

什麼版本？要控制什麼？

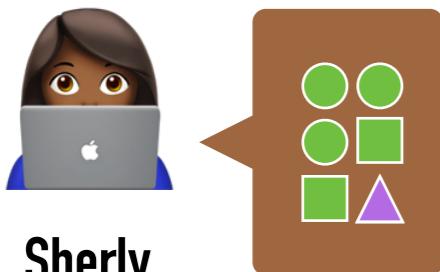
問題

所以，什麼是「版本」？



問題

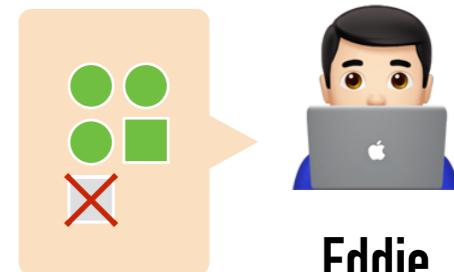
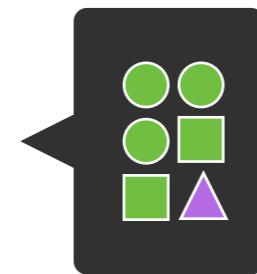
分散式？



Sherly



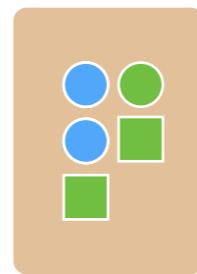
Octocat



Eddie



Emily



Picasso

備份

(就像玩遊戲的時候會儲存遊戲進度一樣)

歷史紀錄及證據

(出事的時候你會知道是從什麼時候開始就有問題,
以及知道該找誰來罵)

Concurrent Versions System (CVS)

since 1990

Subversion (SVN)

問題是...

如果沒有網路，或是 SVN 的伺服器故障的時候便無法使用

沒有網路就不能寫 code ?

GitHub 壞掉的時候大家都會開玩笑的說可以下班了



We're having a really bad day.

The Unicorns have taken over. We're doing our best to get them under control
and get GitHub back up and running.

Git

什麼是 Git?

Git 是一種版本控制系統

Linus Torvalds

for managing Linux kernel
source code in 2005

photo by [Krd](#)



分散式

遠端、本地操作皆可

而且大部份的操作都是本地端的

問題

Git 我知道，現在工程師們都說那個 GitHub 什麼的...

Git != GitHub

優點上

開源、免費

速度快、 檔案小

同時支援本地及遠端操作

容易與其它人共同協作

前提是「其它人」也要知道怎麼用 Git 就是了...

Git 是一種易學難精的工具

80 / 20 法則

20% 的 git 指令就足以應付平日 80% 的工作

今天就是介紹那個 20%

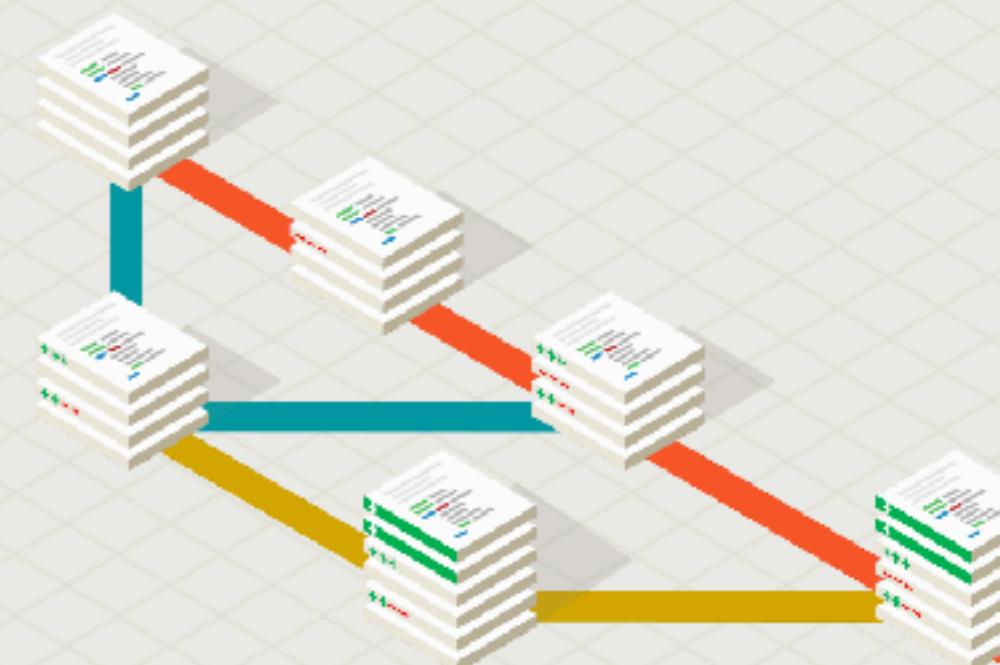
安裝 Git

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.



Learn Git in your browser for free with [Try Git](#).



About

The advantages of Git compared to other source control systems.



Documentation

Command reference pages, Pro Git book content, videos and other material.

Downloads

GUI clients and binary releases for all major platforms.



Community

Get involved! Bug reporting, mailing list, chat, development and more.

Latest source Release
2.12.2

[Release Notes \(2017-03-24\)](#)

[Downloads for Mac](#)



<https://git-scm.com/>

Mac:

```
$ brew install git
```

Ubuntu or some linux OS:

```
$ sudo apt-get install git-core
```

or

```
$ sudo apt-get install git
```

練習：

請在你的電腦上安裝 Git

開始使用 Git

三几
設定

檢視目前設定

```
$ git config --list
```

設定 username 及 email

```
$ git config --global user.name "eddie"  
$ git config --global user.email "eddie@5xruby.tw"
```

練習：

設定你的使用者名稱及 email.

不要害怕端終機或指令模式
仔細看它的輸出結果

新增、初始 Repository

建立目錄

```
$ mkdir git-practice
```

初始化

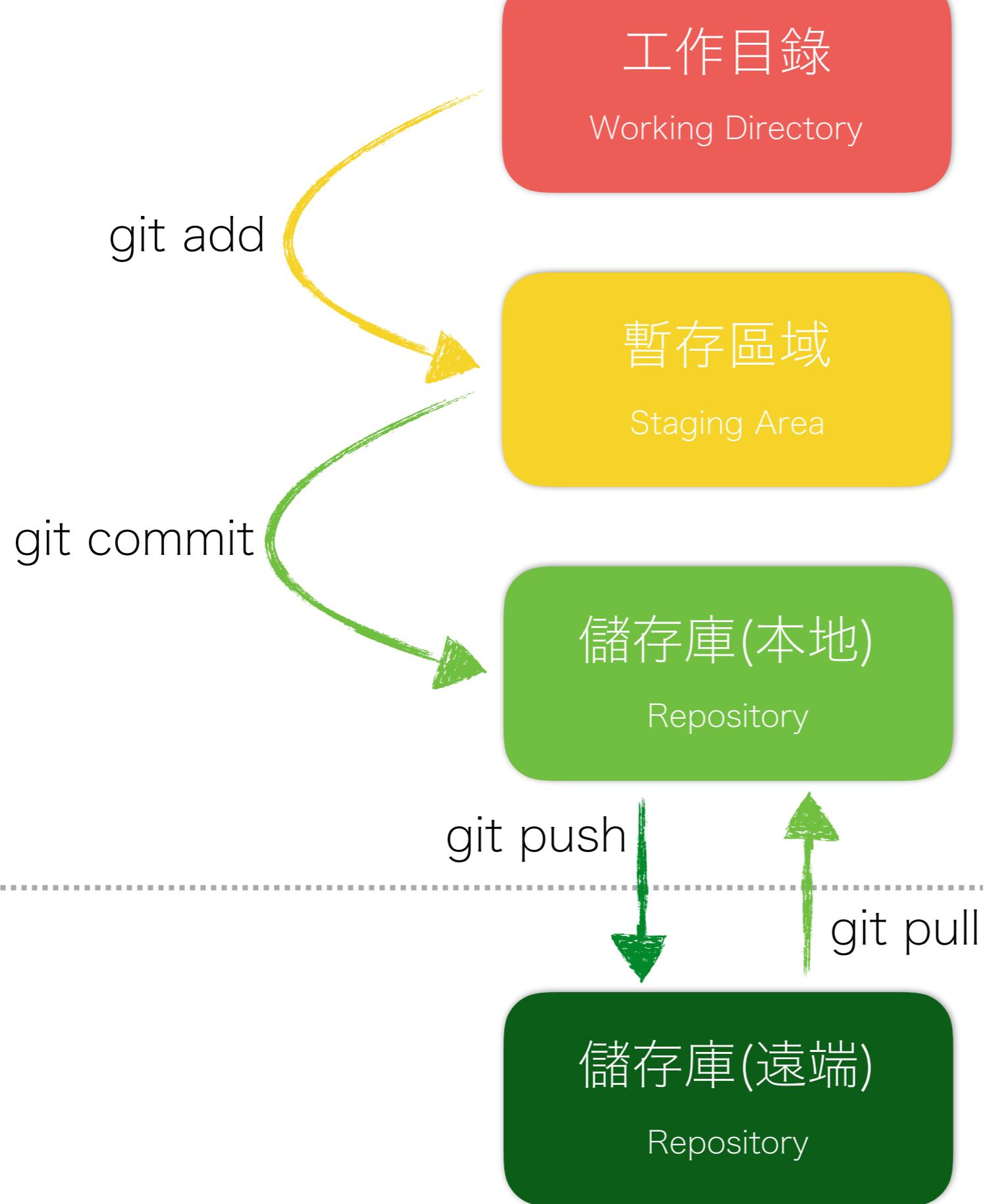
```
$ cd git-practice
$ git init
Initialized empty Git repository in /private/tmp/git-practice/.git/
```

練習：

建立一個新的目錄，並且 Git 進行
初始化。

Working, Staging, and Repository

工作目錄, 暫存區域, 以及 儲存庫



狀況

我剛新增、修改了一些檔案，要怎麼用 Git 來管理它們？

新增檔案 index.html

```
$ touch index.html
```

查看狀態

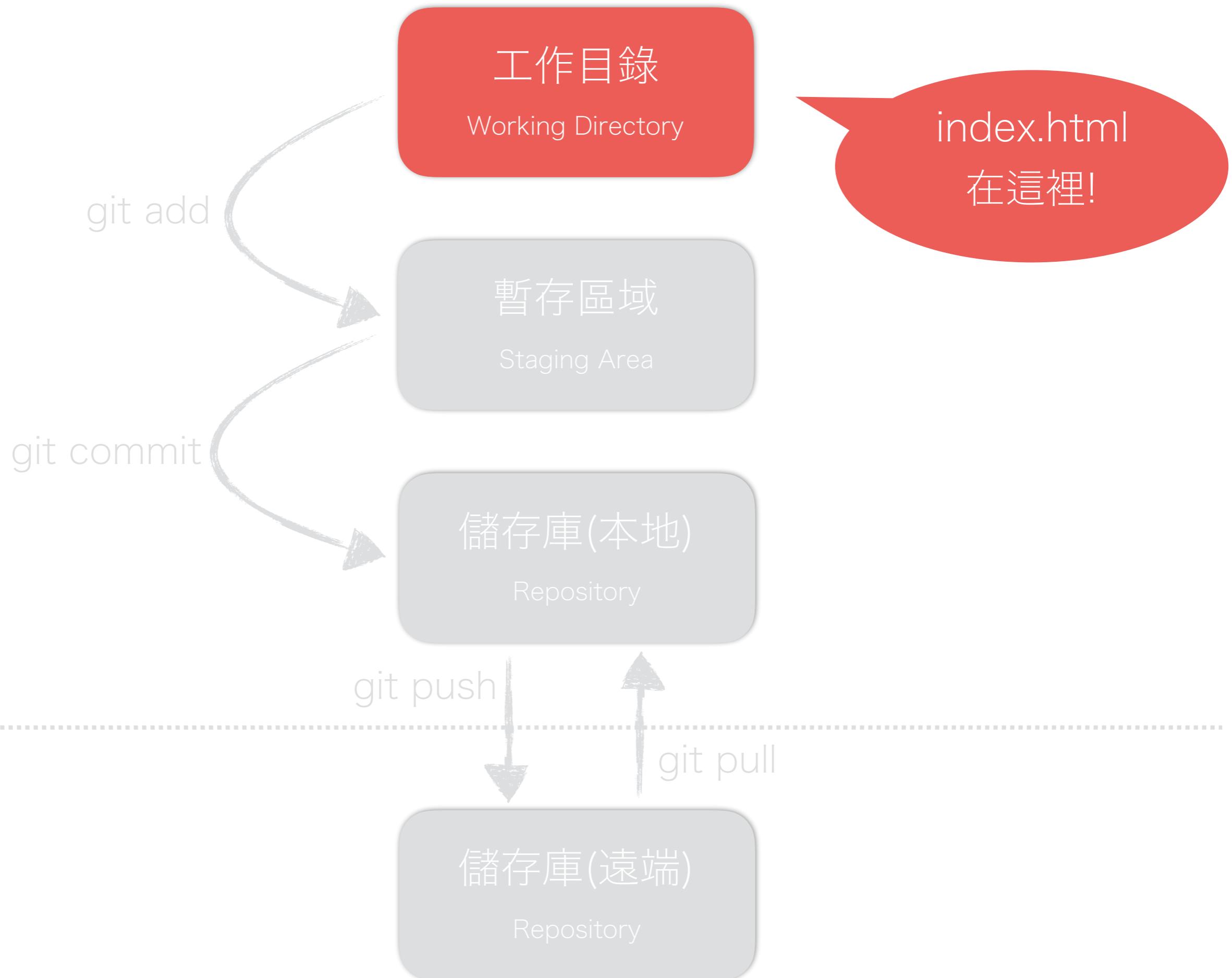
查看狀態

```
$ git status
```

```
Initial commit
```

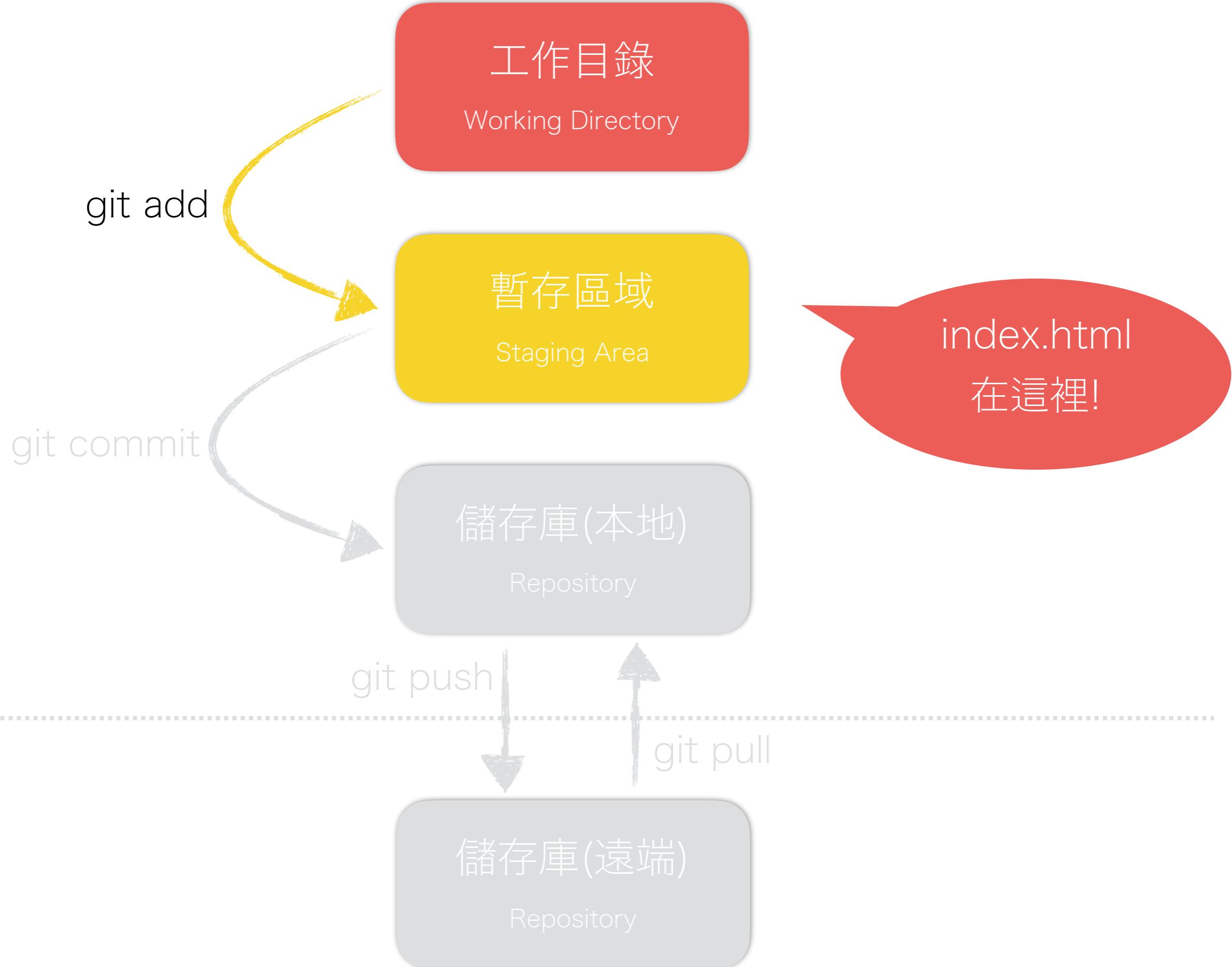
```
Changes to be committed:  
(use "git rm --cached <file>..." to unstage)
```

```
new file:   index.html
```



加到暫存區域 (Staging Area)

```
$ git add index.html
```



練習：

新增一個名為 “index.html” 的檔案

並增存放置暫存區

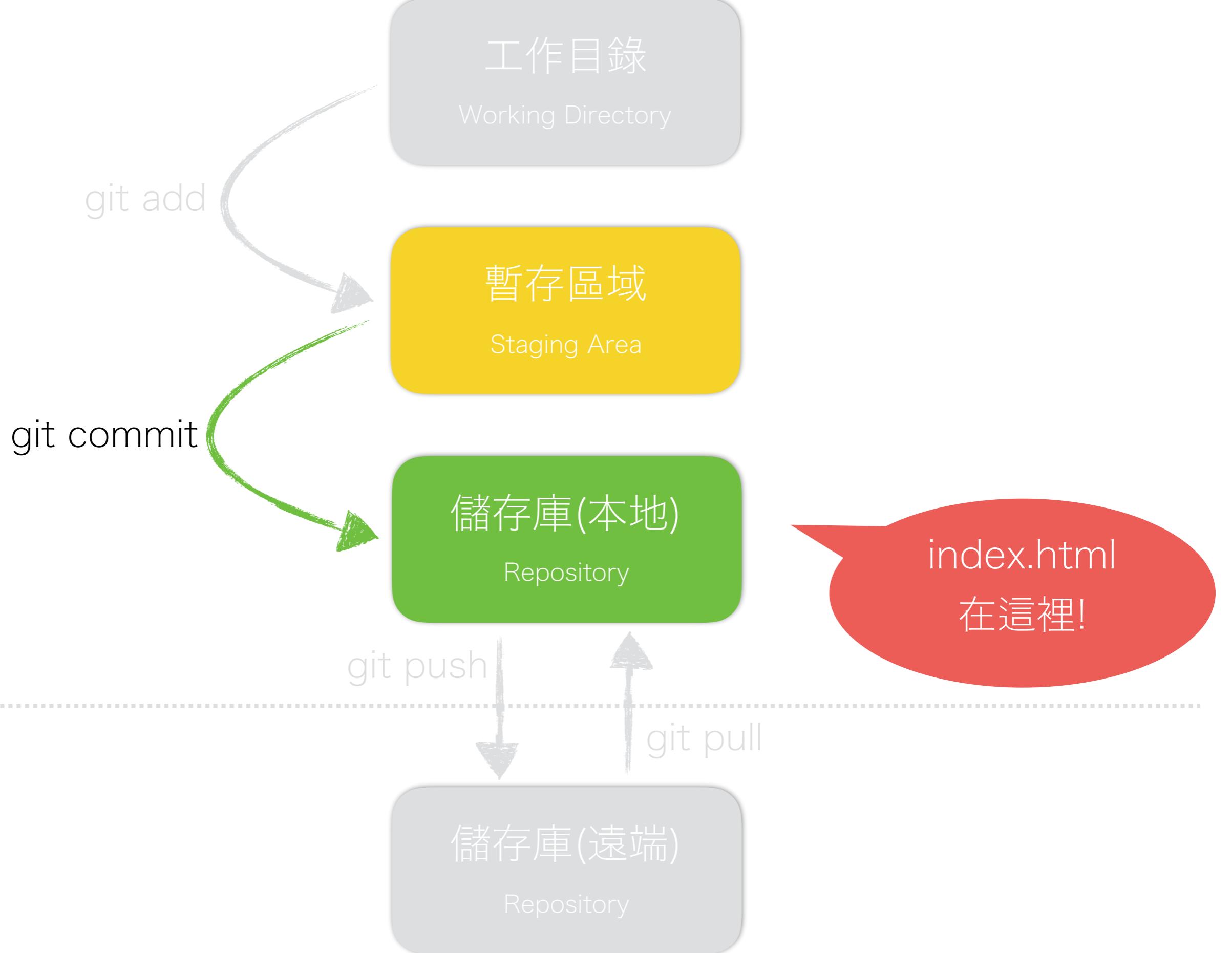
問題

git add 之後就算完成存檔了嗎？

提交 Commit

Commit

```
$ git commit -m "add index.html"  
  
[master (root-commit) cb96971] add index.html  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 index.html
```



練習：

just commit it :)

狀況

我知道怎麼 commit 了，但要怎麼看之前 commit 的結果？

檢視提交紀錄

```
$ git log  
  
commit cb96971765877246f4019e6771fd12d541c951e7  
Author: Eddie Kao <eddie@digik.com.tw>  
Date:   Sat Apr 16 07:34:21 2016 +0800  
  
    add index.html
```

檢視提交紀錄(一行精簡版)

```
$ git log --graph --oneline  
* cb96971 add index.html
```

IRON-RAILS

Stage

BRANCHES

master

REMOTES

origin

TAGS

SUBMODULES

OTHER

Short SHA	Subject
0e93768	update chapter 00
4479068	add chapter 29
f297835	fix typo
96fe1eb	Merge pull request #8 from sudoliyang/master
b1a4af1	fix markdown
3410b23	add chapter 28
ad90d88	typo
d5c3b87	add link
0b7b762	add chapter 27
6e3273c	add chapter 26
200c735	Merge branch 'master' of github.com:kaochenlong/learn-ruby-on-rails
f431771	Merge pull request #7 from sudoliyang/master
8051d86	fixed typo
f308018	update chapter 25
2734e5b	fix typo
d70fcff	add chapter 25

Subject: add chapter 27

Author:  Eddie Kao <eddie@digik.com.tw>

SHA: 0b7b762f5548a05

Date:  Thu Jan 12 2017 23:29:41 GMT+0800
(CST)

Parent: 6e3273c045f00f0

add chapter 27

- README.md
- chapter27-order.md
- images/chapter27/fsm.png

練習：

檢視一下你的提交紀錄

狀況

如果我只想要看某個特定檔案的
歷史紀錄...？

檢視單一檔案的歷史紀錄

```
$ git log -p FILENAME
```

狀況

等等.. 什麼時候有這一行程式碼？這一行誰寫的？

檢視每行程式碼的紀錄

```
$ git blame FILENAME
```

問題

什麼時候要 commit?

訊息很重要！

狀況

遇到澳客，做得很不爽，因為心情不好，不小心在 commit 訊息罵客戶了，要怎麼消掉？

修改最後一次 commit

```
$ git commit --amend
```

練習：

你剛剛不小心在提交的訊息裡罵了
客戶，想辦法把這個問題修正吧！

狀況

剛剛完成 commit, 但發現有一個檔案忘了加到, 又不想為了這個檔案重新再 commit 一次...

把 commit 併到上一次的 commit

```
$ git add hello.rb  
$ git commit --amend -m "update file  
and add hello.rb"
```

狀況

剛剛新增了一個 images 目錄，
但卻發現這個目錄好像沒辦法被
加到 Git 裡面？

注意事項：

空的目錄無法被提交!

如果還是想要提交一個空的目錄，只要隨便放一個檔案，甚至是放一個空的檔案在裡面就行了。慣例上會放個

名為 ".keep" 或 ".gitkeep" 的空白檔案。

練習：

建立空的資料夾，並且想辦法完成
Commit。

狀況

不小心把檔案或目錄刪掉了...

讓檔案或目錄回到最近一次的 commit 的狀態：

```
$ git checkout hello.rb
```

讓當前目錄回到最近一次的 commit 的狀態：

```
$ git checkout .
```

練習：

試著把某個檔案或目錄刪除，再用
git checkout 指令救回來。

狀況

剛剛 commit 了，有點後悔，想要再改點東西再重新 commit...

取消最後一次提交 (預設 mixed 模式)

```
$ git reset HEAD^
```

取消最後一次提交 (soft 模式)

```
$ git reset HEAD^ --soft
```

取消最後一次提交 (hard 模式)

```
$ git reset HEAD^ --hard
```

模式	mixed 模式	soft 模式	hard 模式
工作目錄	不變	不變	丟掉
暫存區	丟掉	不變	丟掉

模式	mixed 模式（預設）	soft 模式	hard 模式
Commit 拆出來的檔案	丟回工作目錄	丟回暫存區	直接丟掉

練習：

試著把取消最後一次的提交

問題

我會 commit 了，那我就一路
commit 到底就行了吧？

分支

Branch

對分支的誤解

你想像中的分支長什麼樣子？



photo by [Mark Fischer](#)



photo by Caroline

分支只是一張
貼在某個 Commit 上的貼紙

分支很便宜的
(為什麼?)

什麼時候要使用分支？

新增分支 "cat"

```
$ git branch cat
```

刪除分支 "cat"

```
$ git branch -d cat  
Deleted branch cat (was cb96971).
```

檢視分支

```
$ git branch
```

```
  cat
```

```
* master
```

切換到 "cat" 分支

```
$ git checkout cat
```

切換到 "dog" 分支，如果該分支不存在，
會自動建立新的分支

```
$ git checkout -b dog
```

練習：

1. 建立分支名稱 "fruit"
2. 切換到 "fruit" 分支
3. 新增檔案 "banana.html" 並且完成提交

合併分支

其實分支是沒辦法合併的

你合併的是分支所指到的 Commit

合併分支 "cat"

```
$ git checkout master  
$ git merge cat
```

問題

合併過的分支可以刪掉嗎？

練習：

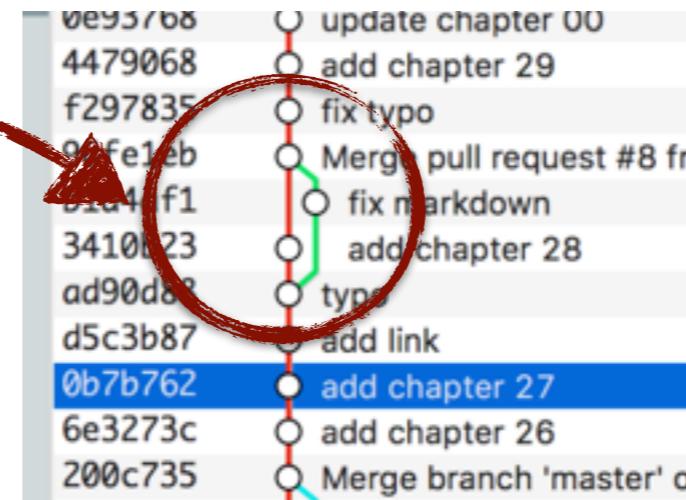
1. 切換回 "master" 分支
2. 合併 "fruit" 分支到 "master"
3. 刪除 "fruit" 分支(如果不想留的話)

狀況

要怎麼取消剛剛這次的合併？

狀況

就是這種東西



我看有些人的分支都有看起來像
分支的「小耳朵」線圖耶，怎麼
我的分支都沒有...？

因為不需要啊！

非快轉合併(Fash Forward)

```
$ git merge cat --no-ff
```

狀況

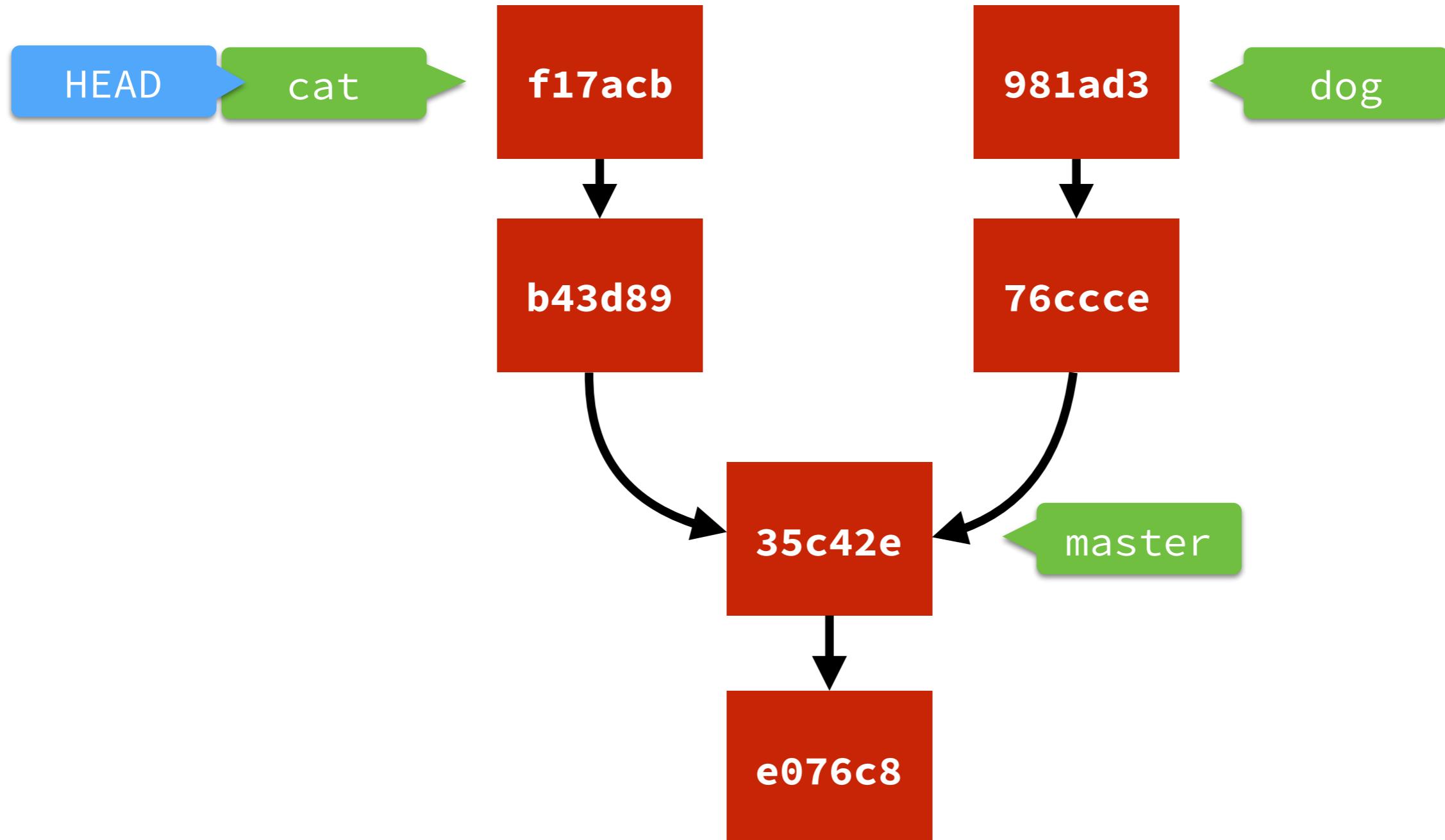
發生衝突(Conflict)了，怎麼辦？

使用 Rebase 合併



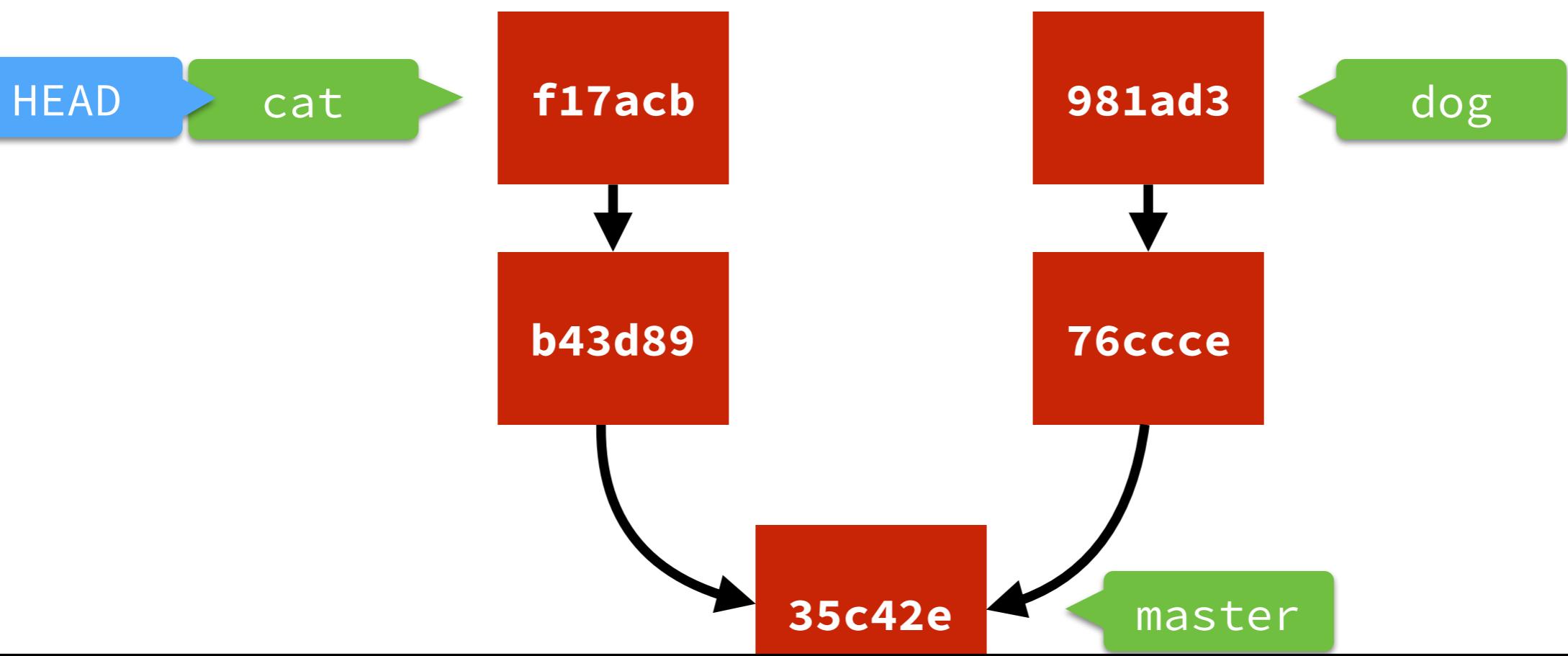
photo by [UGA College of Ag & Environmental Sciences](#)

Rebase 合併



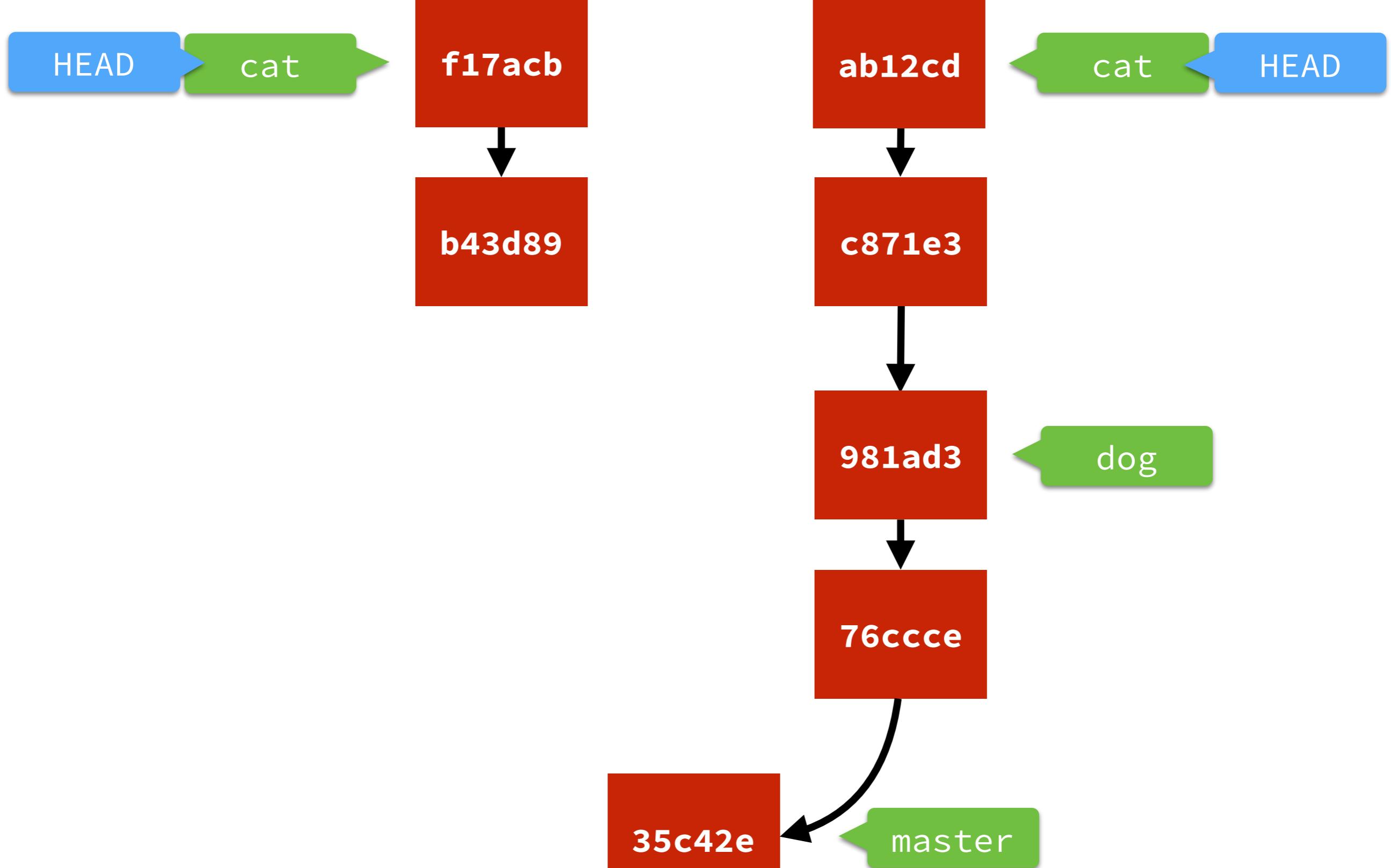
\$

Rebase 合併



```
$ git rebase dog
```

Rebase 合併



```
$ git rebase dog
```

Rebase 不是剪下貼上

狀況

要怎麼取消
剛剛這次的 rebase 合併？

問題

歷史紀錄可以修改嗎？

要小心修改歷史的副作用

修改提交

```
$ git rebase -i cb96971

# p, pick = use commit
# r, reword = use commit, but edit the commit message
# e, edit = use commit, but stop for amending
# s, squash = use commit, but meld into previous commit
# f, fixup = like "squash", but discard this commit's
log message
# x, exec = run command (the rest of the line) using
shell
```

狀況

有看到別人的軟體會說什麼1.0、
2.0 的，那個跟 Git 有關係嗎？

標籤

里程碑

新增標籤 "1.0.0"

```
$ git tag 1.0.0
```

檢視標籤

```
$ git tag
```

練習：

請新增一個 "1.0.0-beta" 的標籤

問題

標籤跟分支有什麼不同？

狀況

我知道怎麼在自己電腦用Git了，
但要怎麼上傳到 GitHub 跟別人
共用？

遠端操作

遠端節點

新增遠端節點

```
$ git remote add origin  
git@github.com:kaochenlong/dummy-  
git.git
```

刪除遠端節點

```
$ git remote rm origin
```

檢視遠端節點

```
$ git remote -v
```

```
origin git@github.com:kaochenlong/dummy-git.git (fetch)
origin git@github.com:kaochenlong/dummy-git.git (push)
```

上傳 Push

上傳

```
$ git push origin master
```

上傳 "cat" 分支到 origin 節點

```
$ git push origin cat
```

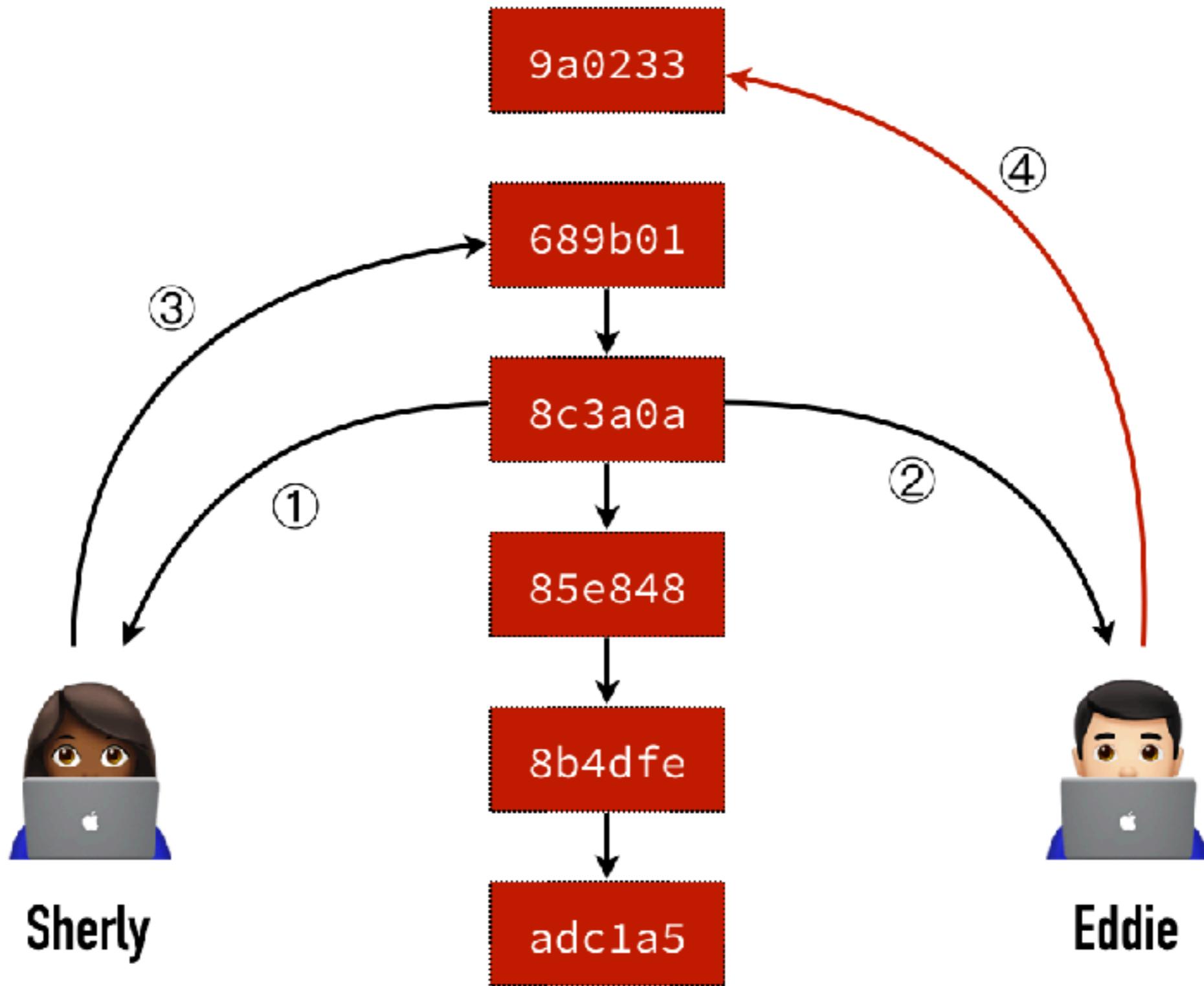
問題

怎麼刪除遠端分支？

刪除遠端 "cat" 分支

```
$ git push origin :cat
```

為什麼有時候推不上去？



下載更新 Pull

下載

```
$ git pull origin master
```

git pull = git fetch + git merge

從伺服器上取得 Repository

取得 Repository

```
$ git clone git://github.com/  
kaochenlong/eddie-vim.git
```

問題

Git 的 pull 是下載, clone 好像
也是下載... 有不一樣嗎?

狀況

你手邊的工作做到一半...

老闆：「那個誰誰誰，網站掛了，你趕快先來修一下這個功能...」

把目前狀況存下來

```
$ git stash save
```

把最後一次的 stash 拿出來用

```
$ git stash apply
```

刪除最後一次的 stash

```
$ git stash drop
```

其它常見狀況

狀況

有些比較機密的檔案我不想放在
Git 裡面一起備份...

.gitignore

<https://github.com/github/gitignore>

狀況

你不小心把帳號密碼寫在某個檔案裡，提交而且推出去了...

針對每個節點刪除特定檔案

```
$ git filter-branch --tree-filter  
"rm -f config/password.txt"
```

狀況

你正在 develop 分支進行某個錯誤的修正，但你突然發現這個錯誤在某個分支的某個 commit 已經修過了...

合併某個特定節點

```
$ git cherry-pick 823520ed
```

合併某個特定節點前編輯訊息

```
$ git cherry-pick 823520ed --edit
```

取得某個特定節點內容但不進行合併

```
$ git cherry-pick 823520ed --no-commit
```

狀況

精神不好，不小心輸入了 git
reset HEAD[^] --hard 指令，檔
案還救得回來嗎？

調閱 reflog

```
$ git reflog
```

回到 reset 之前的節點

```
$ git reset --hard 823520ed
```

狀況

前一天沒睡飽，不小心把還沒合併的分支刪掉了，救得回來嗎？

調閱 reflog

```
$ git reflog
```

使用那個節點做出新的 branch

```
$ git checkout -b new_branch_name  
823520ed
```

Github

GitHub 是什麼？

— 款 Git Repository Server

開發者們的 facebook :)



Search GitHub

Pull requests Issues Marketplace Gist

**RUBYIST!**

Eddie Kao

kaochenlong

iOS App/Ruby/Rails Developer and Instructor, PHPConf / WebConf Taiwan Founder, Rails Girls Taipei Organizer, 5xRuby Founder.

Developer Program Member

5xRuby 五倍の紅宝石

Taiwan, Taipei

eddie@digik.com.tw

<http://kaochenlong.com/>

Organizations

[Overview](#)[Repositories 108](#)[Stars 607](#)[Followers 450](#)[Following 11](#)

Popular repositories

Customize your pinned repositories

[eddie-vim](#)

Yet another vimrc

VimL ★ 333 ⚡ 143

[eddie-vim2](#)

Yet another vimrc

VimL ★ 96 ⚡ 26

[learn-ruby-on-rails](#)

為你自己學 Ruby on Rails

★ 81 ⚡ 22

[programming-basic](#)

程式設計入門 - 使用 Ruby 及 Swift

★ 20 ⚡ 4

[rails_app_template](#)

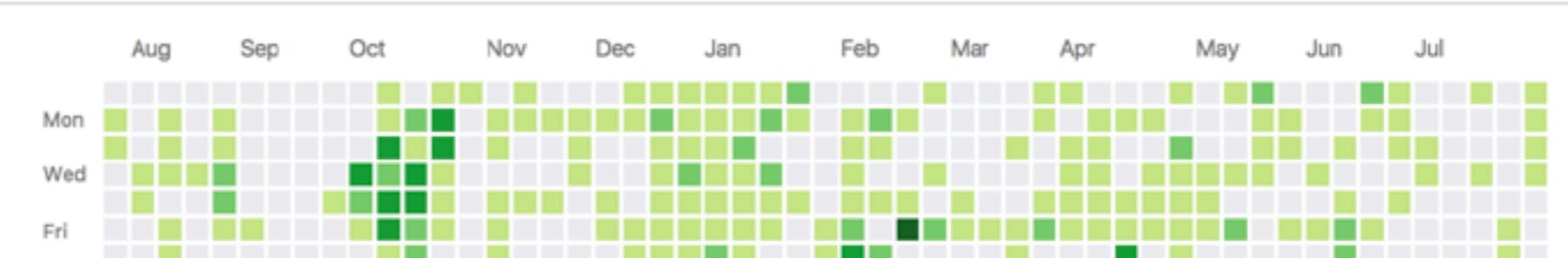
Ruby ★ 17 ⚡ 12

[ntub_homework](#)

Ruby ★ 12 ⚡ 47

1,206 contributions in the last year

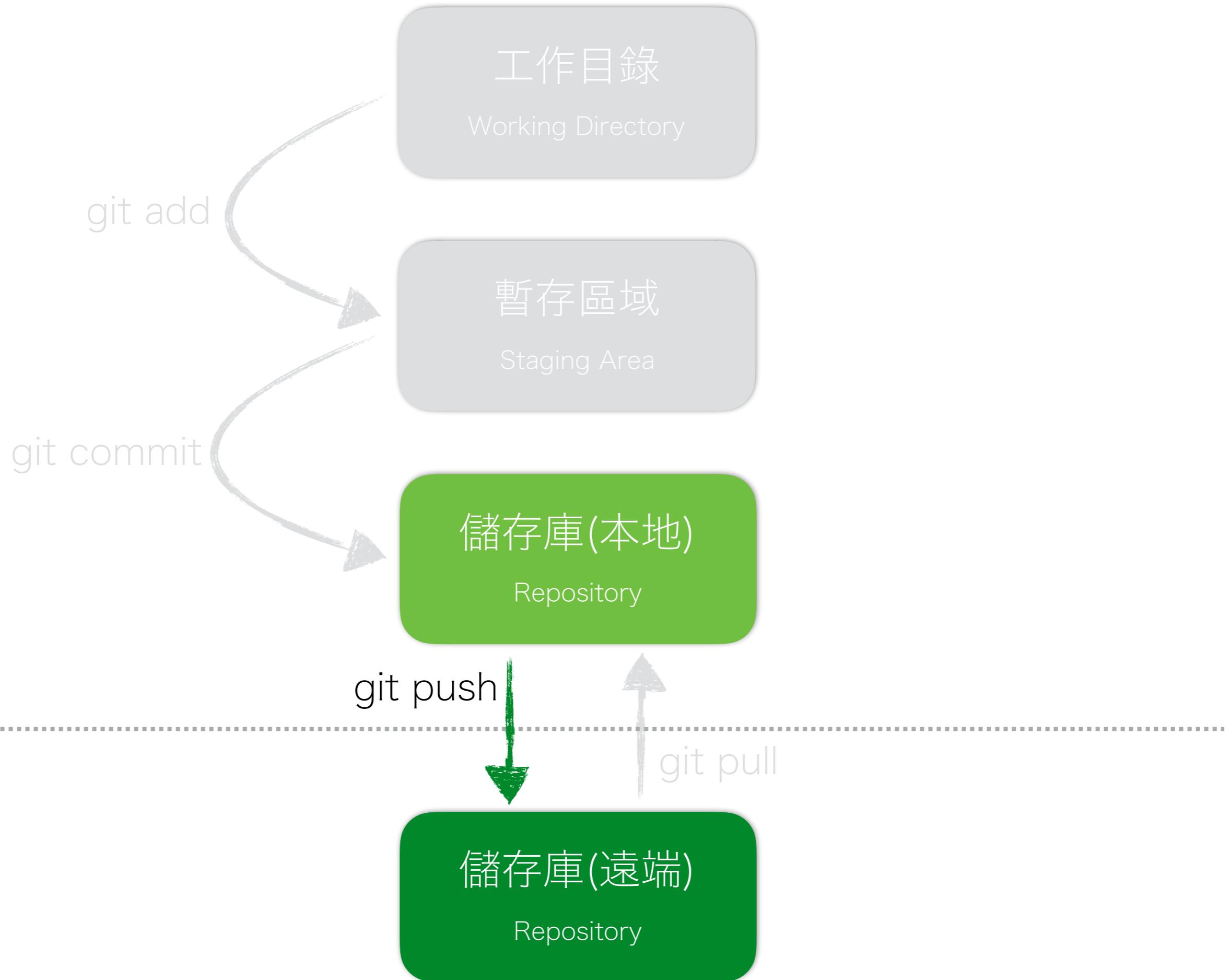
Contribution settings ▾

[Learn how we count contributions.](#)

跟厲害的開發者們交朋友 :)

開發者最好的履歷!

使用 Github



加入遠端節點

加入遠端節點，並命名為 origin

```
$ git remote add origin ....
```

推上去！

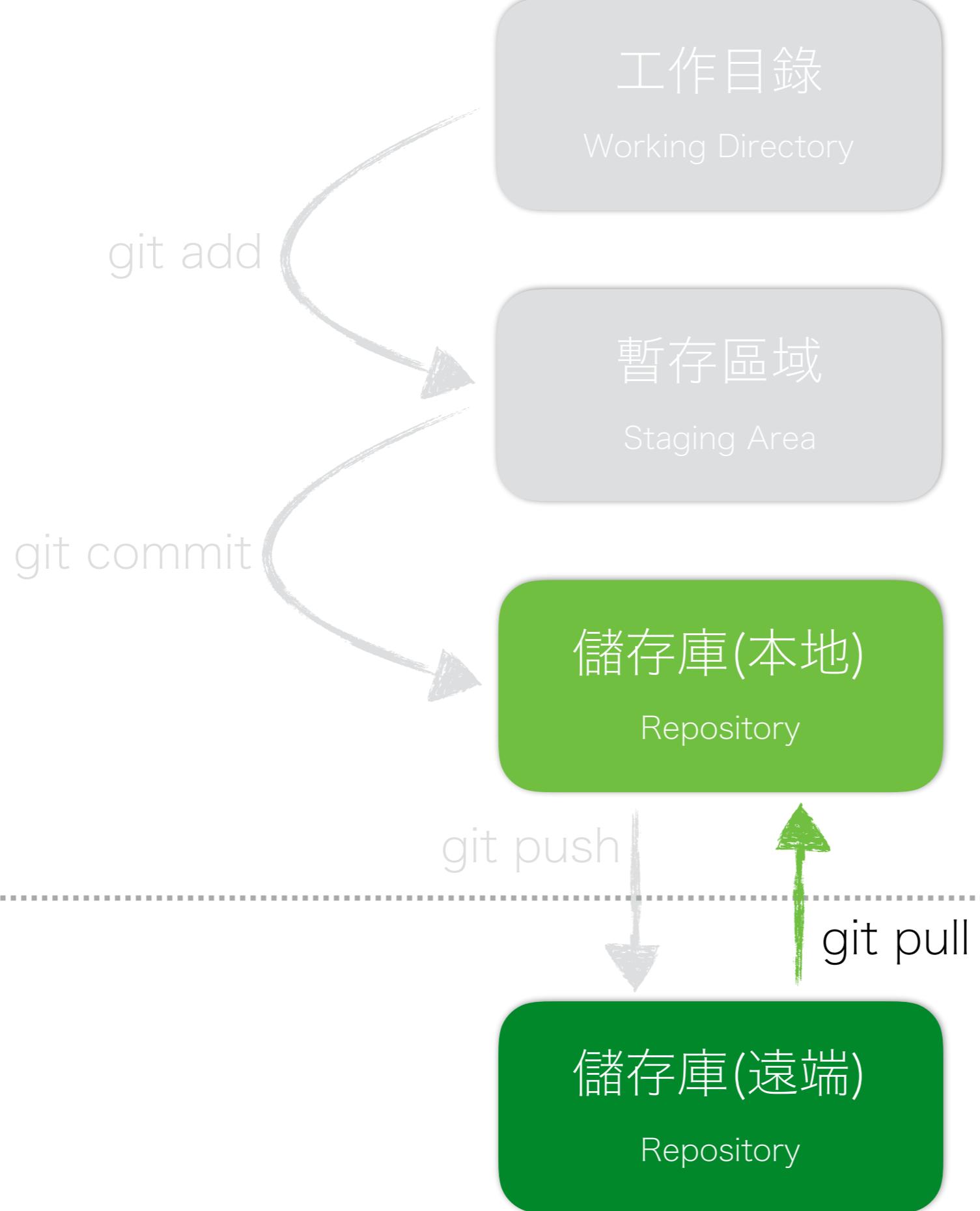
把本地的 master 分支推上 origin 節點

```
$ git push -u origin master
```

練習：

在 GitHub 上註冊帳號，新增一個 Repository，並上傳你目前的進度。

拉下來！



把本地的 master 分支推上 origin 節點

```
$ git pull origin master
```

取得遠端 origin 上有但本地沒有的東西

```
$ git fetch origin
```

用本地的 master 合併遠端的 master

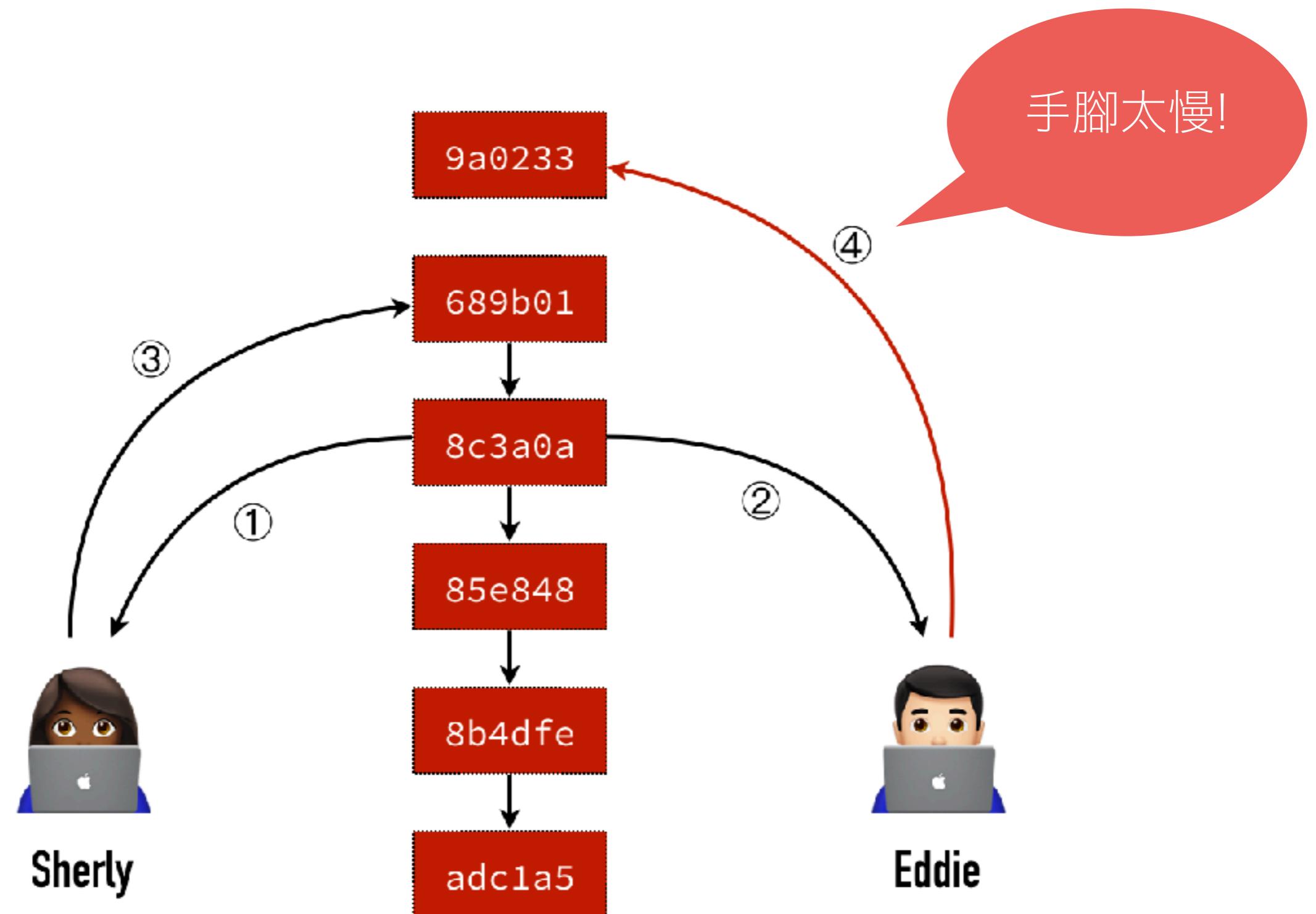
```
$ git merge origin/master
```

pull = fetch + merge

直接從 origin 拉一份下來，並直接合併

```
$ git pull origin
```

為什麼有時候會推不上去？



A close-up of Yoda's face from Star Wars. He has his characteristic green skin, large ears, and wrinkles. A red speech bubble originates from his mouth, containing the text "本來不想用這招的！".

本來不想用
這招的！

git push -f

怎麼刪除遠端分支？

刪除在 origin 節點上的 cat 分支

```
$ git push origin :cat
```

與其它開發者互動

Fork & Pull Request (PR)

練習：

1. fork 一個坐在你旁邊的那位同學的專案.
2. 做些簡單的修改
3. 發送一個 pull request.

狀況

你從 GitHub 上 fork 了專案，該專案後來因為不斷的更新，但你的專案還是停在當初 fork 時候的狀態...

step 1: 新增原來 repo 的遠端節點

```
$ git remote add upstream ORIGIN_REPO
```

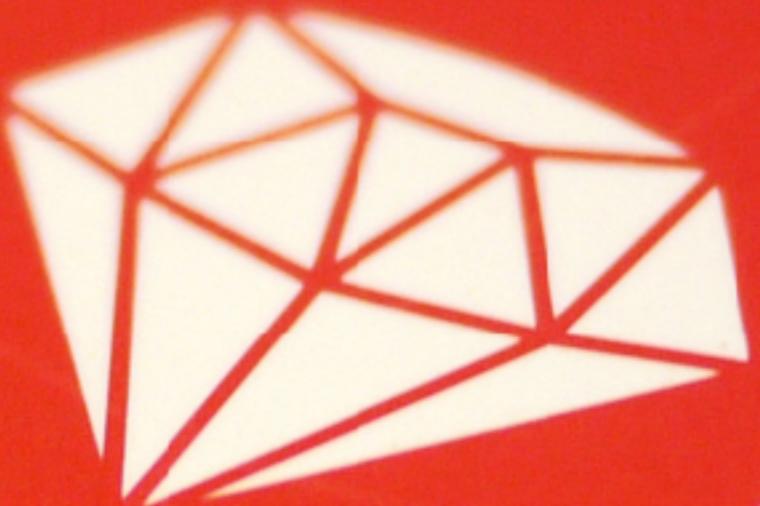
step 2: 取得該節點檔案

```
$ git fetch upstream
```

step 3: 合併

```
$ git merge upstream/master
```

5x



五倍紅寶石
KAOCHENLONG.COM

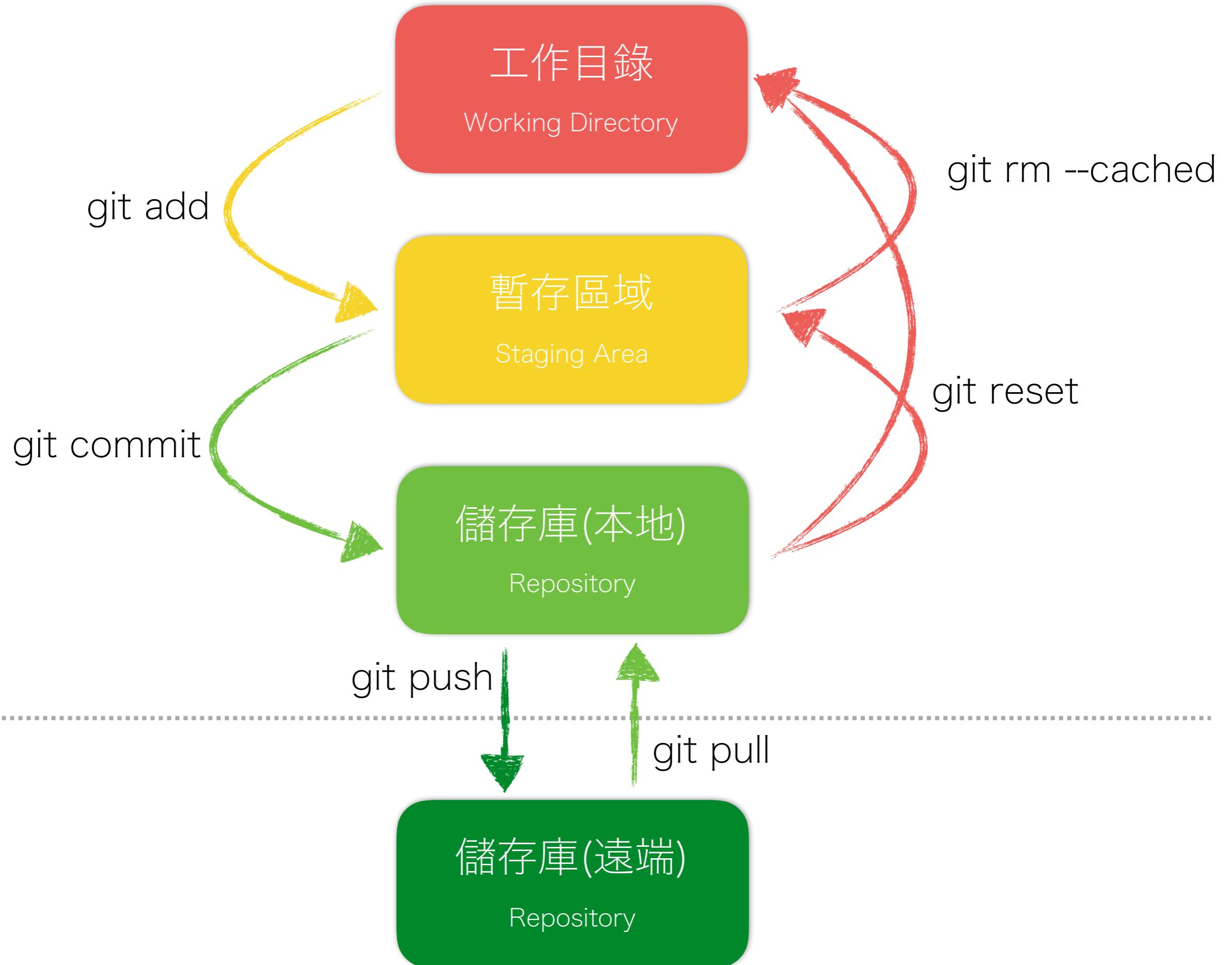
j.tw

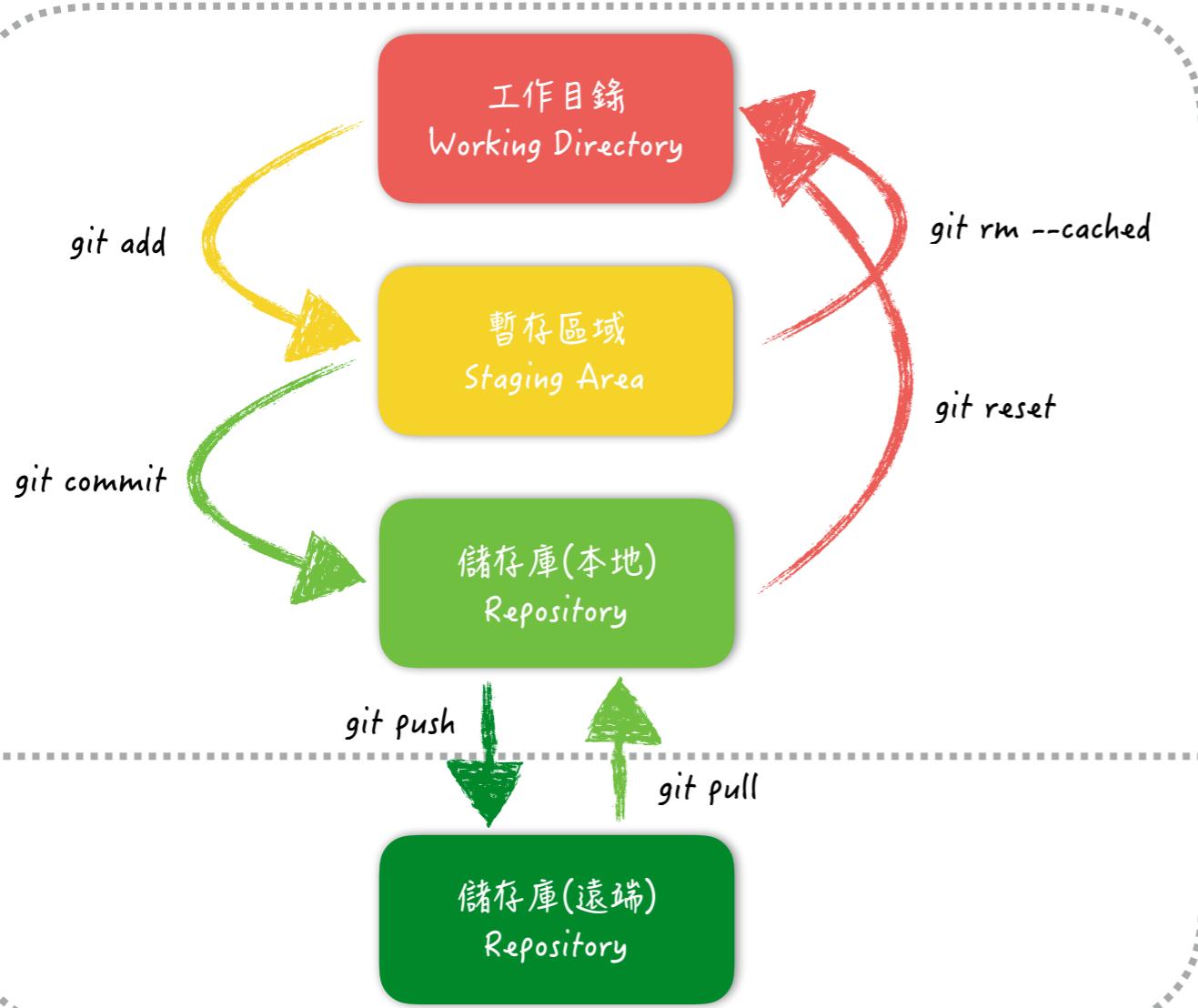
高見龍

- Blog
- Facebook
- Twitter
- Email
- Mobile

<http://kaochenlong.com>
<http://www.facebook.com/eddiekao>
<https://twitter.com/eddiekao>
eddie@5xruby.tw
[+886-928-617-687](tel:+886-928-617-687)

Git 小抄





新增遠端節點 "origin" :

```
$ git remote add origin REMOTE_URL
```

刪除遠端節點 "origin" :

```
$ git remote rm origin
```

檢視遠端節點 :

```
$ git remote -v
```

把 "master" 分支內容推往 "origin" 節點 :

```
$ git push origin master
```

把遠端 "origin" 節點的 "master" 拉回本機並進行合併 :

```
$ git pull origin master
```

設定 :

```
$ git config --global user.name "5xruby"
$ git config --global user.email "hi@5xruby.tw"
```

初始化 :

```
$ git init
```

把檔案加到暫存區域 :

```
$ git add FILENAME
```

查看狀態 :

```
$ git status
```

提交 :

```
$ git commit -m "add index.html"
```

檢視紀錄 :

```
$ git log
```

取消最後一次提交 :

```
$ git reset HEAD^
```

檢視目前分支 :

```
$ git branch
```

新增分支 "5xruby" :

```
$ git branch 5xruby
```

切換分支到 "5xruby" :

```
$ git checkout 5xruby
```

合併分支 "5xruby" :

```
$ git merge 5xruby
```

刪除已合併分支 "5xruby" :

```
$ git branch -d 5xruby
```



Line 群組