git

從微觀到宏觀

http://ihower.tw

2014/10/29@livehouse.in

我是誰

- 張文鈿 a.k.a. ihower
  - http://ihower.tw
- Instructor at ALPHA Camp
  - http://alphacamp.tw
- Git user since 2008

Agenda

- 微觀：Git 的內部設計
- 宏觀：Git 的分支開發流程和策略

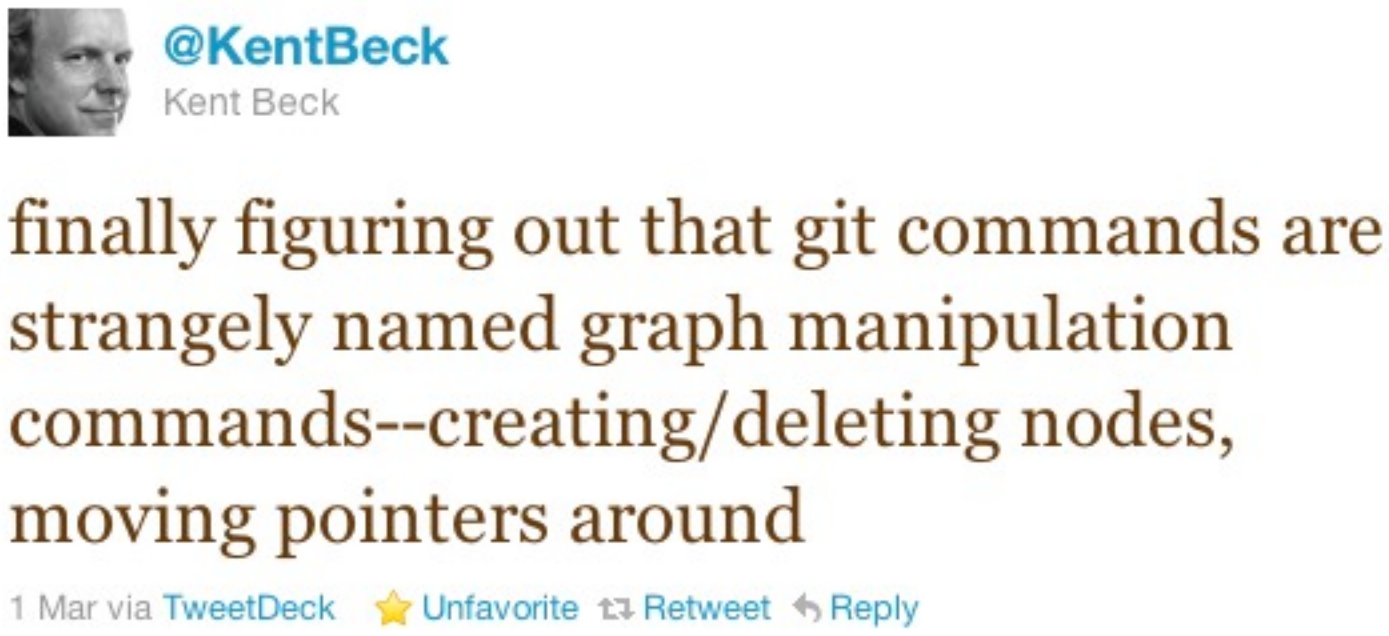
不是 Git 入門教學!

Part1. Git 的內部設計

小測驗

如何用 Git 底層指令，不用 git add 和 git commit 指令進行 commit 動作?

用 Graph 概念理解

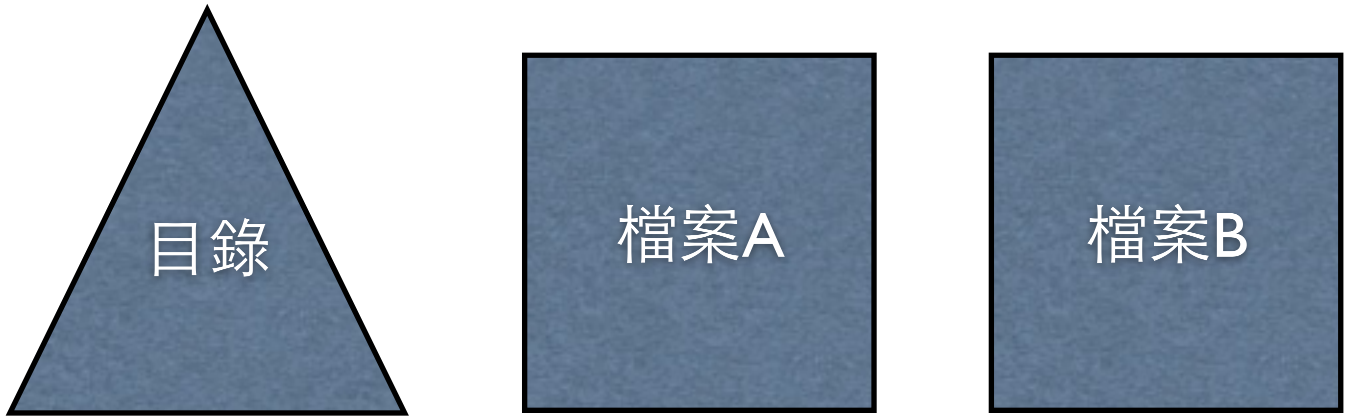


@KentBeck
Kent Beck

finally figuring out that git commands are strangely named graph manipulation commands--creating/deleting nodes, moving pointers around

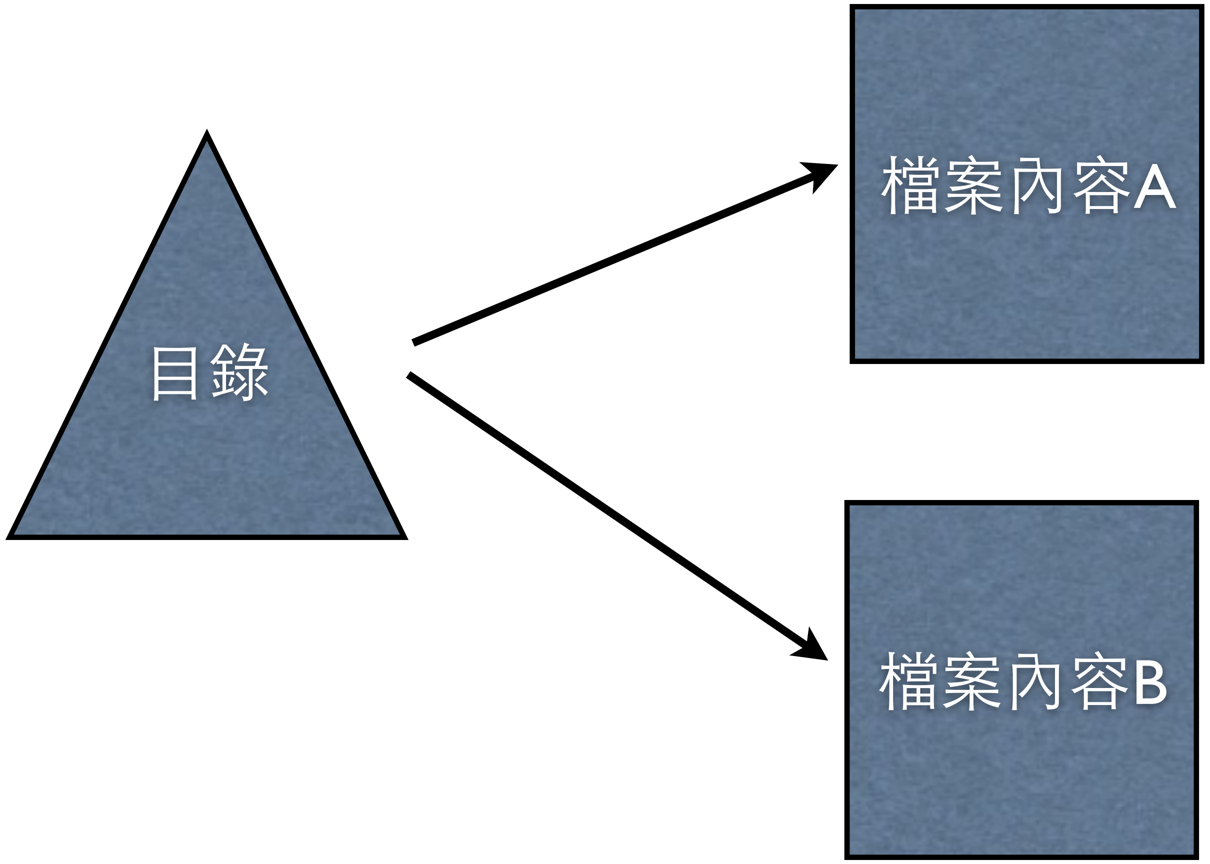
1 Mar via TweetDeck

working area



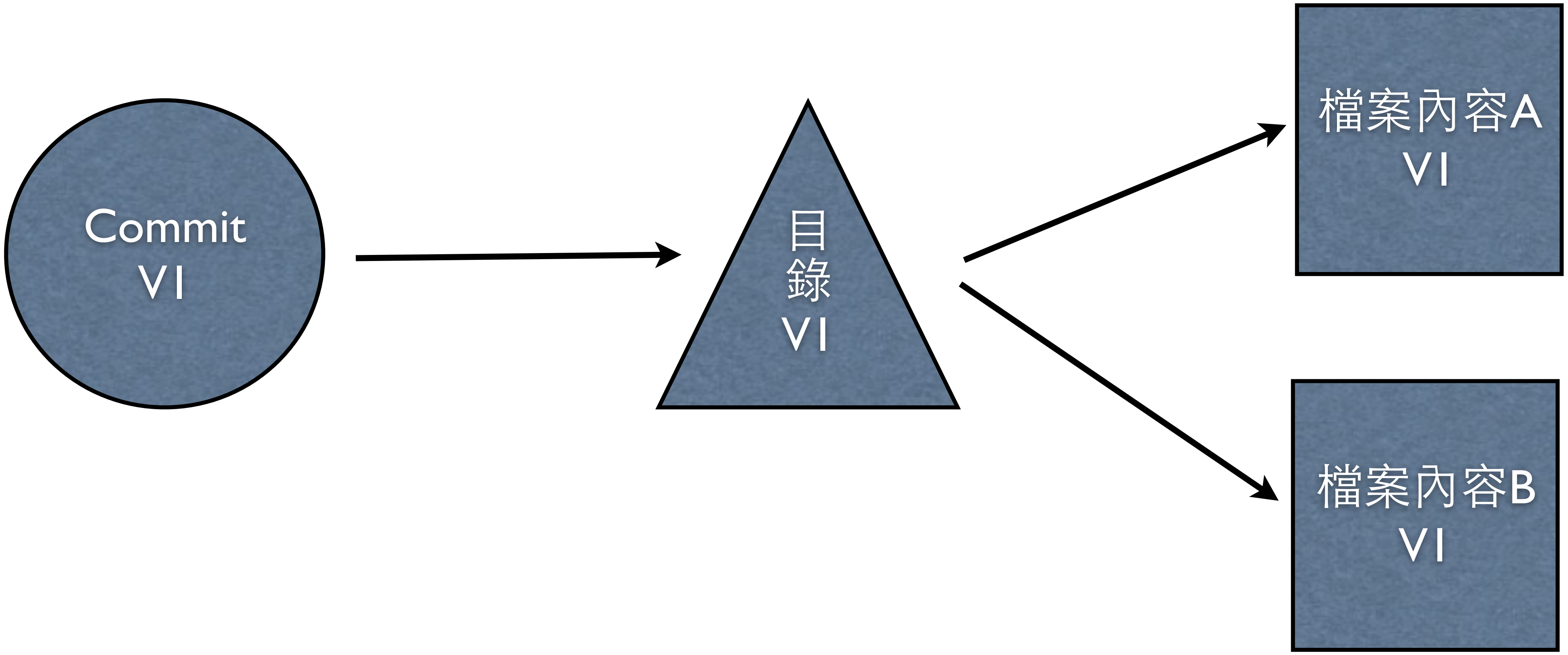
git add .

(將目錄節點和檔案內容節點關聯起來)



git commit

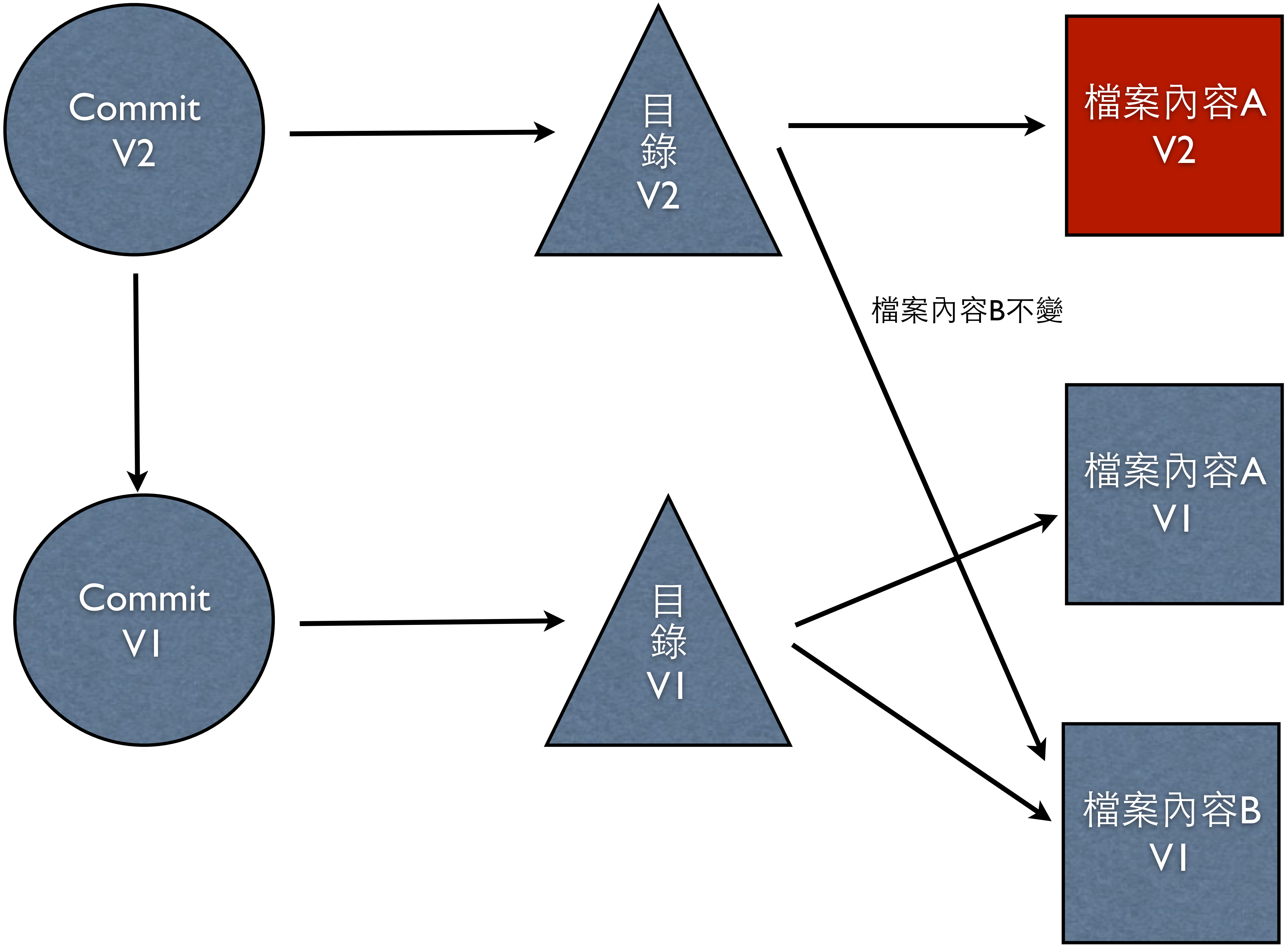
(產生commit節點，指向目錄節點)



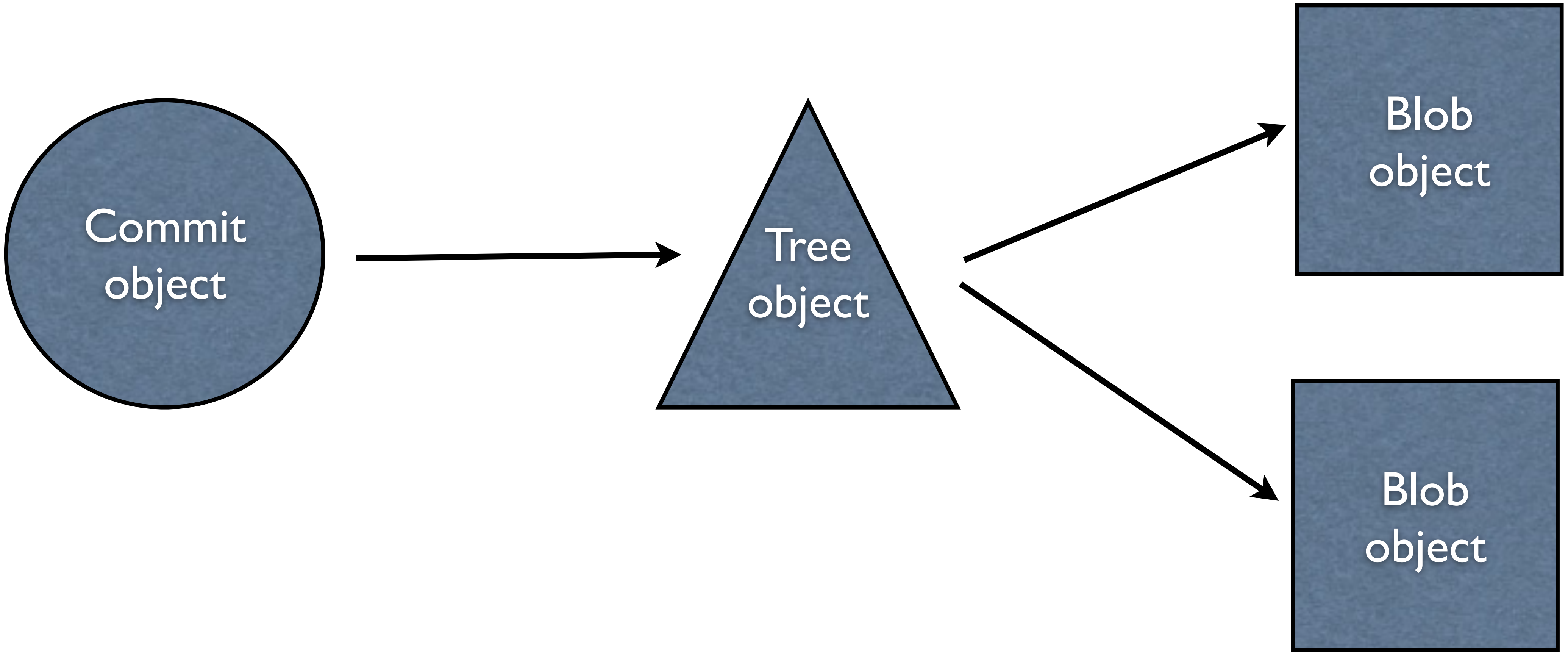
接下來我們修改檔案 A 成為V2 版本，檔案B不變

git commit (cont.)

(產生commit V2節點，指向parent commit節點)

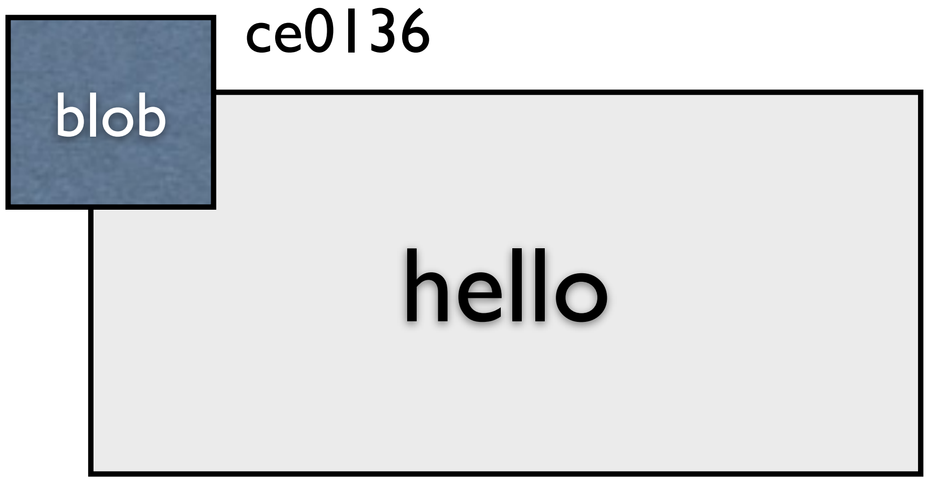


Git is objects database



儲存內容(demo)

- git init
- echo hello > hello.txt
- git add .
- tree .git
- 存在 .git/objects/ce/013625030ba8dba906f756967f9e9ca394464a
- 這是 hello 內容的 SHA1
  - printf "blob 6\x00hello\n" | shasum
  - echo "hello" | git hash-object --stdin
- git cat-file -p ce0136



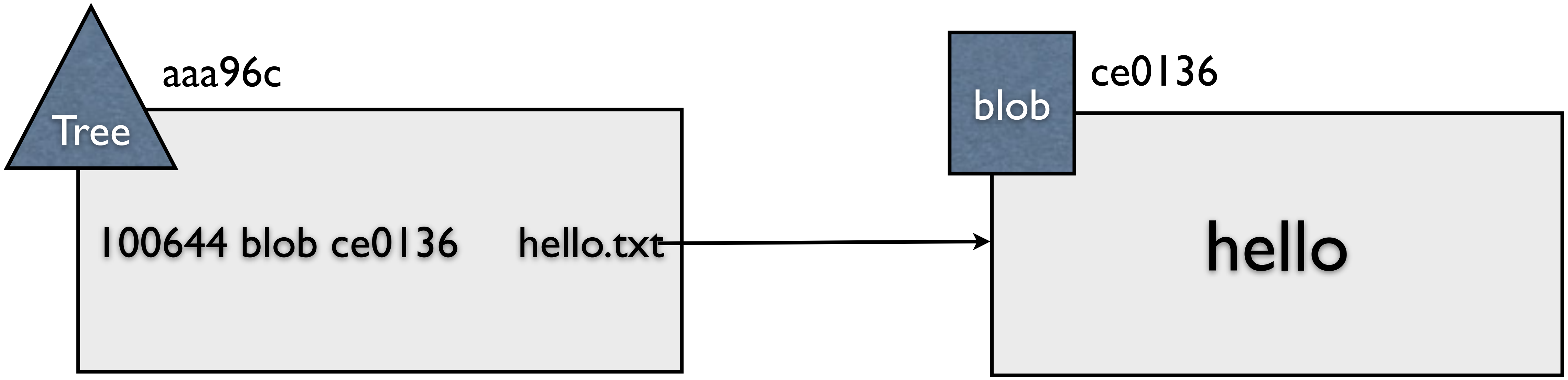
blob object 的實際檔案名稱
.git/objects/ce/013625030ba8dba906f756967f9e9ca394464a

Blob object

- Git 是 Content-addressable filesystem
- Blob 沒有 metadata，沒有檔名資訊
- Blob object 的儲存檔名，是根據內容產生的 SHA1
- 內容一樣的檔案，根據 SHA1 演算法只會存成同一份檔案，不會浪費空間

儲存目錄 (demo)

- git write-tree
  (根據 staging area 產生 Tree object)
- git cat-file -p aaa96c

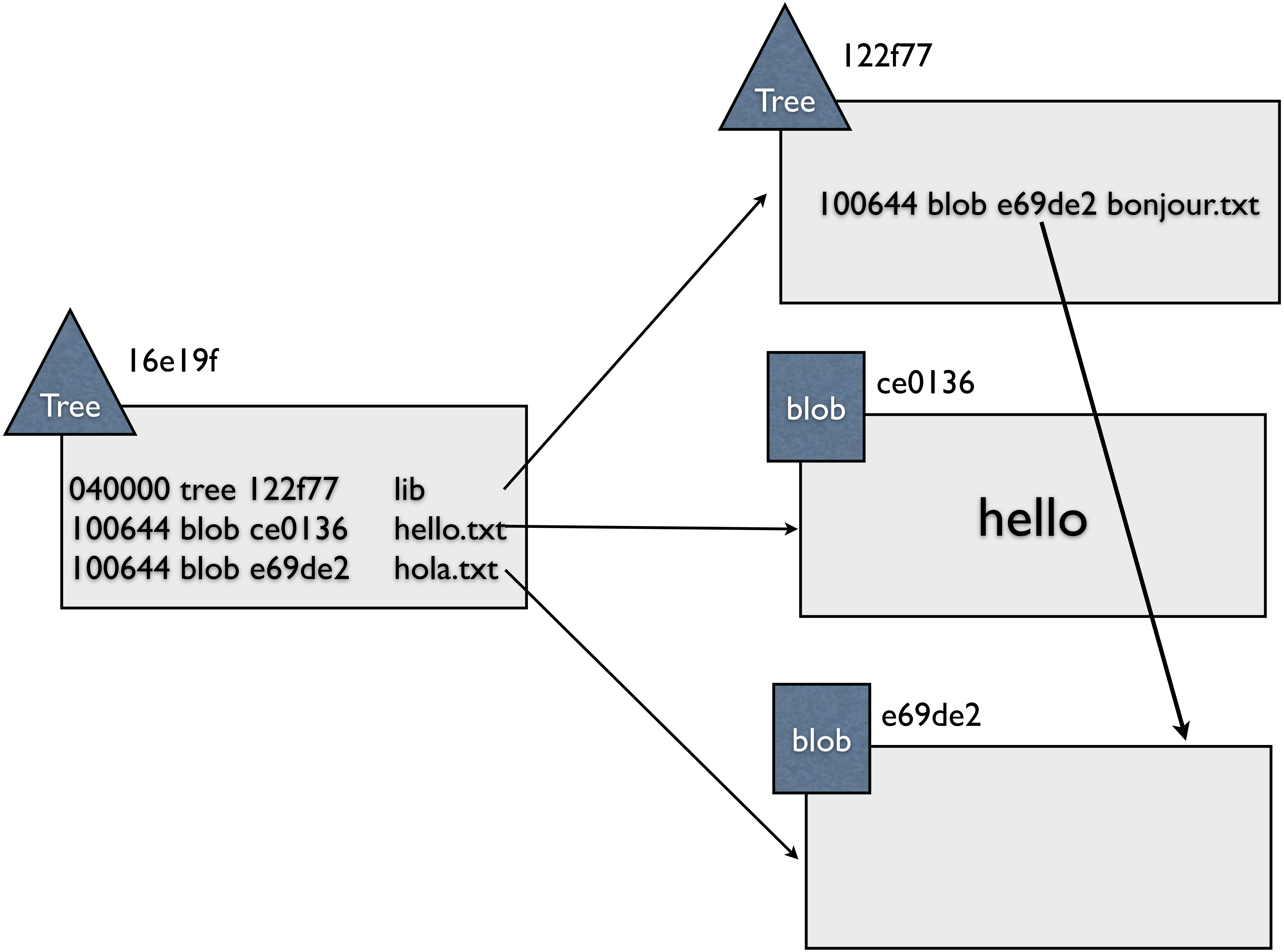


Tree object 的實際檔案名稱
.git/objects/aa/a96ced2d9a1c8e72c56b253a0e2fe78393feb7

其中 100644 為檔案模式，表示這是一個普通檔案
100755 表示可執行檔，120000 表示 symbolic link

儲存目錄 (cont.)

- 新增兩個空白檔案和子目錄
  - touch hola.txt & mkdir lib & touch lib/bonjour.txt
  - git add .
  - git write-tree
  - git cat-file -p 16e19f (觀察這個 tree)
  - git cat-file -p e69de2 (觀察其中的 lib tree)



Tree 16e19f

040000 tree 122f77 lib
100644 blob ce0136 hello.txt
100644 blob e69de2 hola.txt

Tree 122f77

100644 blob e69de2 bonjour.txt

blob ce0136

hello

blob e69de2

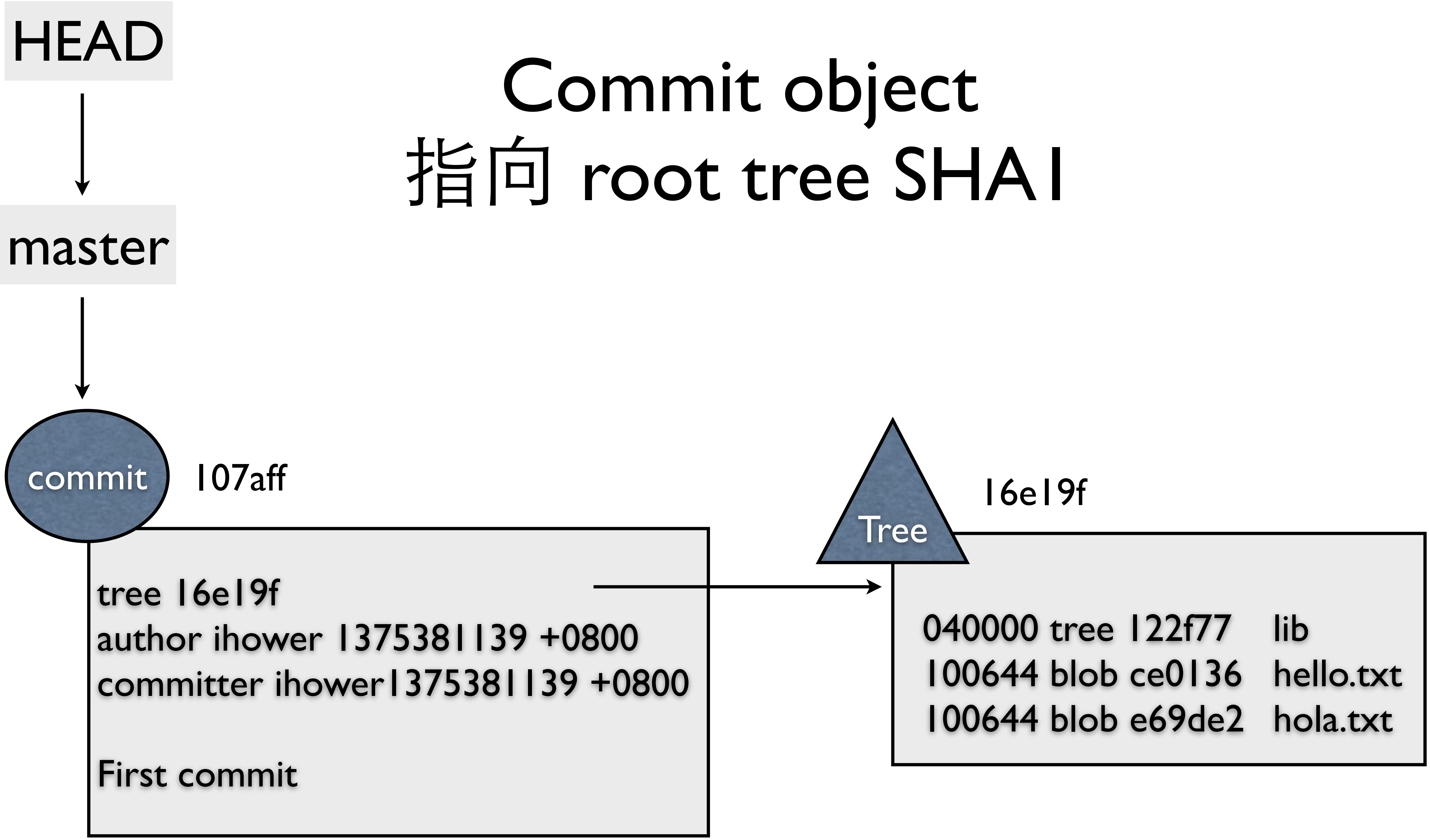
Tree object

- Git 用 Tree object 把 Blob object 組織起來，包括檔案命名和目錄結構
  - Blob object 並沒有包含檔案名稱和目錄結構
  - Tree object 裡面還可以有 Tree object 子目錄
- Tree object 的檔名，一樣是根據內容產生 SHA1

遞交 Commit (demo)

- git commit-tree 16e19f -m “First commit”
- git cat-file -p 107aff
- cat .git/HEAD
- cat .git/refs/heads/master
- git update-ref refs/heads/master 107aff
- git rev-parse HEAD

Commit object
指向 root tree SHA1



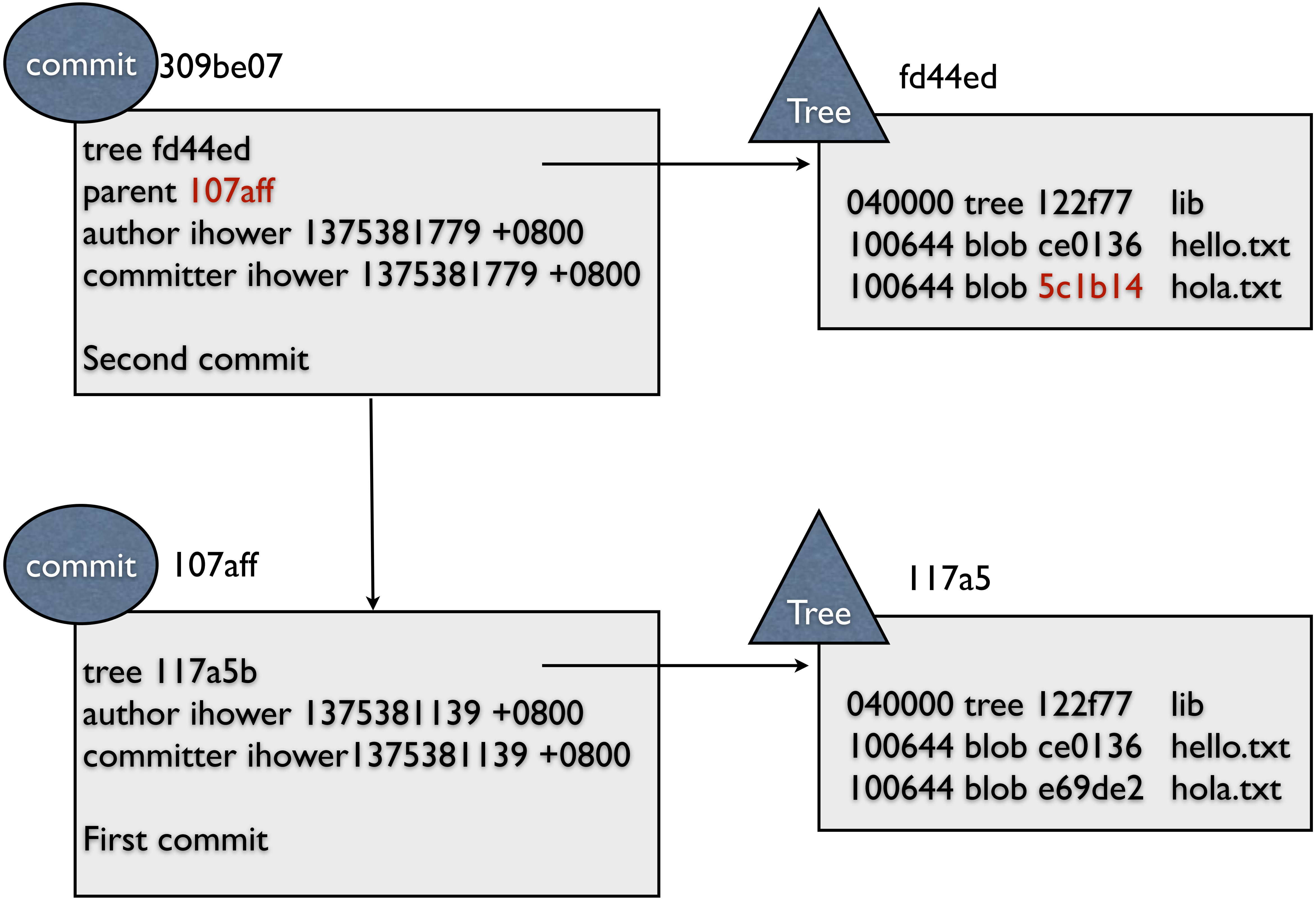
```
tree 16e19f
author ihower 1375381139 +0800
committer ihower1375381139 +0800

First commit
```

```
040000 tree 122f77    lib
100644 blob ce0136    hello.txt
100644 blob e69de2    hola.txt
```

再次遞交 Commit (demo)

- 修改 hola.txt 檔案，加入 hola 字串
- git commit -am “Second commit”
- git cat-file -p 309be07

Commit object
指向 parent commit SHA1



commit 309be07

```
tree fd44ed
parent 107aff
author ihower 1375381779 +0800
committer ihower 1375381779 +0800

Second commit
```

Tree fd44ed

```
040000 tree 122f77    lib
100644 blob ce0136    hello.txt
100644 blob 5c1b14    hola.txt
```

commit 107aff

```
tree 117a5b
author ihower 1375381139 +0800
committer ihower1375381139 +0800

First commit
```

Tree 117a5

```
040000 tree 122f77    lib
100644 blob ce0136    hello.txt
100644 blob e69de2    hola.txt
```

Commit object

- 紀錄 root tree SHA1
- 紀錄 parent commit SHA1
- 紀錄作者、時間和 commit message 資訊
- Commit object 的檔名，一樣是根據內容產生 SHA1

Git commit 動作流程

- 用內容產生 blob object
- 寫入 file mode, blob SHA1, file name 到 staging area
- 根據 staging area 產生 Tree object
- 用 root tree SHA1 和 parent commit SHA1 產生 commit object
- 用 commit SHA1 更新 master 參考

測驗解答

如何不用 git add 和 git commit 指令進行 commit 動作?

git add

```
echo "hola" | git hash-object -w --stdin
git update-index --add --cacheinfo \
    100644 5c1b14949828006ed75a3e8858957f86a2f7e2eb hola.txt
```

---

git commit

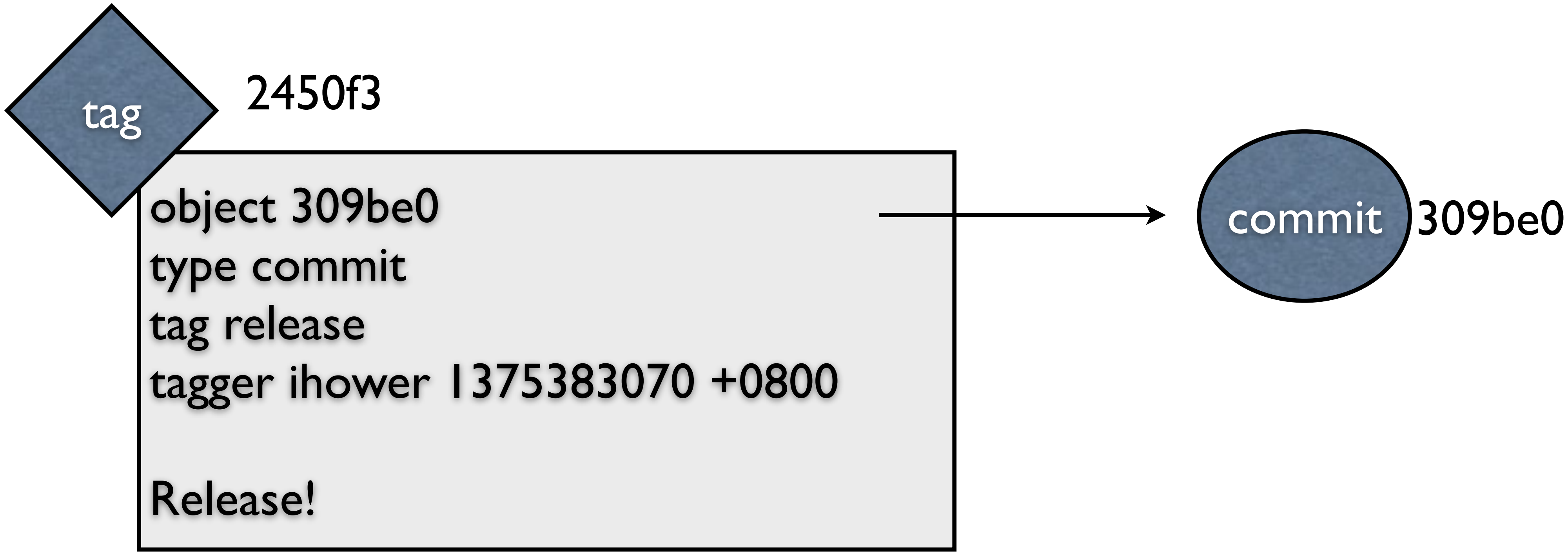
```
git write-tree
git commit-tree 27b9d5 -m "Second commit" -p 30b060
git update-ref refs/heads/master 97b806c9e5561a08e0df1f1a60857baad3a1f02e
```

https://gist.github.com/ihower/6132576

Tag object

(Tag 分兩種：annotated tag 才會產生 object)

- git tag -a release
- git rev-parse release
- git cat-file -p 2450f3



tag

2450f3

```
object 309be0
type commit
tag release
tagger ihower 1375383070 +0800

Release!
```

commit

309be0

小結論:

Git 有四種 Objects

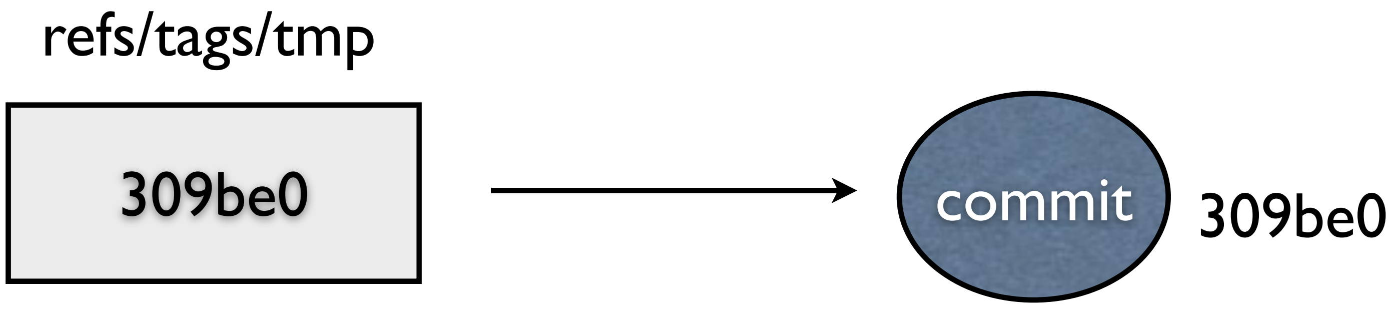
- Blob
- Tree
- Commit
- Tag

References 參照

- 單純一個檔案紀錄一個 SHA1參照
  - Tag reference
  - Branch reference
  - HEAD reference (指向目前所在的 branch)

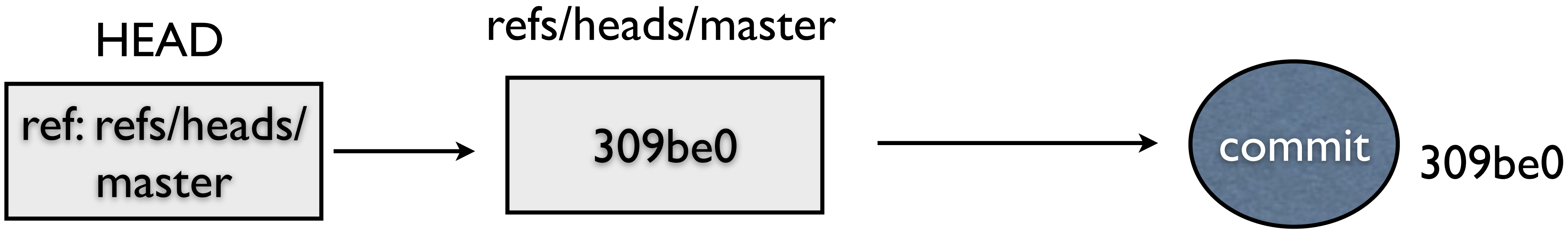
Tag reference

- git tag tmp
- cat .git/refs/tags/tmp
- 不像 object 資訊豐富，reference 內容只有 Commit object SHA1



Branch 和 HEAD reference

- 每次 commit 就會變動 reference
- HEAD 指向目前在哪一個 branch
- cat .git/HEAD
- cat .git/refs/heads/master



HEAD

ref: refs/heads/master

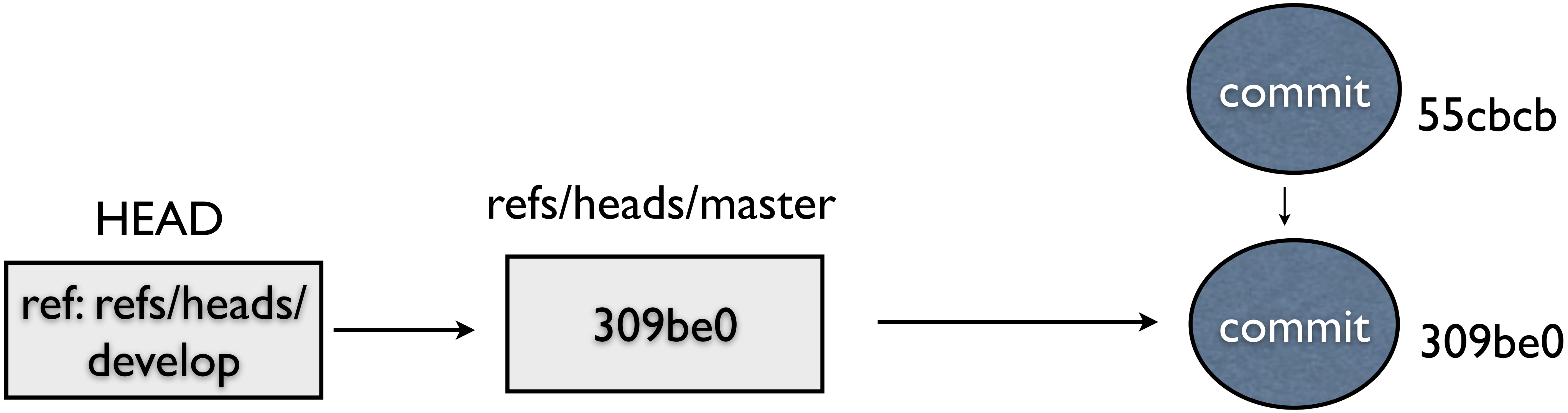
refs/heads/master

309be0

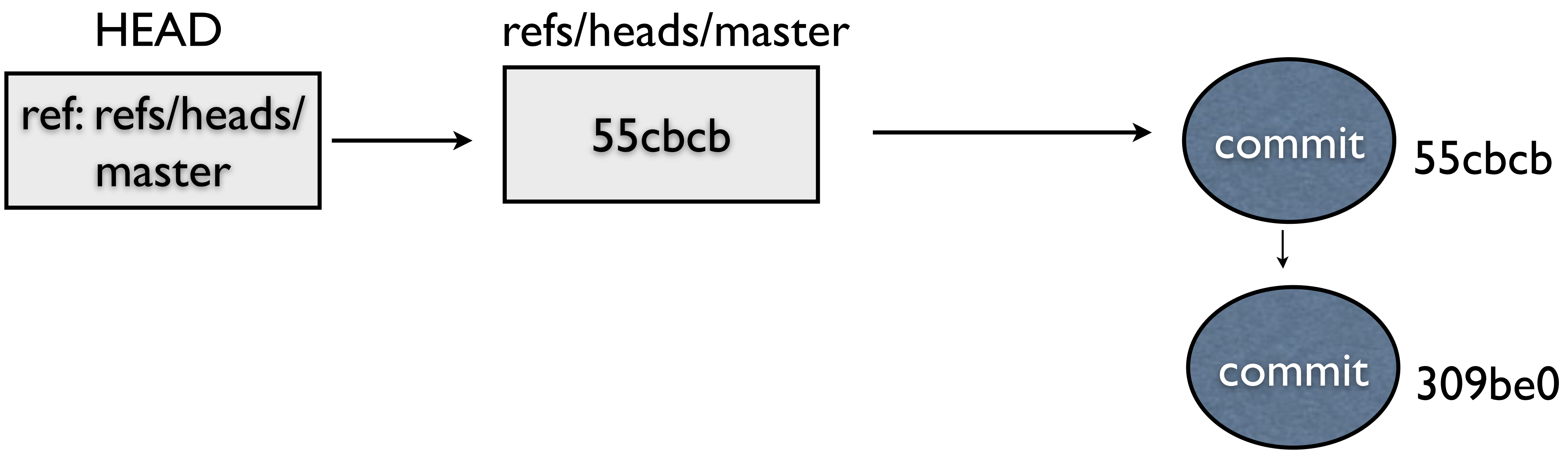
commit

309be0

如果在 Branch 上產生新 Commit...

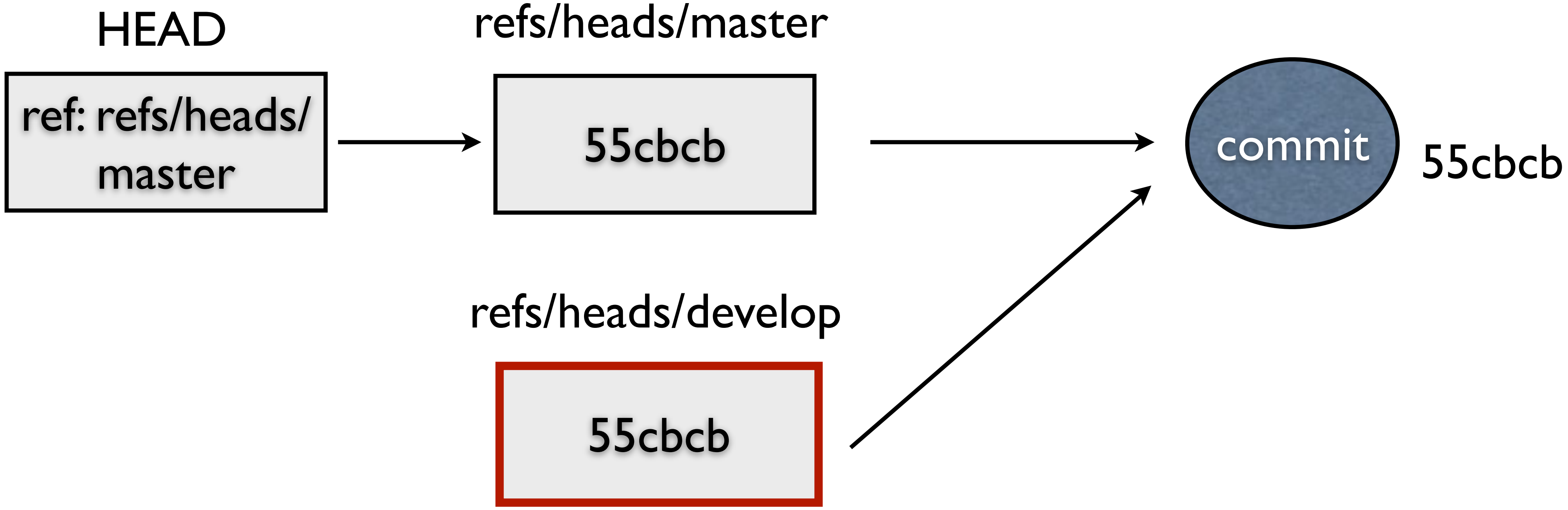


Branch reference 就會自動改指到新的 commit



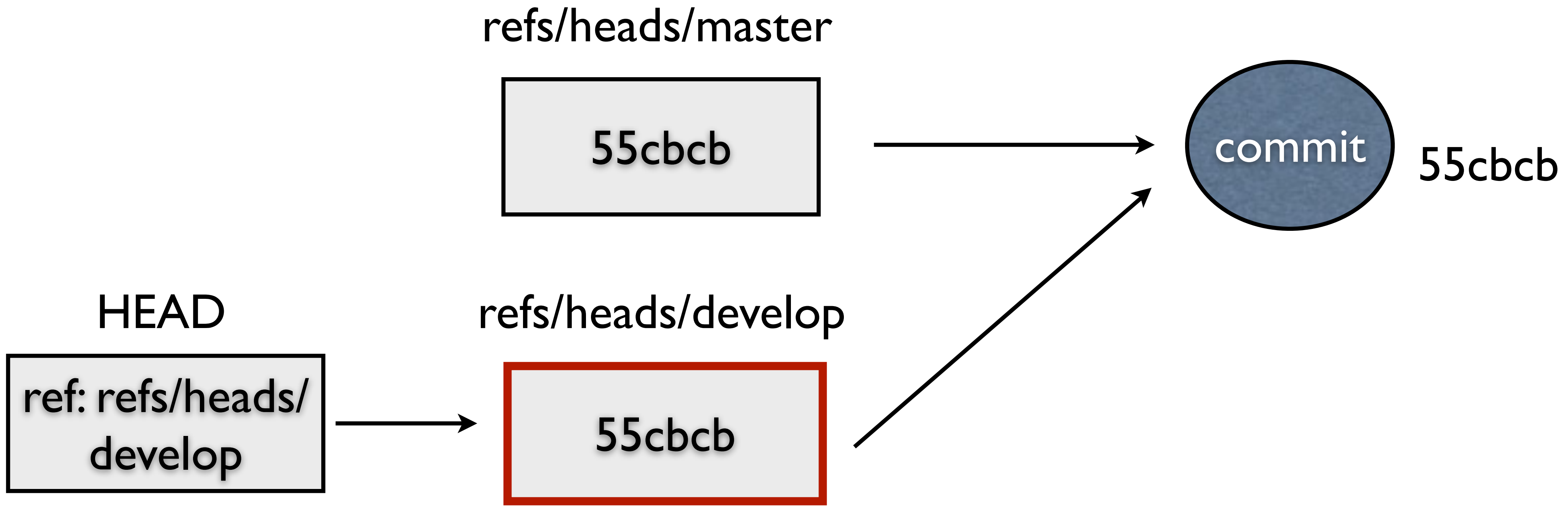
開新 Branch develop

git branch develop



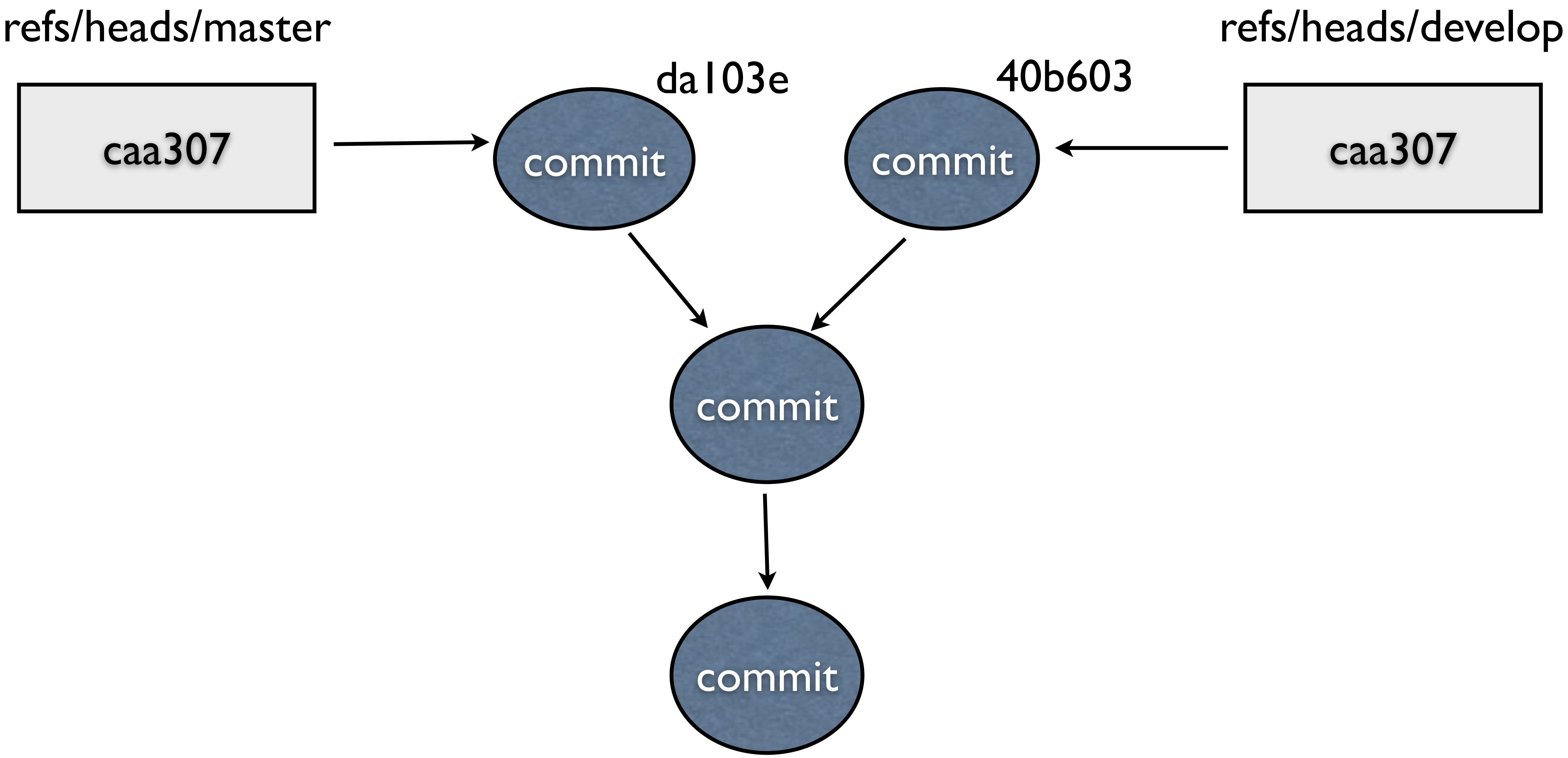
切換 Branch：改HEAD

git checkout develop



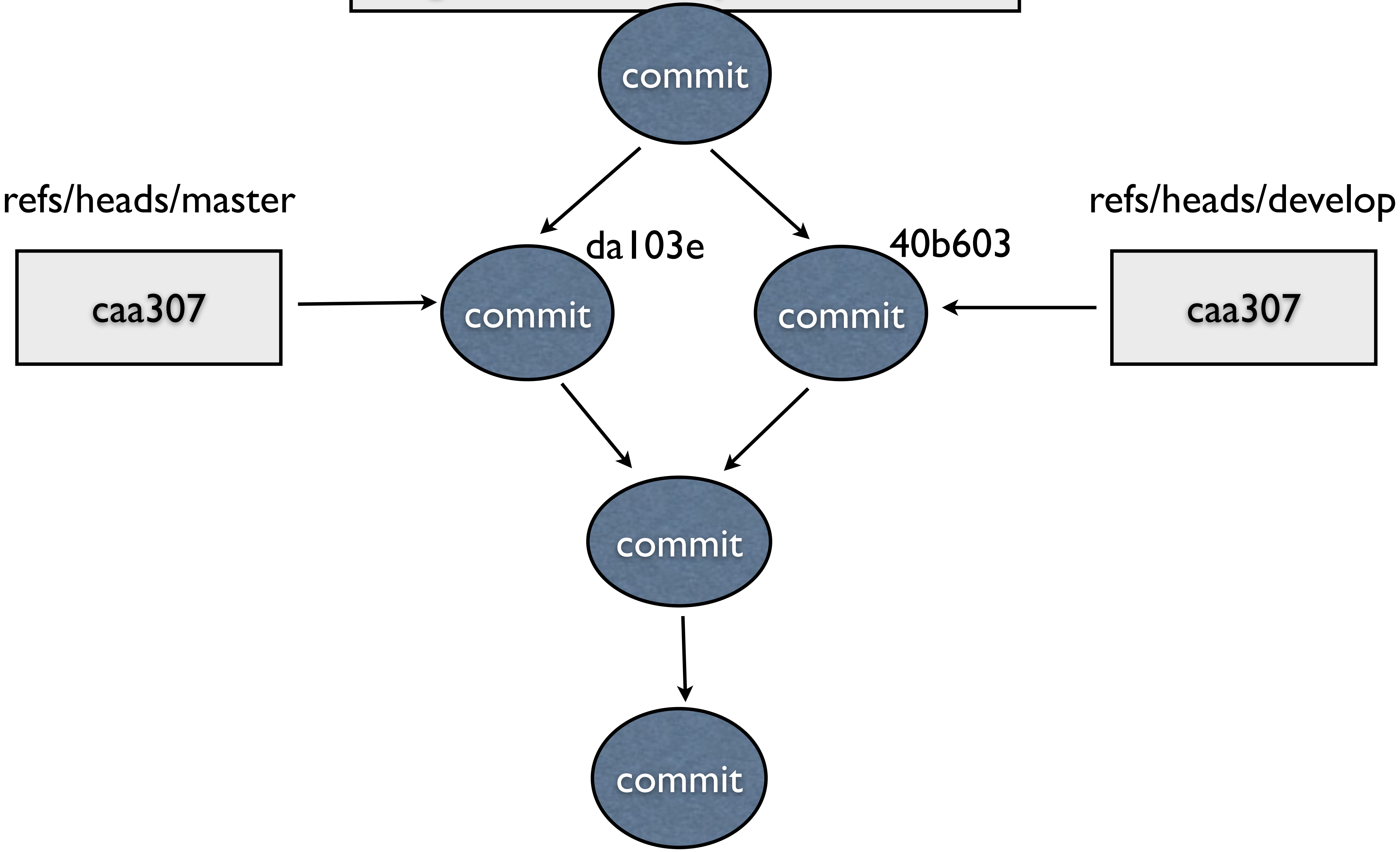
合併 Branch

git merge develop



產生的 merge commit 節點有兩個 parents

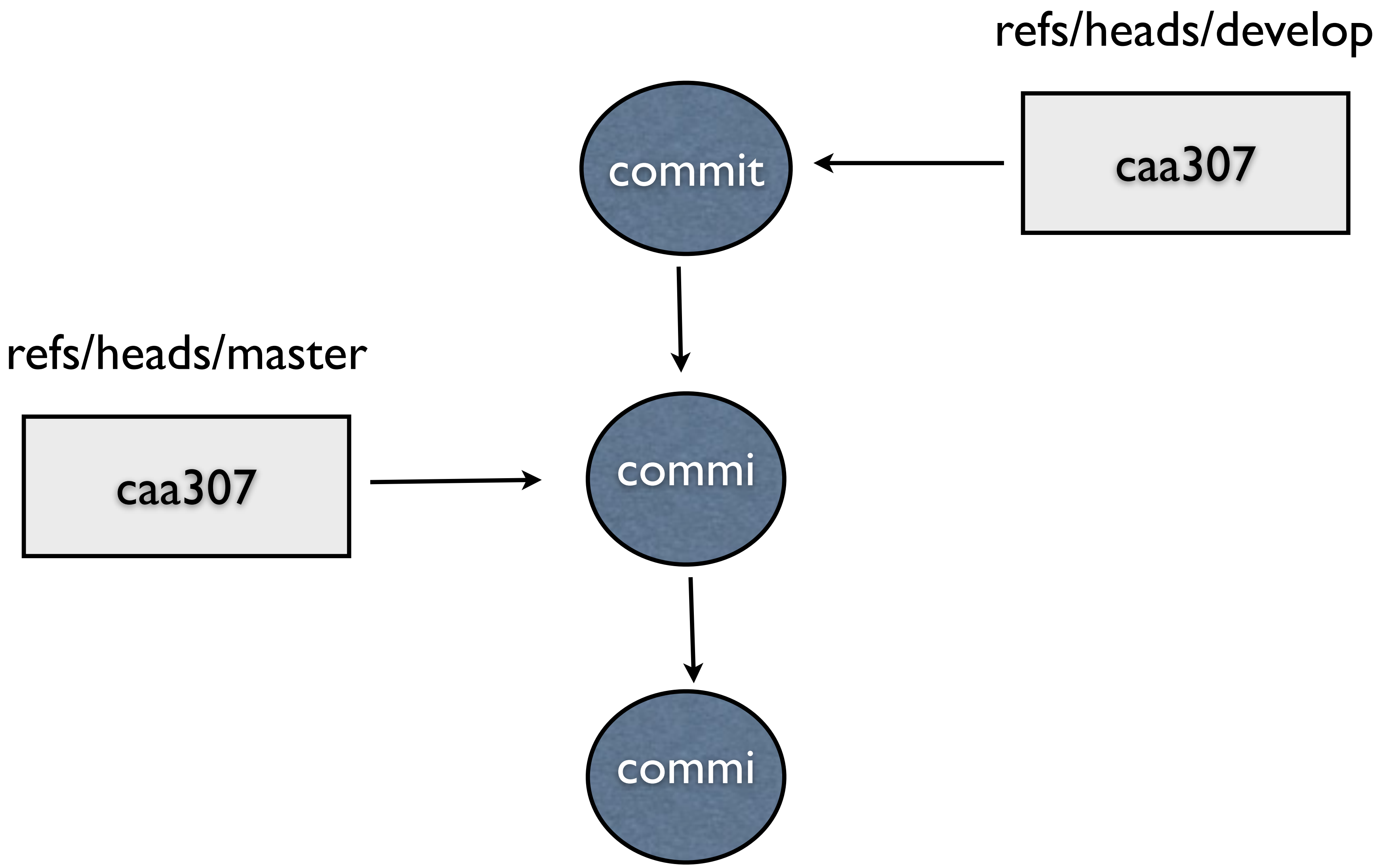
```
tree 5f398a5
parent 40b603
parent da103e
author ihower 1375381779 +0800
committer ihower 1375381779 +0800

Merge branch ‘develop’ into master
```



另一種合併情況 fast-forward

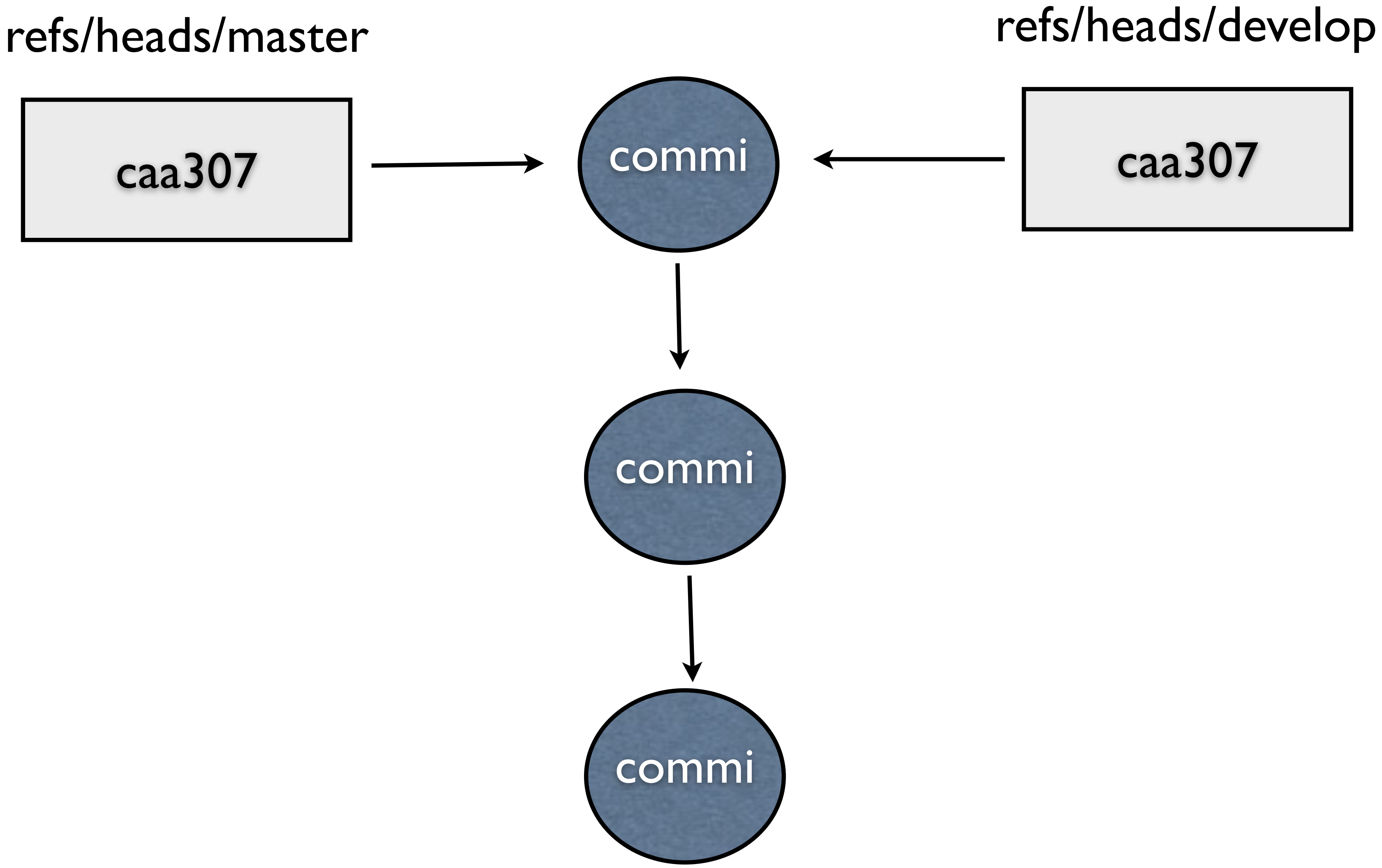
將 develop 合併進 master

(git merge develop)



另一種合併情況 fast-forward

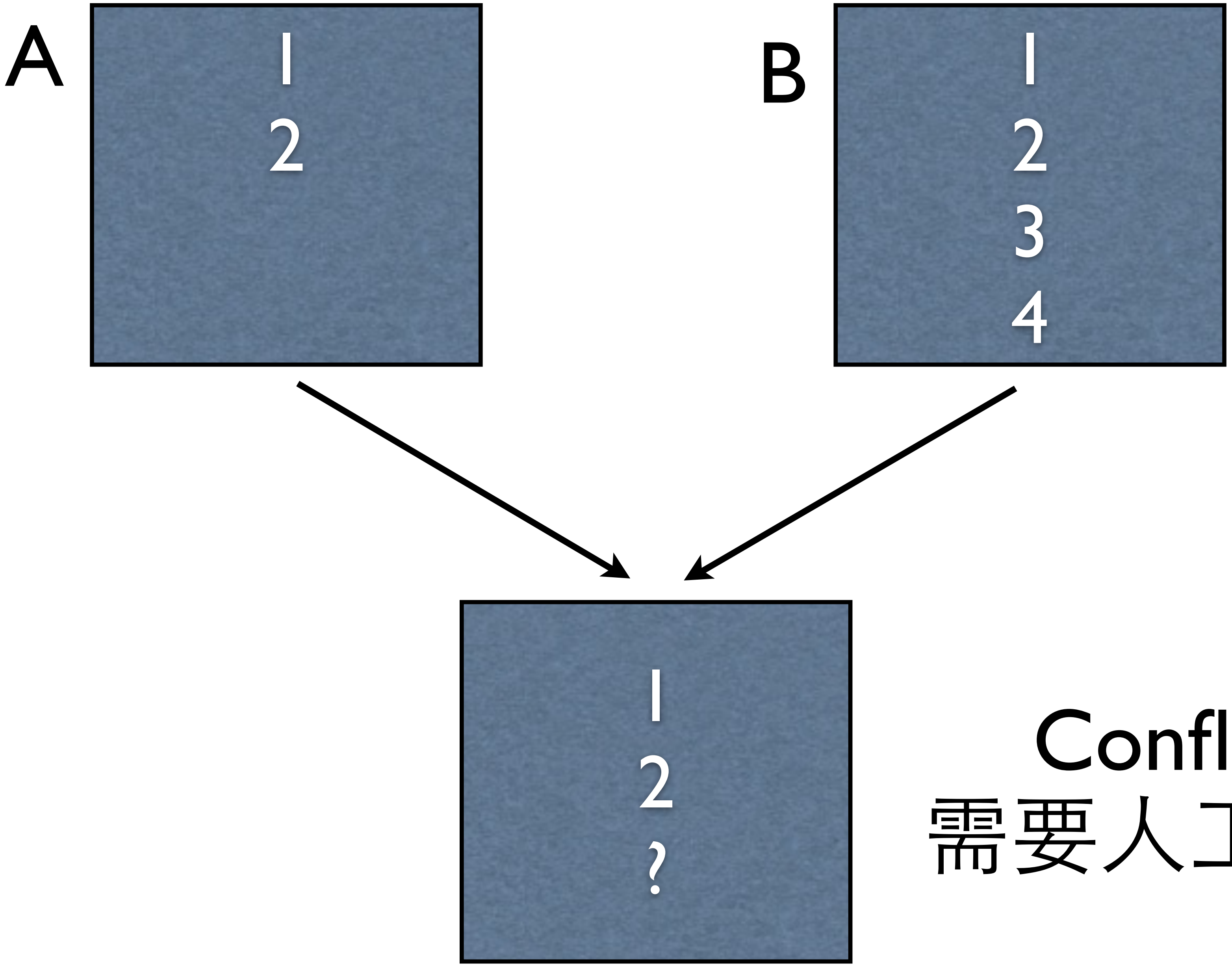
沒有產生 merge 節點，只是移動參考



Git 如何 Merge commits?

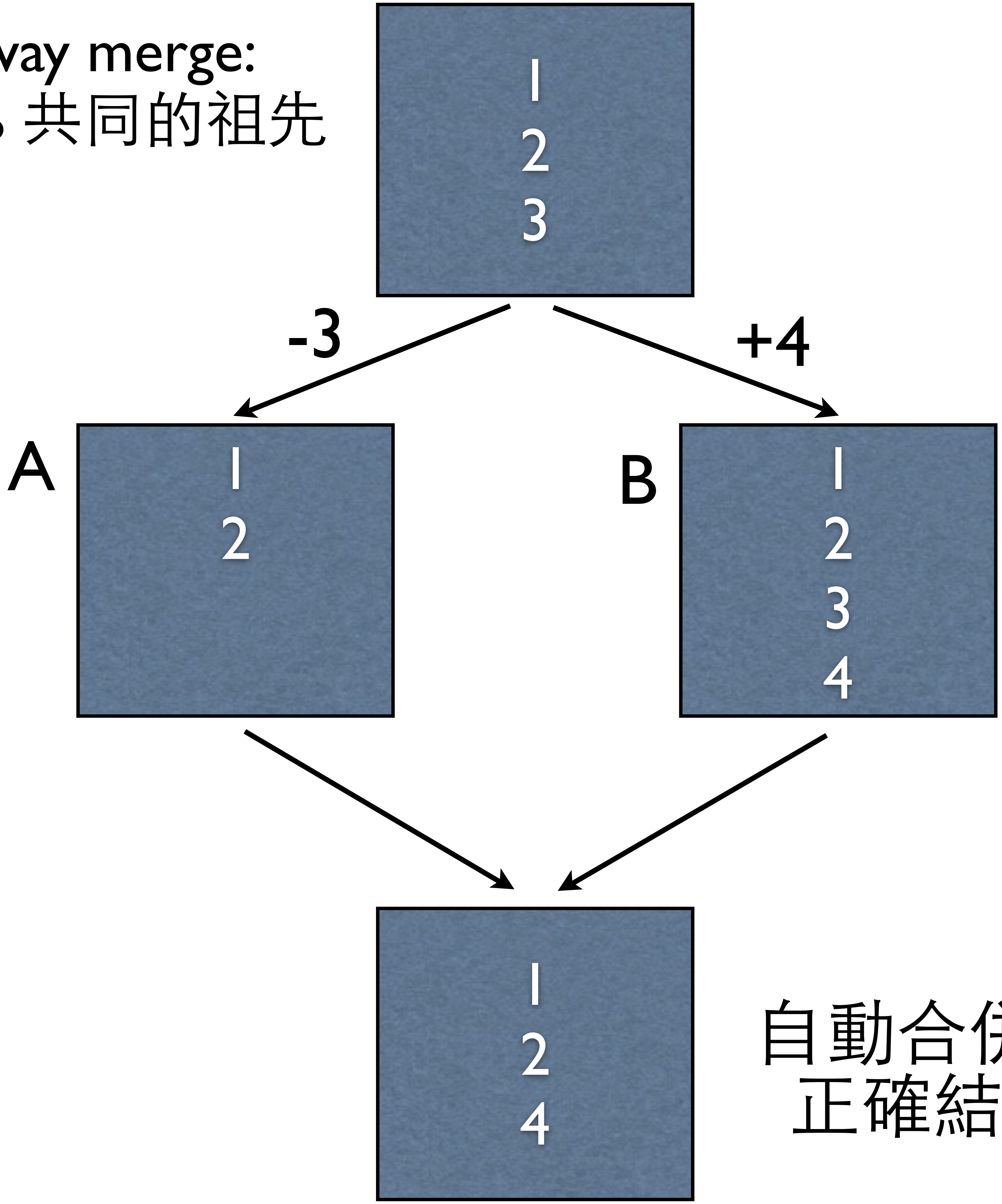
- Git 進行了一個 Three-way merge 的動作
- three-way merge 除了要合併的兩個檔案，還加上兩個檔案的共同祖先。如此可以大大減少人為處理 conflict 的情況。
- two-way merge 則只用兩個檔案進行合併 (svn預設即 two-way merge)

Two-way merge



Conflict!
需要人工判斷

Three-way merge:
先找出 AB 共同的祖先



自動合併出
正確結果

Part1 小結

additive

- 跟 Unix filesystem 有類似的結構，除了
  - Git filesystem 的設計是一直累加的，不會有東西被刪除
  - Blob object 沒有 metadata

Reference is cheap

- 開新 branch 只是 refs 而已，直到 commit 前都沒有負擔。
- 不像有些VCS 開分支會複製一份原始碼，非常耗費資源。

Integrity

- SHA1 是內容的 checksum
- 如果檔案內容有損毀，就會發現跟SHA1不同。如果 tree 被偷改檔名，也會被發現。
- HEAD 指向的 SHA1，就是整個 repository 的 checksum
- 這在分散式系統非常重要：資料從一個開發者傳到另一個開發者時，確保資料沒有被修改。

"I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his code or his data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships."

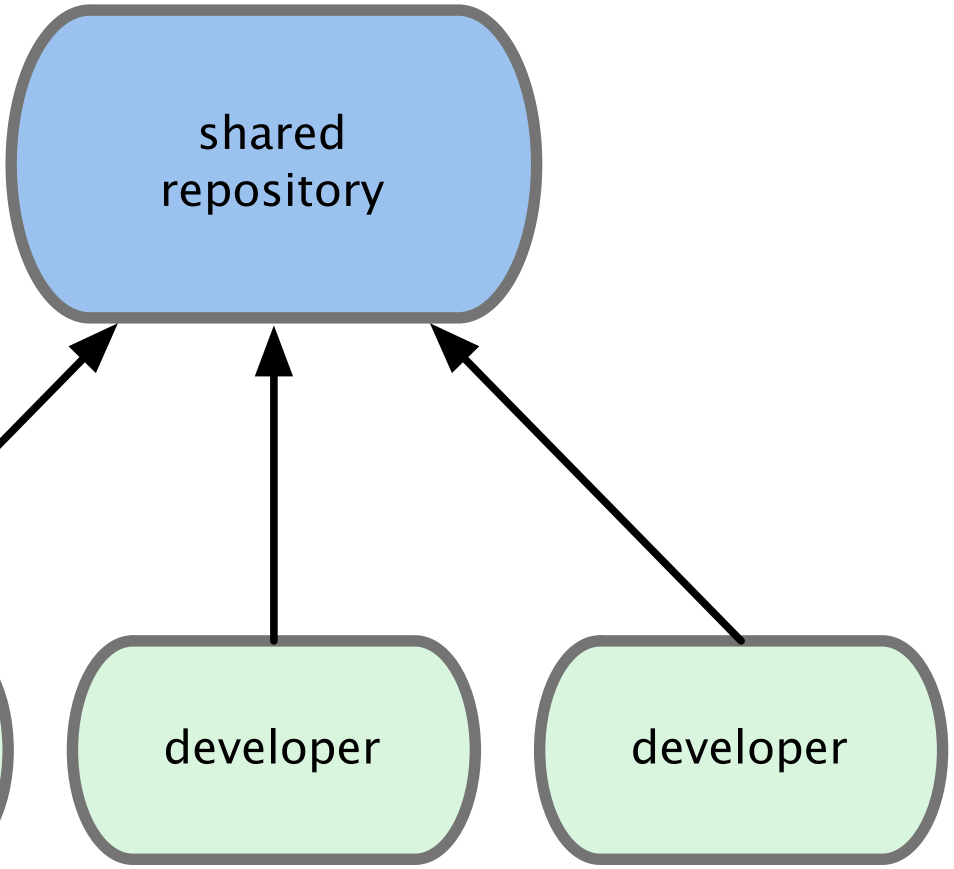
- Linus Torvalds

Part2. Git 的分支開發流程和策略

1. 常見的 Remote Repositories 的管理方式

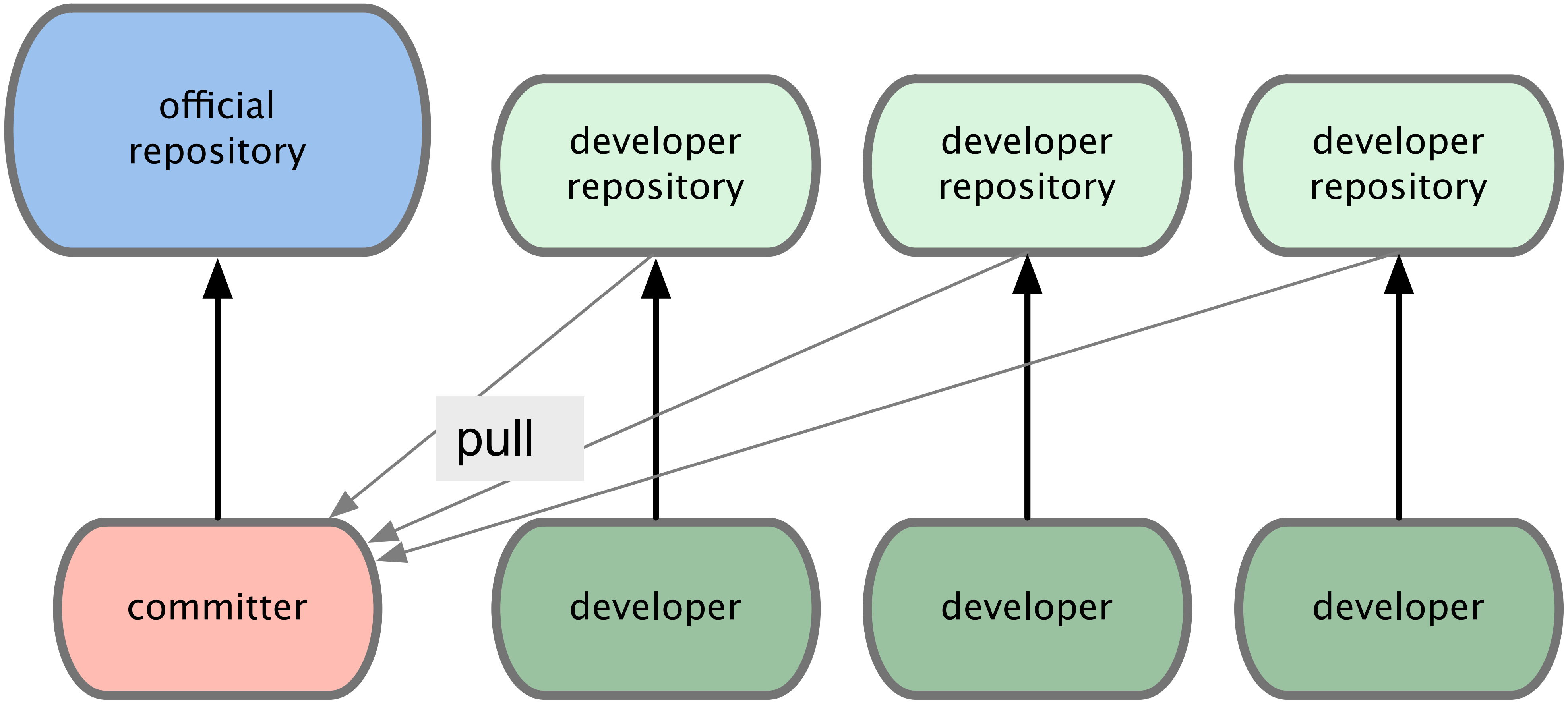
集中式工作流程

團隊內部私有專案，大家都有權限 Push 到共用的 Repository



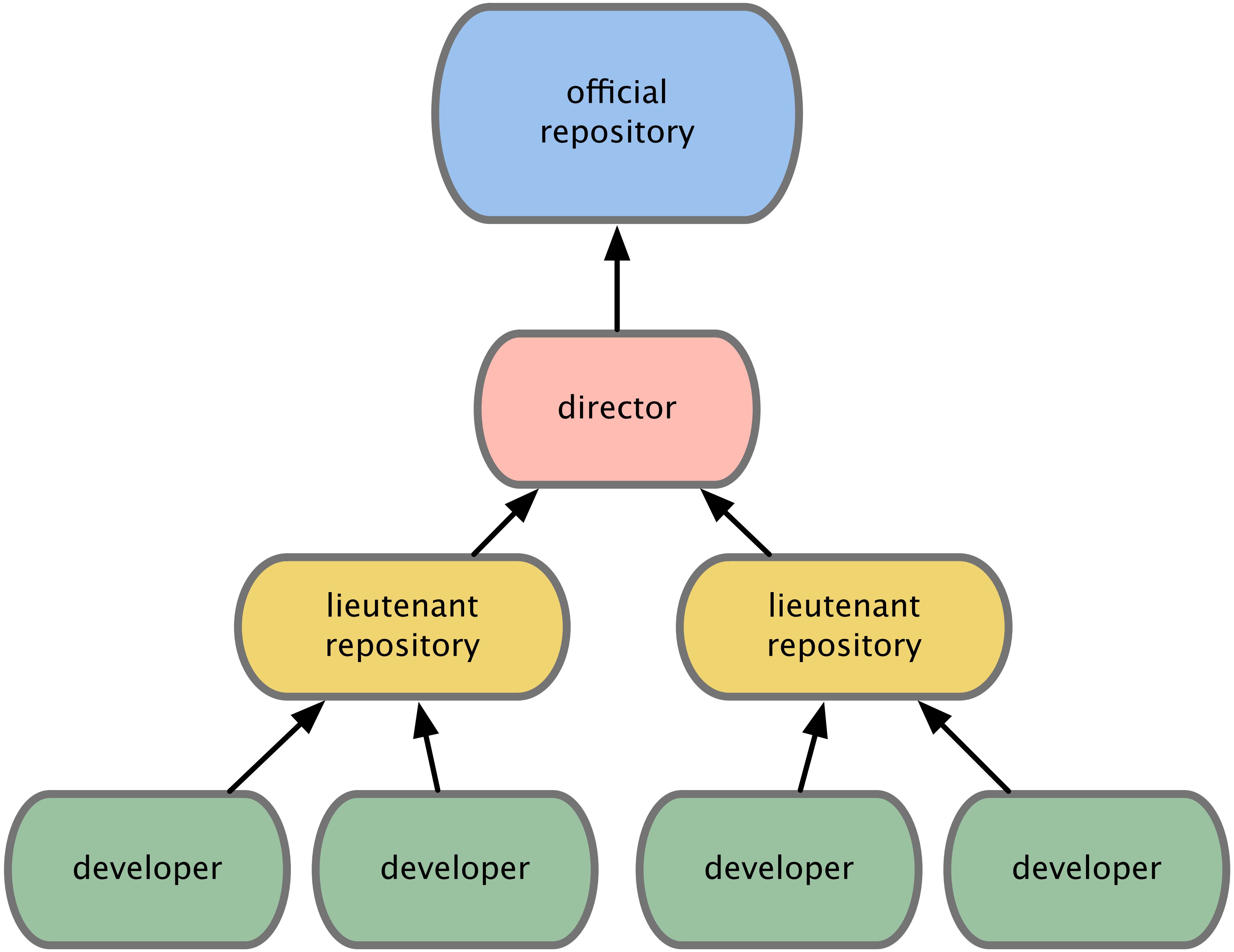
集成管理員工作流程

適合一般 Open Source 專案，只有少部分人有權限可以 Push 到 Repository，其他開發者用 request pull 請求合併。
例如 GitHub 提供的 Fork 和 Pull Request 功能



司令官與副官工作流程

多層權限控管，適合大型 Open Source 專案，例如 Linux Kernel



2. 團隊如何管理 Branches 分支?

Case Study

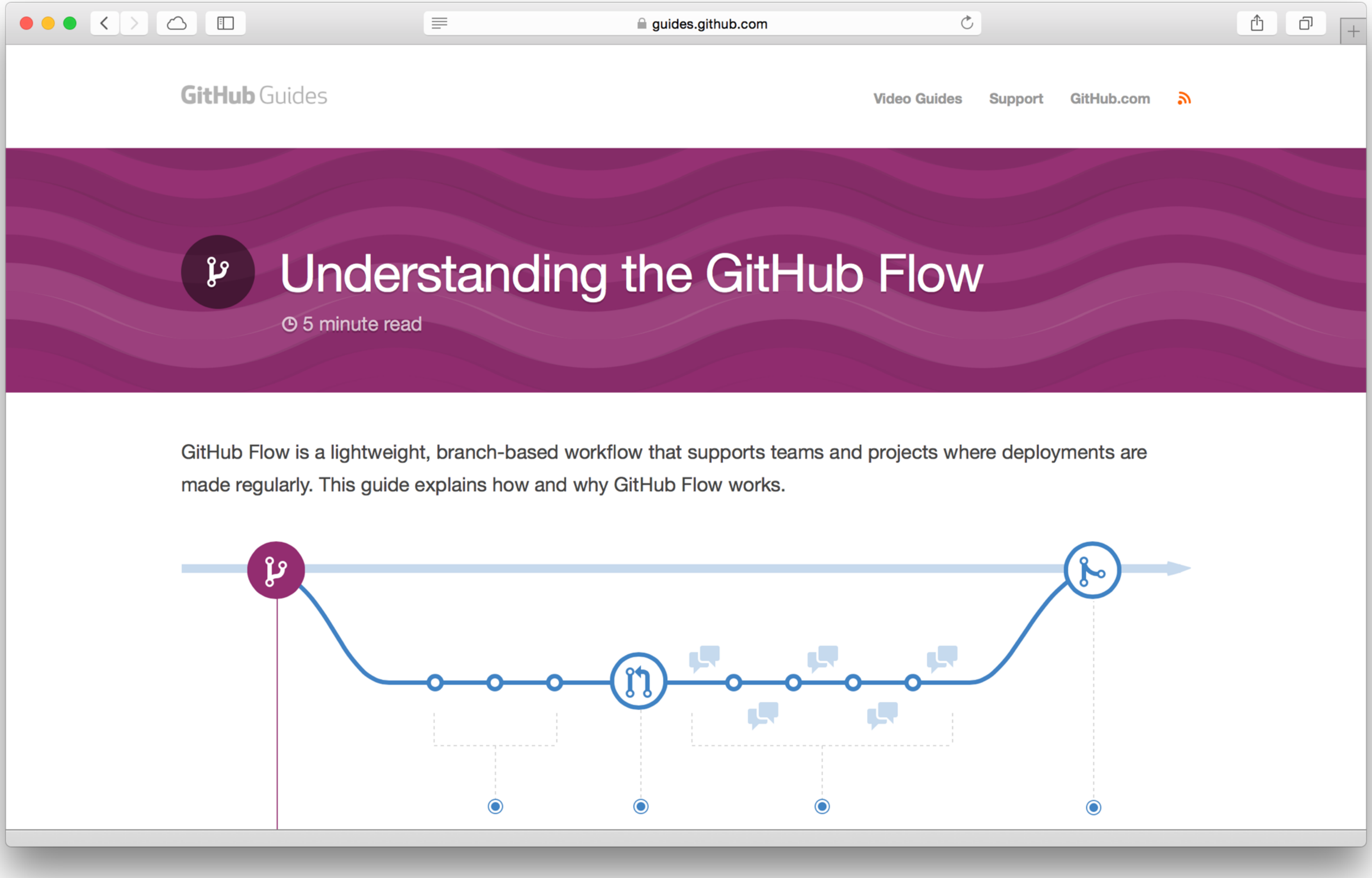
- Github flow
- Rails
- CMake
- Homebrew
- Gitflow

Github flow

- master 是 stable/production 可佈署的版本
- 任何開發從 master branch 分支出 feature branch
- 送 pull request 開始進行討論、code review 和測試
- 最後合併回 master 代表可以佈署了

http://ihower.tw/blog/archives/7798

https://guides.github.com/introduction/flow/index.html



Understanding the GitHub Flow

5 minute read

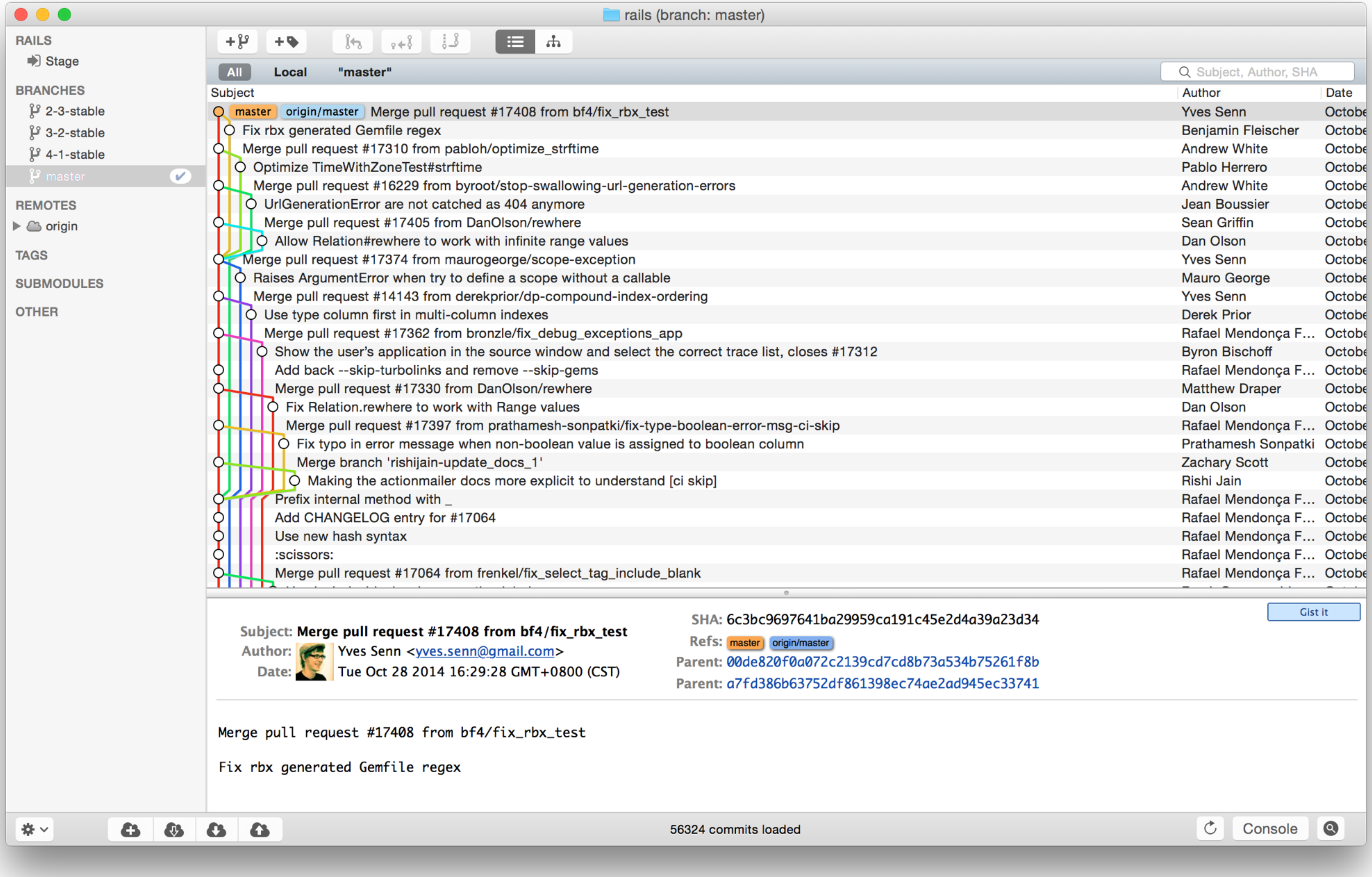
GitHub Flow is a lightweight, branch-based workflow that supports teams and projects where deployments are made regularly. This guide explains how and why GitHub Flow works.

pros and cons

- 簡單、清楚、容易了解
- 搭配 Github 的 Pull Request 介面
- 沒有 release branch，東西一進 master 就上 production
  - Web app 如果有問題很容易 revert，如果是 desktop/mobile 軟體就 GG 了

Ruby on Rails

- Rails 目前的版本是 4.1
- master 是開發版本，開發下一次的主要版本 4.2
- feature branches 審核完後，合併進 master
- maintenance branches，用 cherry-pick 做 backporting
  - 2-3-stable 已停止維護
  - 3-2-stable security fixes
  - 4-0-stable bugs fixes
  - 4-1-stable bugs fixes



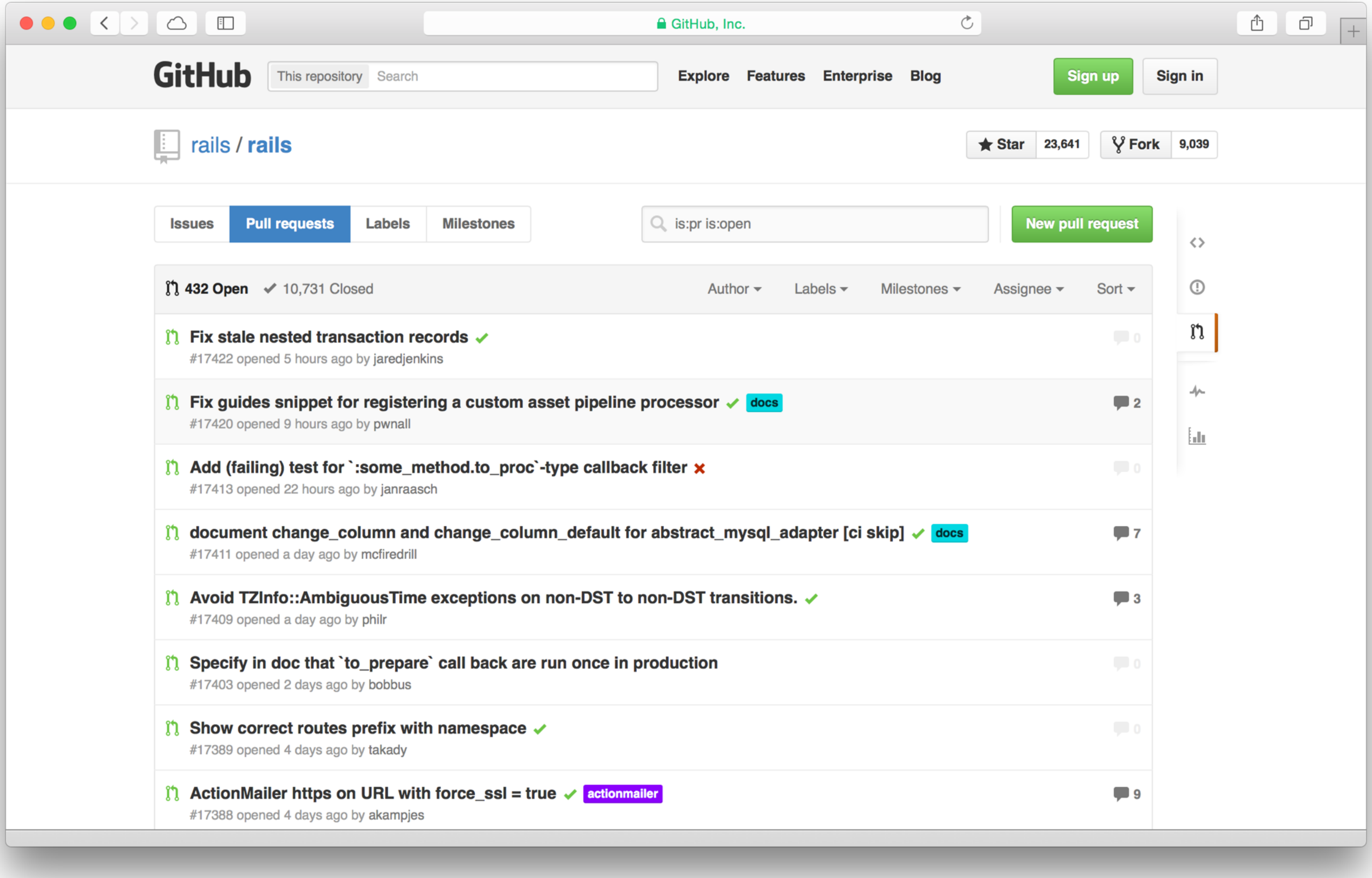
Subject	Author	Date
master origin/master Merge pull request #17408 from bf4/fix_rbx_test	Yves Senn	Octobe
Fix rbx generated Gemfile regex	Benjamin Fleischer	Octobe
Merge pull request #17310 from pabloh/optimize_strftime	Andrew White	Octobe
Optimize TimeWithZoneTest#strftime	Pablo Herrero	Octobe
Merge pull request #16229 from byroot/stop-swallowing-url-generation-errors	Andrew White	Octobe
UrlGenerationError are not catched as 404 anymore	Jean Boussier	Octobe
Merge pull request #17405 from DanOlson/rewhere	Sean Griffin	Octobe
Allow Relation#rewhere to work with infinite range values	Dan Olson	Octobe
Merge pull request #17374 from maurogeorge/scope-exception	Yves Senn	Octobe
Raises ArgumentError when try to define a scope without a callable	Mauro George	Octobe
Merge pull request #14143 from derekprior/dp-compound-index-ordering	Yves Senn	Octobe
Use type column first in multi-column indexes	Derek Prior	Octobe
Merge pull request #17362 from bronzle/fix_debug_exceptions_app	Rafael Mendonça F…	Octobe
Show the user's application in the source window and select the correct trace list, closes #17312	Byron Bischoff	Octobe
Add back --skip-turbolinks and remove --skip-gems	Rafael Mendonça F…	Octobe
Merge pull request #17330 from DanOlson/rewhere	Matthew Draper	Octobe
Fix Relation.rewhere to work with Range values	Dan Olson	Octobe
Merge pull request #17397 from prathamesh-sonpatki/fix-type-boolean-error-msg-ci-skip	Rafael Mendonça F…	Octobe
Fix typo in error message when non-boolean value is assigned to boolean column	Prathamesh Sonpatki	Octobe
Merge branch 'rishijain-update_docs_1'	Zachary Scott	Octobe
Making the actionmailer docs more explicit to understand [ci skip]	Rishi Jain	Octobe
Prefix internal method with _	Rafael Mendonça F…	Octobe
Add CHANGELOG entry for #17064	Rafael Mendonça F…	Octobe
Use new hash syntax	Rafael Mendonça F…	Octobe
:scissors:	Rafael Mendonça F…	Octobe
Merge pull request #17064 from frenkel/fix_select_tag_include_blank	Rafael Mendonça F…	Octobe

Subject: **Merge pull request #17408 from bf4/fix_rbx_test**
Author: Yves Senn <yves.senn@gmail.com>
Date: Tue Oct 28 2014 16:29:28 GMT+0800 (CST)

SHA: 6c3bc9697641ba29959ca191c45e2d4a39a23d34
Refs: master origin/master
Parent: 00de820f0a072c2139cd7cd8b73a534b75261f8b
Parent: a7fd386b63752df861398ec74ae2ad945ec33741

```
Merge pull request #17408 from bf4/fix_rbx_test

Fix rbx generated Gemfile regex
```

56324 commits loaded



Fix stale nested transaction records ✓ 0
#17422 opened 5 hours ago by jaredjenkins

Fix guides snippet for registering a custom asset pipeline processor ✓ docs 2
#17420 opened 9 hours ago by pwnall

Add (failing) test for `:some_method.to_proc`-type callback filter ✗ 0
#17413 opened 22 hours ago by janraasch

document change_column and change_column_default for abstract_mysql_adapter [ci skip] ✓ docs 7
#17411 opened a day ago by mcfiredrill

Avoid TZInfo::AmbiguousTime exceptions on non-DST to non-DST transitions. ✓ 3
#17409 opened a day ago by philr

Specify in doc that `to_prepare` call back are run once in production 0
#17403 opened 2 days ago by bobbus

Show correct routes prefix with namespace ✓ 0
#17389 opened 4 days ago by takady

ActionMailer https on URL with force_ssl = true ✓ actionmailer 9
#17388 opened 4 days ago by akampjes

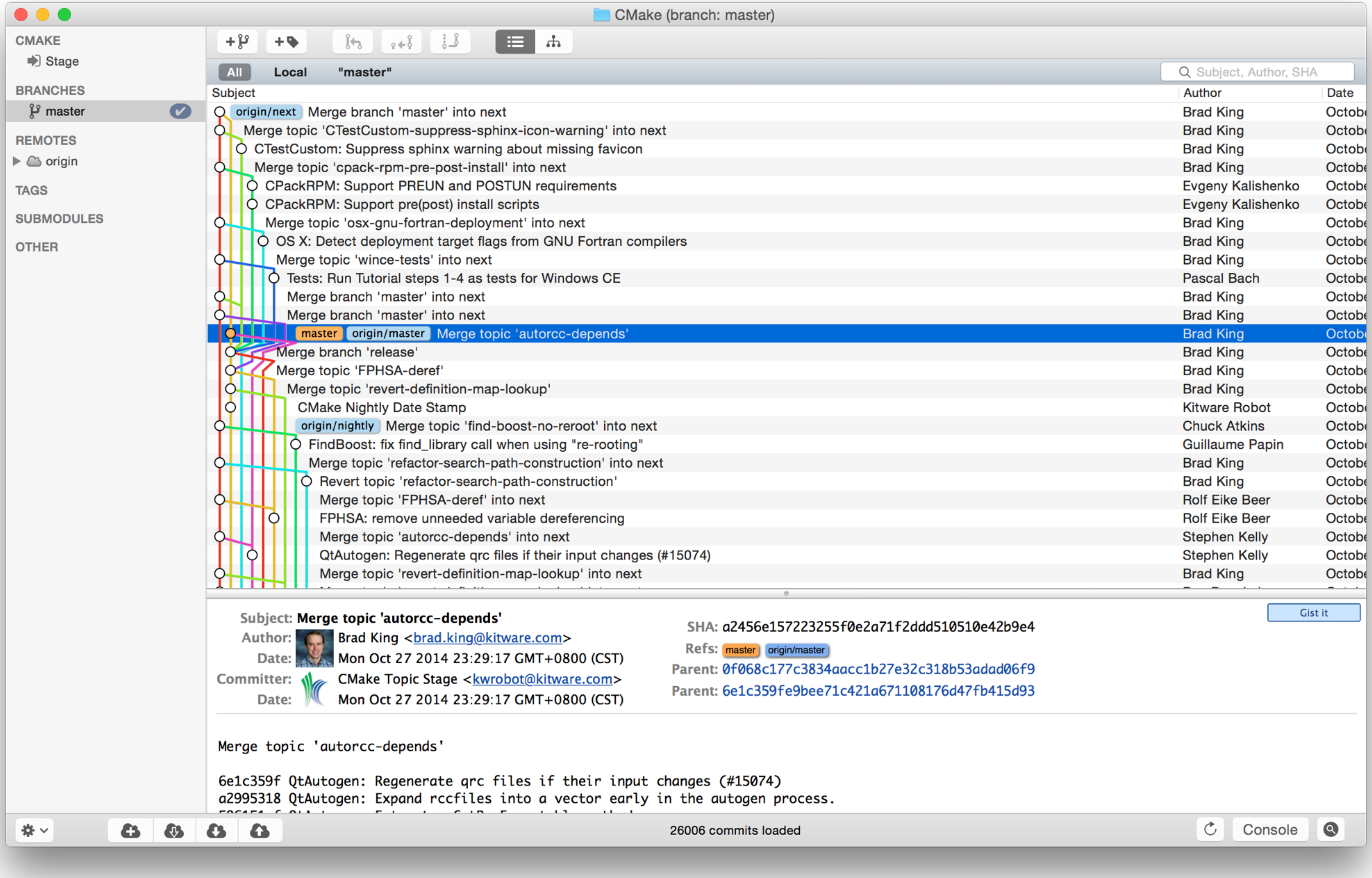
pros and cons

- 基本上就是 Github flow 加上 maintenance branches 維護舊版的設計
- 非常多的 Open Source 專案採用這種形式
- 版本號(Tag)打在 master 上，透過 preview 和 beta 的版本號進行提前釋出

CMake

http://public.kitware.com/Wiki/Git/Workflow/Topic

- master 預備釋出的版本，feature branches 從這裡分支出去
- feature branch 完成後，合併進 <u>next</u>
- next 整合版本，完成的 feature branch 先合併到這裡進行測試
  - 在 next 測好的 feature branch，才合併進 master
  - 可以將 master 合併進 next，減少之後的 code conflicts
  - 不會將 next 合併進 master
- nightly 每天 1:00 UTC 自動從 next branch 分支出來跑自動測試



Subject	Author	Date
origin/next Merge branch 'master' into next	Brad King	Octob
Merge topic 'CTestCustom-suppress-sphinx-icon-warning' into next	Brad King	Octob
CTestCustom: Suppress sphinx warning about missing favicon	Brad King	Octob
Merge topic 'cpack-rpm-pre-post-install' into next	Brad King	Octob
CPackRPM: Support PREUN and POSTUN requirements	Evgeny Kalishenko	Octob
CPackRPM: Support pre(post) install scripts	Evgeny Kalishenko	Octob
Merge topic 'osx-gnu-fortran-deployment' into next	Brad King	Octob
OS X: Detect deployment target flags from GNU Fortran compilers	Brad King	Octob
Merge topic 'wince-tests' into next	Brad King	Octob
Tests: Run Tutorial steps 1-4 as tests for Windows CE	Pascal Bach	Octob
Merge branch 'master' into next	Brad King	Octob
Merge branch 'master' into next	Brad King	Octob
master origin/master Merge topic 'autorcc-depends'	Brad King	Octob
Merge branch 'release'	Brad King	Octob
Merge topic 'FPHSA-deref'	Brad King	Octob
Merge topic 'revert-definition-map-lookup'	Brad King	Octob
CMake Nightly Date Stamp	Kitware Robot	Octob
origin/nightly Merge topic 'find-boost-no-reroot' into next	Chuck Atkins	Octob
FindBoost: fix find_library call when using "re-rooting"	Guillaume Papin	Octob
Merge topic 'refactor-search-path-construction' into next	Brad King	Octob
Revert topic 'refactor-search-path-construction'	Brad King	Octob
Merge topic 'FPHSA-deref' into next	Rolf Eike Beer	Octob
FPHSA: remove unneeded variable dereferencing	Rolf Eike Beer	Octob
Merge topic 'autorcc-depends' into next	Stephen Kelly	Octob
QtAutogen: Regenerate qrc files if their input changes (#15074)	Stephen Kelly	Octob
Merge topic 'revert-definition-map-lookup' into next	Brad King	Octob

Subject: Merge topic 'autorcc-depends'
Author: Brad King <brad.king@kitware.com>
Date: Mon Oct 27 2014 23:29:17 GMT+0800 (CST)
Committer: CMake Topic Stage <kwrobot@kitware.com>
Date: Mon Oct 27 2014 23:29:17 GMT+0800 (CST)

SHA: a2456e157223255f0e2a71f2ddd510510e42b9e4
Refs: master origin/master
Parent: 0f068c177c3834aacc1b27e32c318b53adad06f9
Parent: 6e1c359fe9bee71c421a671108176d47fb415d93

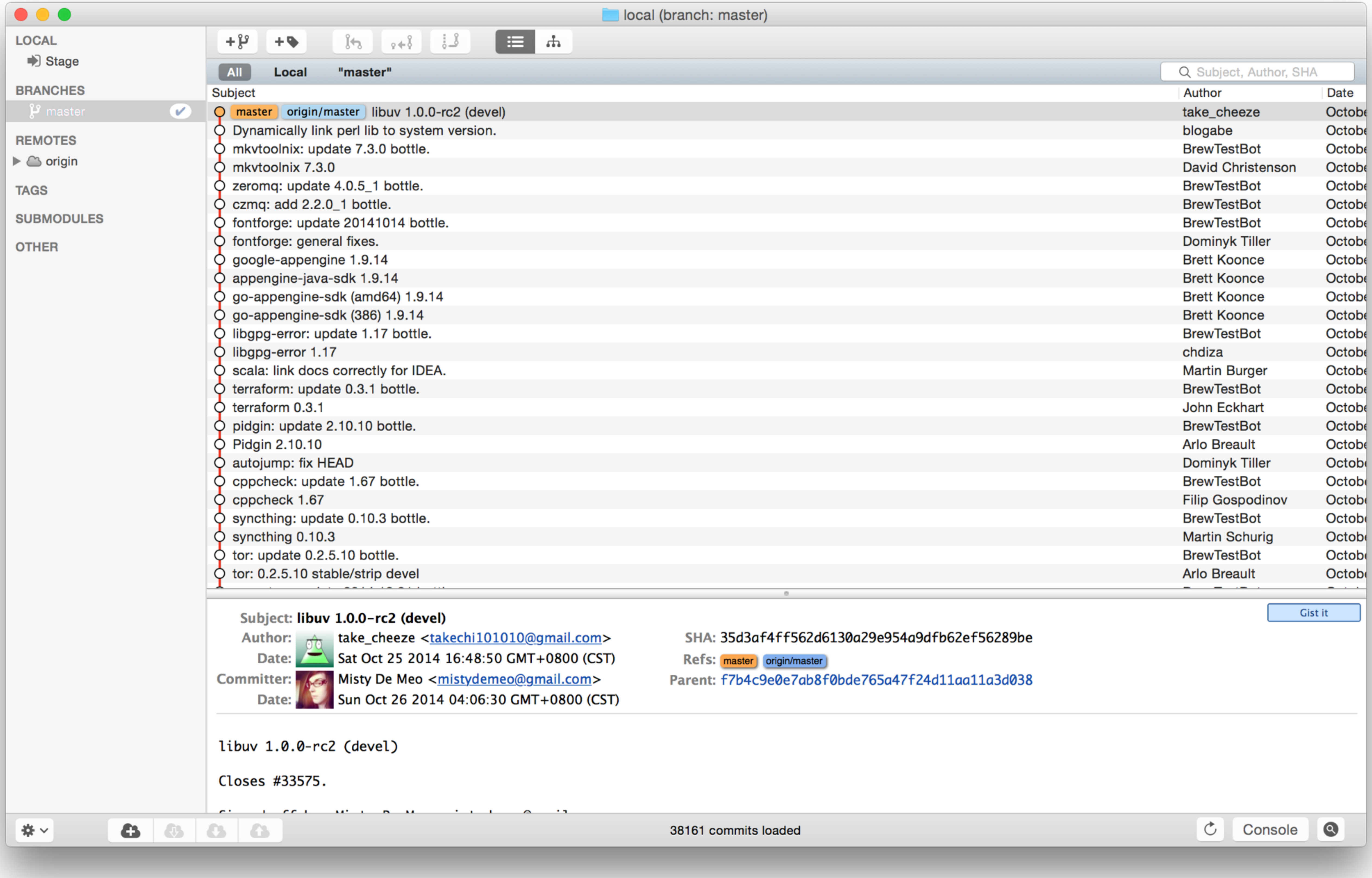
```
Merge topic 'autorcc-depends'

6e1c359f QtAutogen: Regenerate qrc files if their input changes (#15074)
a2995318 QtAutogen: Expand rccfiles into a vector early in the autogen process.
```

pros and cons

- 個別 feature branch 的狀態很清楚
- 需要發行 binary release 的軟體需要較正式、較花時間的釋出過程(feature branch -> next -> master -> release branch)。來確保釋出的軟體有足夠的測試。
- 但是線圖其實很不好追蹤，很多 merge commits 跟真正的 commits 混雜在一起
- Revert a merge commit 其實不好做

Homebrew

- 只有 master branch，沒有 merge commit
- 所有 feature branch 一律 rebase 和 squashed 後進主幹



Subject	Author	Date
master origin/master libuv 1.0.0-rc2 (devel)	take_cheeze	Octobe
Dynamically link perl lib to system version.	blogabe	Octobe
mkvtoolnix: update 7.3.0 bottle.	BrewTestBot	Octobe
mkvtoolnix 7.3.0	David Christenson	Octobe
zeromq: update 4.0.5_1 bottle.	BrewTestBot	Octobe
czmq: add 2.2.0_1 bottle.	BrewTestBot	Octobe
fontforge: update 20141014 bottle.	BrewTestBot	Octobe
fontforge: general fixes.	Dominyk Tiller	Octobe
google-appengine 1.9.14	Brett Koonce	Octobe
appengine-java-sdk 1.9.14	Brett Koonce	Octobe
go-appengine-sdk (amd64) 1.9.14	Brett Koonce	Octobe
go-appengine-sdk (386) 1.9.14	Brett Koonce	Octobe
libgpg-error: update 1.17 bottle.	BrewTestBot	Octobe
libgpg-error 1.17	chdiza	Octobe
scala: link docs correctly for IDEA.	Martin Burger	Octobe
terraform: update 0.3.1 bottle.	BrewTestBot	Octobe
terraform 0.3.1	John Eckhart	Octobe
pidgin: update 2.10.10 bottle.	BrewTestBot	Octobe
Pidgin 2.10.10	Arlo Breault	Octob
autojump: fix HEAD	Dominyk Tiller	Octob
cppcheck: update 1.67 bottle.	BrewTestBot	Octob
cppcheck 1.67	Filip Gospodinov	Octob
syncthing: update 0.10.3 bottle.	BrewTestBot	Octob
syncthing 0.10.3	Martin Schurig	Octob
tor: update 0.2.5.10 bottle.	BrewTestBot	Octob
tor: 0.2.5.10 stable/strip devel	Arlo Breault	Octob

Subject: **libuv 1.0.0-rc2 (devel)**
Author: take_cheeze <takechi101010@gmail.com>
Date: Sat Oct 25 2014 16:48:50 GMT+0800 (CST)
Committer: Misty De Meo <mistydemeo@gmail.com>
Date: Sun Oct 26 2014 04:06:30 GMT+0800 (CST)
SHA: 35d3af4ff562d6130a29e954a9dfb62ef56289be
Refs: master origin/master
Parent: f7b4c9e0e7ab8f0bde765a47f24d11aa11a3d038

```
libuv 1.0.0-rc2 (devel)

Closes #33575.
```

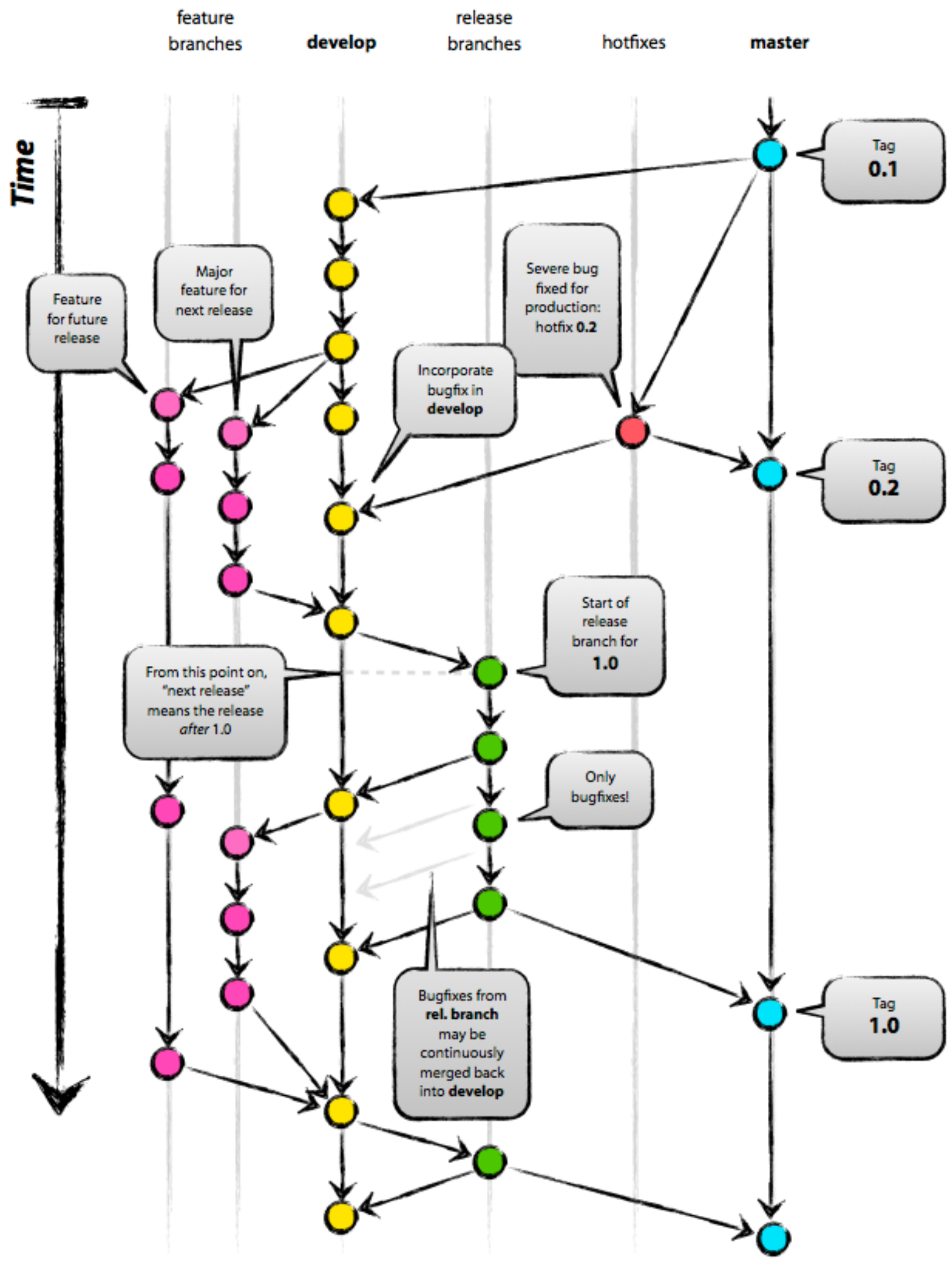
38161 commits loaded

pros and cons

- Homebrew 不只把 Git 當 VCS，甚至當成 delivery 部署機制
- 歷史紀錄超級 Readable 容易讀
- 沒有 release branch，釋出很快。有 bug 也很容易 revert、re-apply 或 bisect
- 合併前的 branch commits 訊息丟失

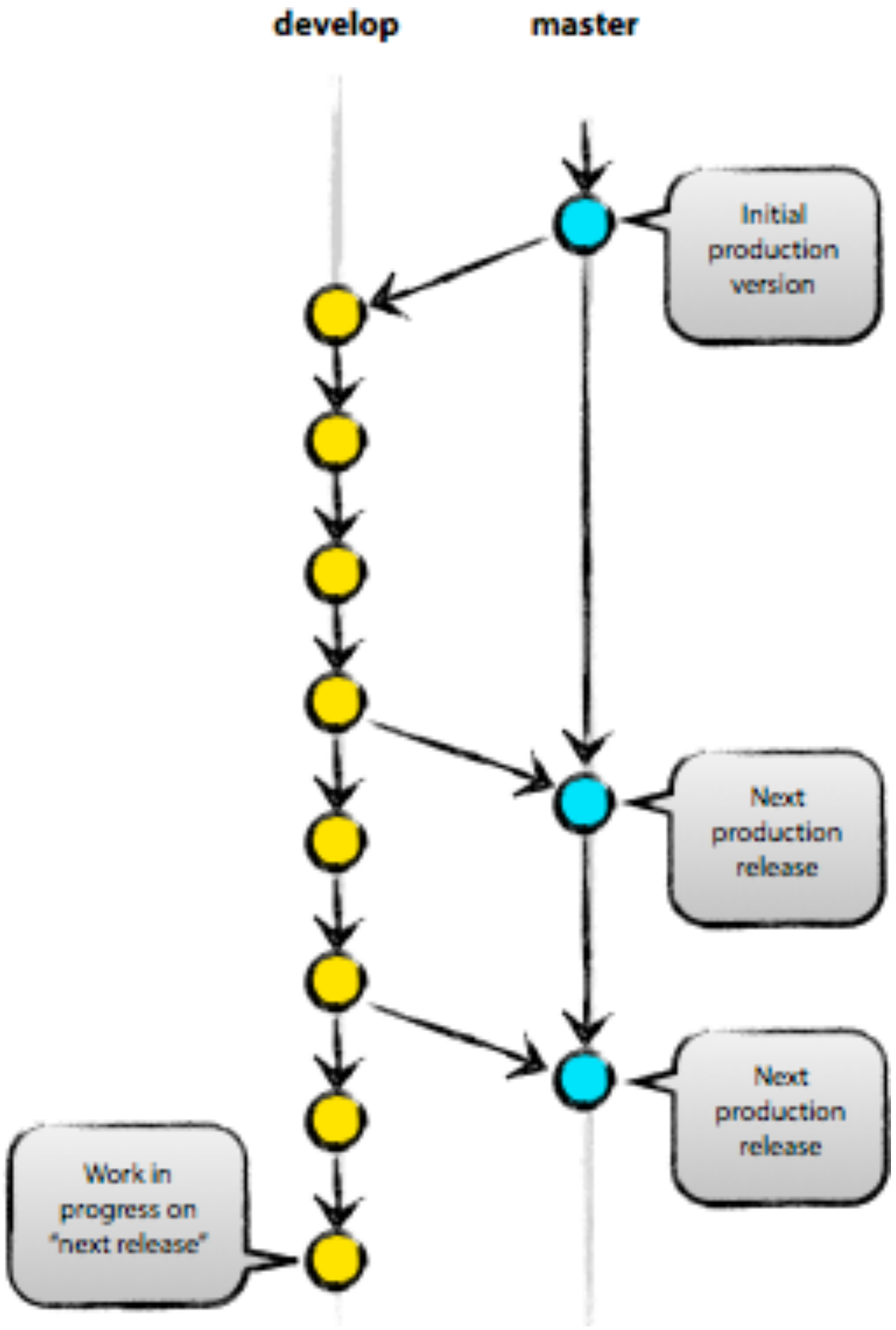
Git flow

http://nvie.com/posts/a-successful-git-branching-model/


feature branches
develop
release branches
hotfixes
master
Time
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release after 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

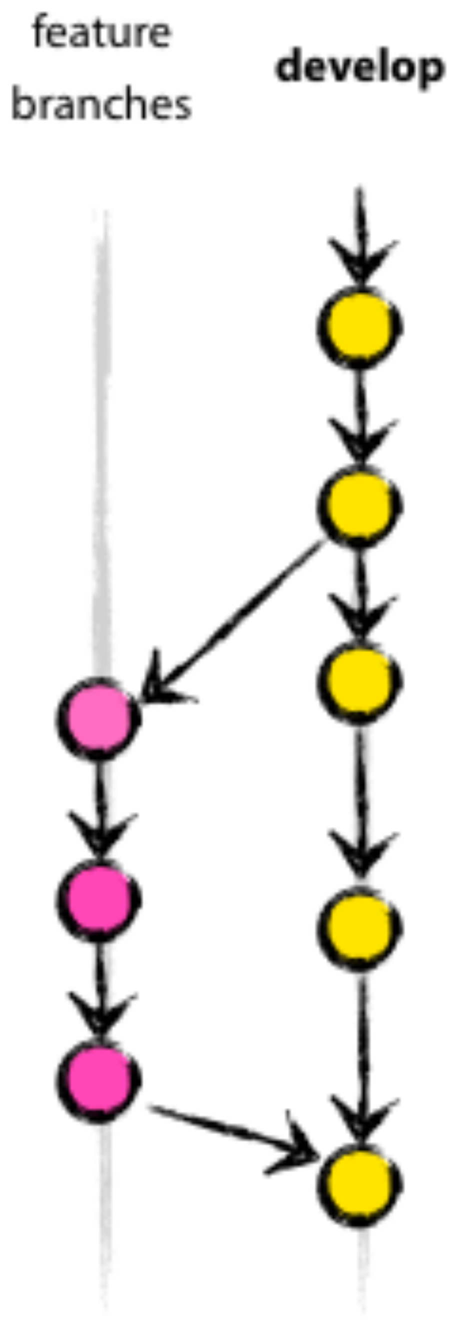
兩個主要分支

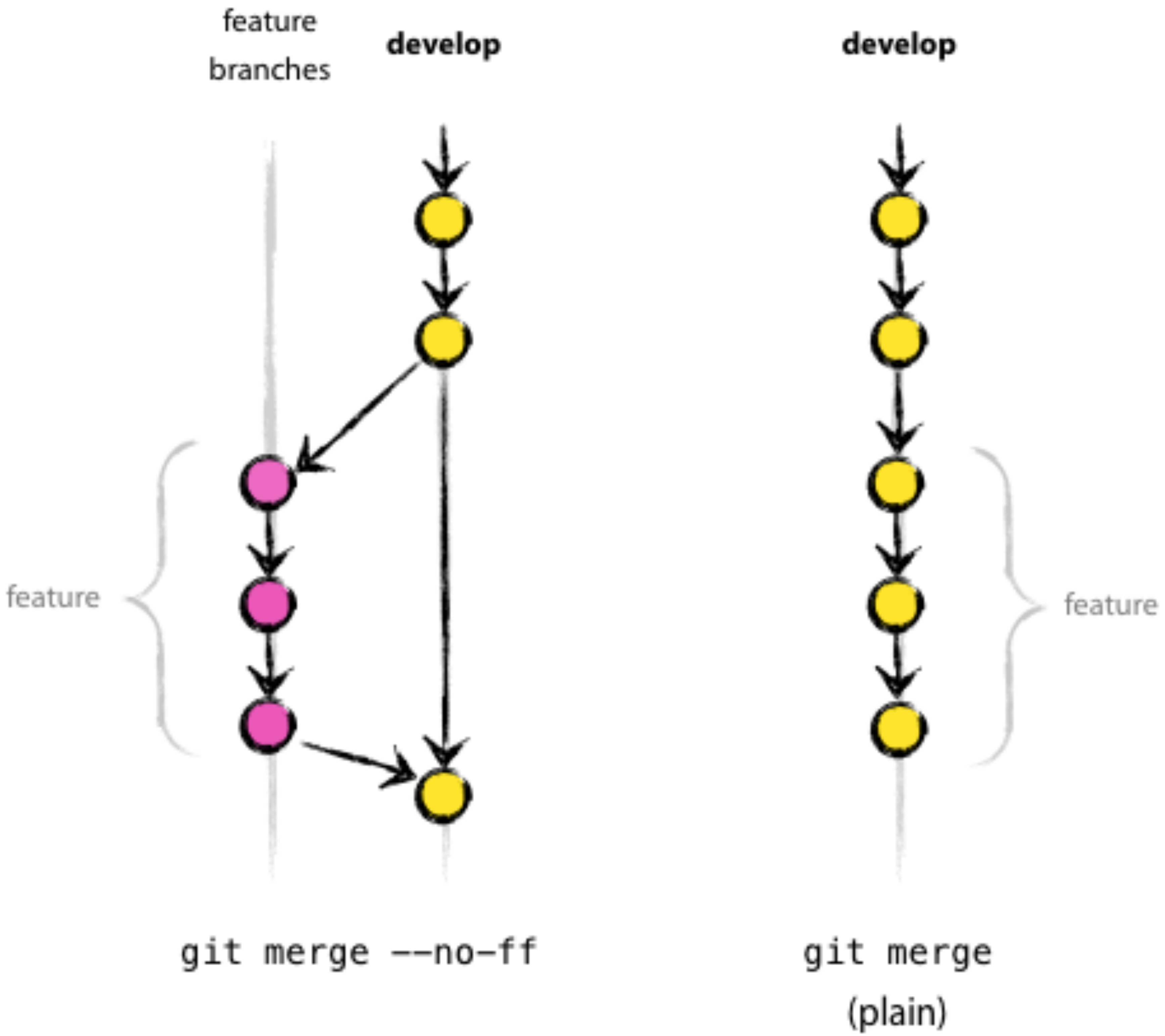
- master: 穩定的 production 版
- develop: 開發版本，從 master 分支出來


develop
master
Initial production version
Next production release
Next production release
Work in progress on "next release"

三種支援性分支(1)

- Feature branches
  - 開發新功能或修 bugs
  - 從 develop 分支出來
  - 完成後 merge 回 develop
  - 如果開發時間較長，則需定期同步 develop 主幹的程式(初學可用 merge，建議改用 rebase) ，不然最後會合併不回去。


feature branches
develop

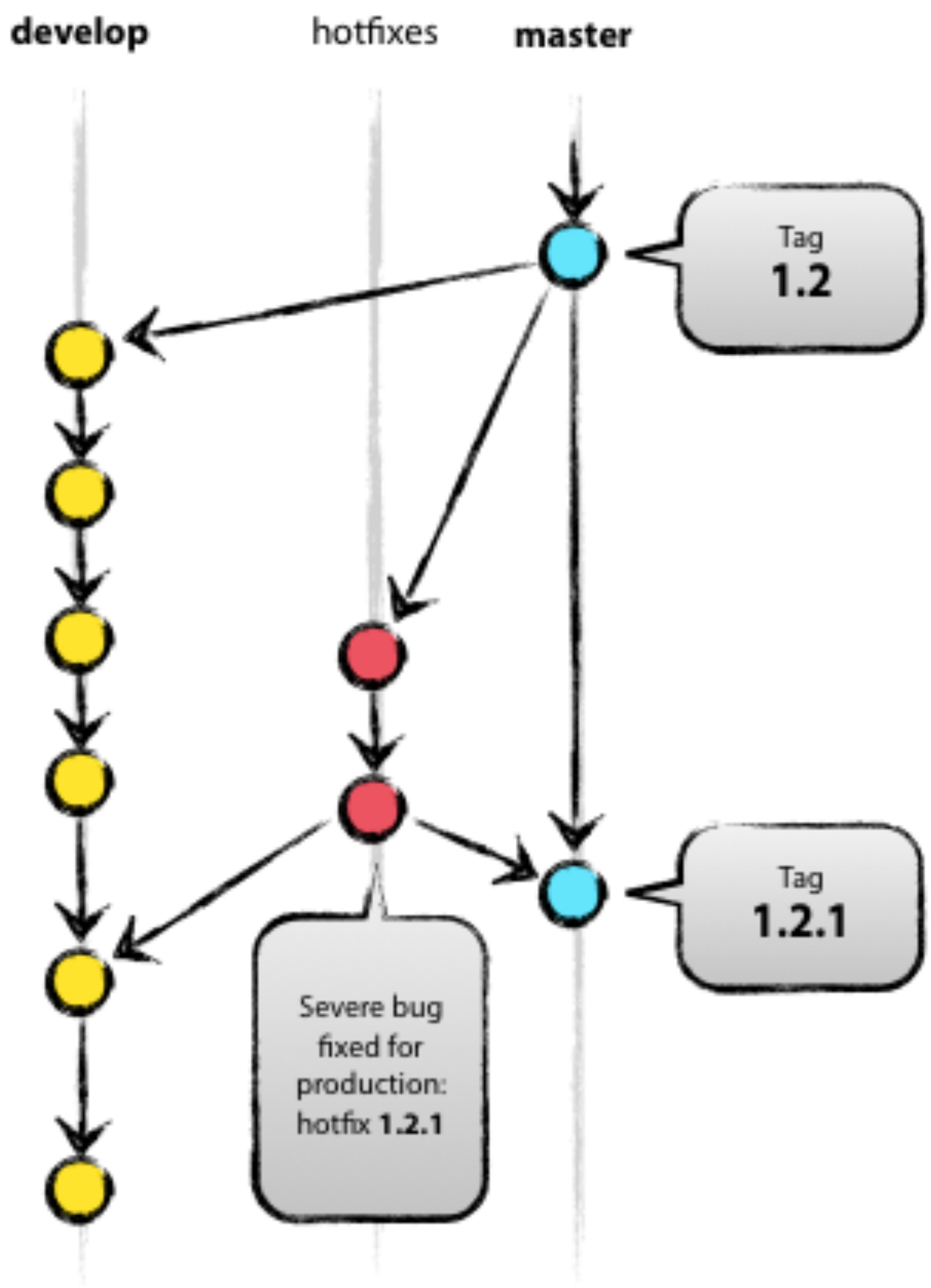
feature branches
develop
develop
feature
feature
git merge --no-ff
git merge
(plain)

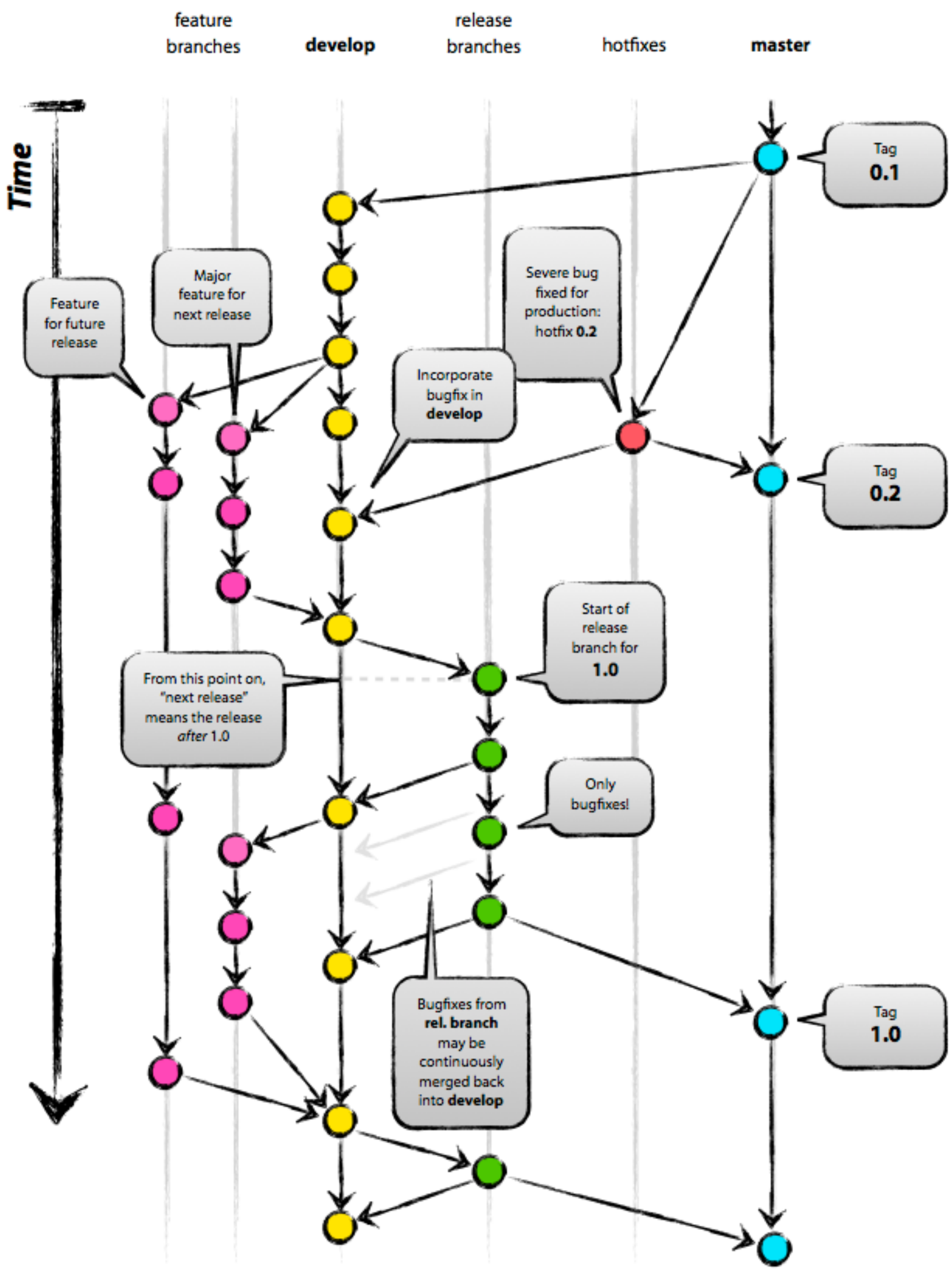
三種支援性分支(2)

- Release branches
  - 準備要 release 的版本，只修 bugs
  - 從 develop 分支出來
  - 完成後 merge 回 master 和 develop

三種支援性分支(3)

- Hotfix branches
  - 等不及 release 版本就必須馬上修 master 趕著上線
  - 會從 master 分支出來
  - 完成後 merge 回 master 和 develop


develop
hotfixes
master
Tag
1.2
Tag
1.2.1
Severe bug fixed for production: hotfix **1.2.1**


feature branches
develop
release branches
hotfixes
master
Time
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release *after* 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

pros and cons

- 第一個出名的、有形式的 Git 分支工作流程
- 很清楚的追蹤 feature, release, hotfix 等 branches，確保釋出和審核流程
- 有點複雜，不適合小專案或 Open Source 專案
- 不適合頻繁釋出(Continuous Deployment)

Discussions

- Rebase before merge
- Feature branch issue
- Code review policy
- Release branch

1. Rebase before merge

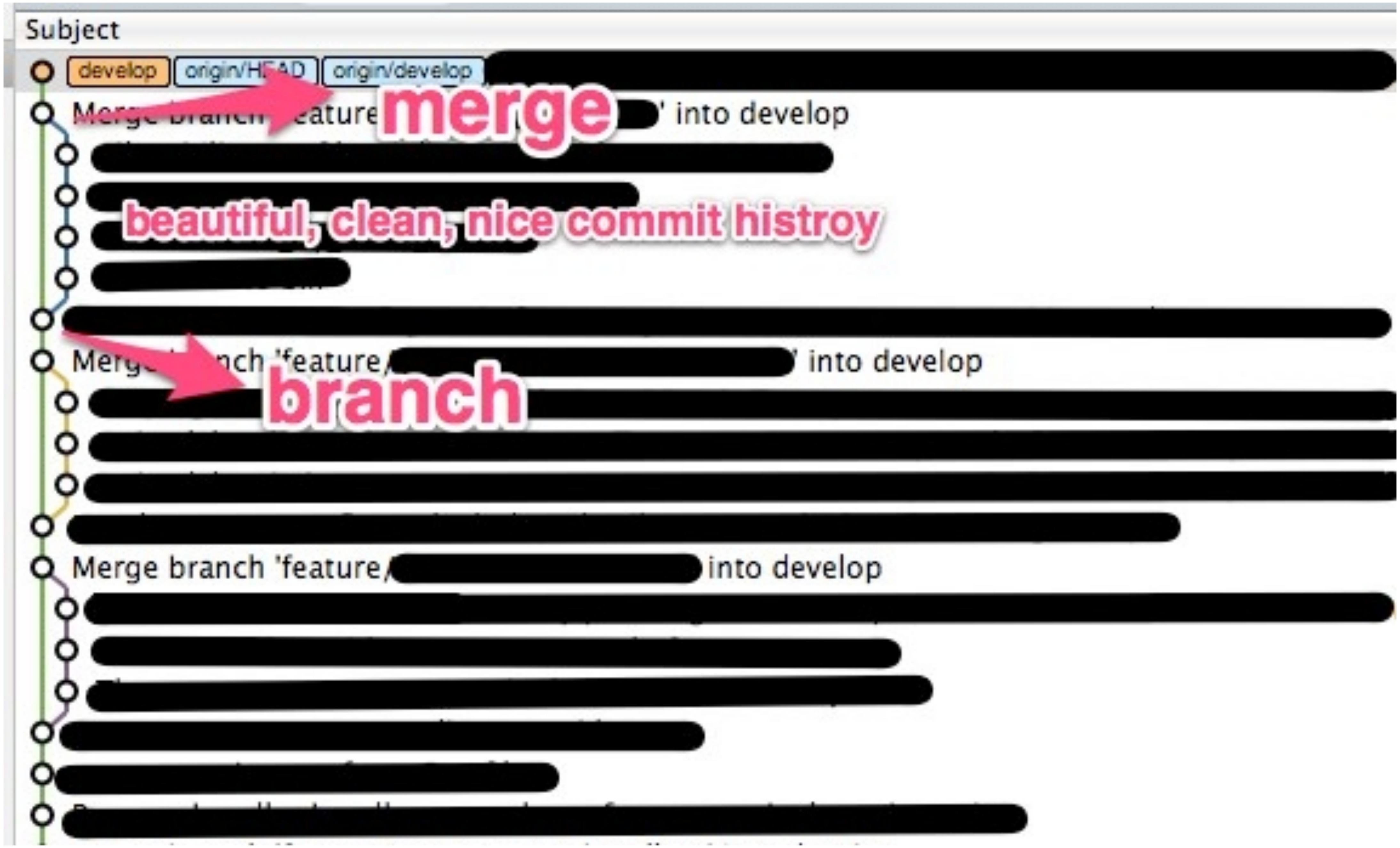
- 在 merge 前，做不做 rebase 整理 commits?
  - 建議能力和時間所及，就去做

rebase + merge 的完美合併法

(假設我們要將 feature branch 合併回主幹 develop)

- 原因
  - feature branch 很亂，不時 merge 與主幹同步
  - feature branch 有 typo，commit 訊息想改
  - feature branch 有些 commits 想合併或拆開
- 作法
  - 先在 feature branch 做 git rebase develop -i
  - (反覆整理直到滿意) git rebase 分岔點 -i
  - 在從 develop branch 做 git merge feature --no-ff

超級乾淨，每一次的 merge commit 就代表一個功能完成



Subject

develop origin/HEAD origin/develop [redacted]

Merge branch 'feature[redacted]' into develop

[redacted]

Merge branch 'feature/[redacted]' into develop

[redacted]

Merge branch 'feature/[redacted] into develop

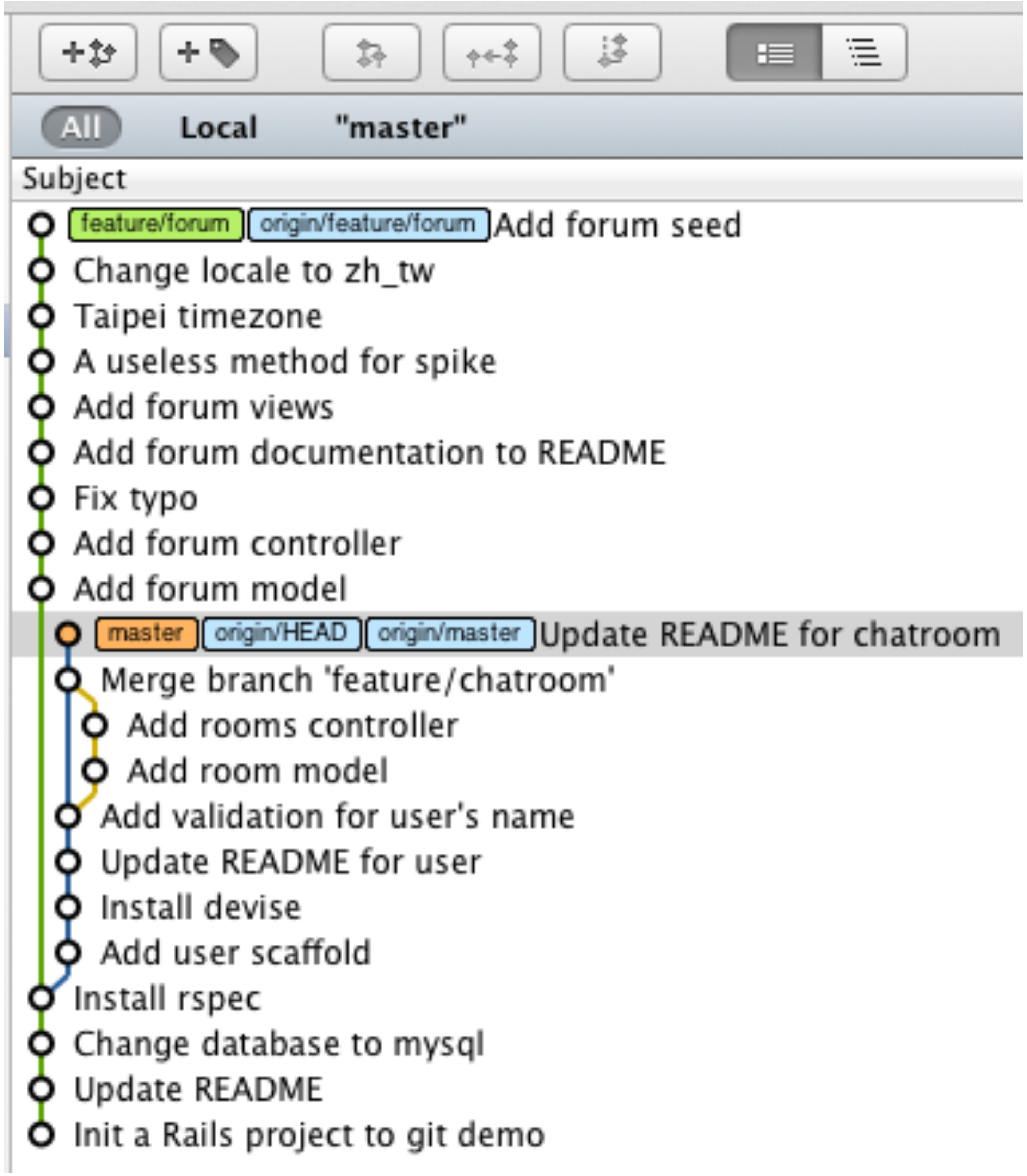
[redacted]

merge

beautiful, clean, nice commit histroy

branch

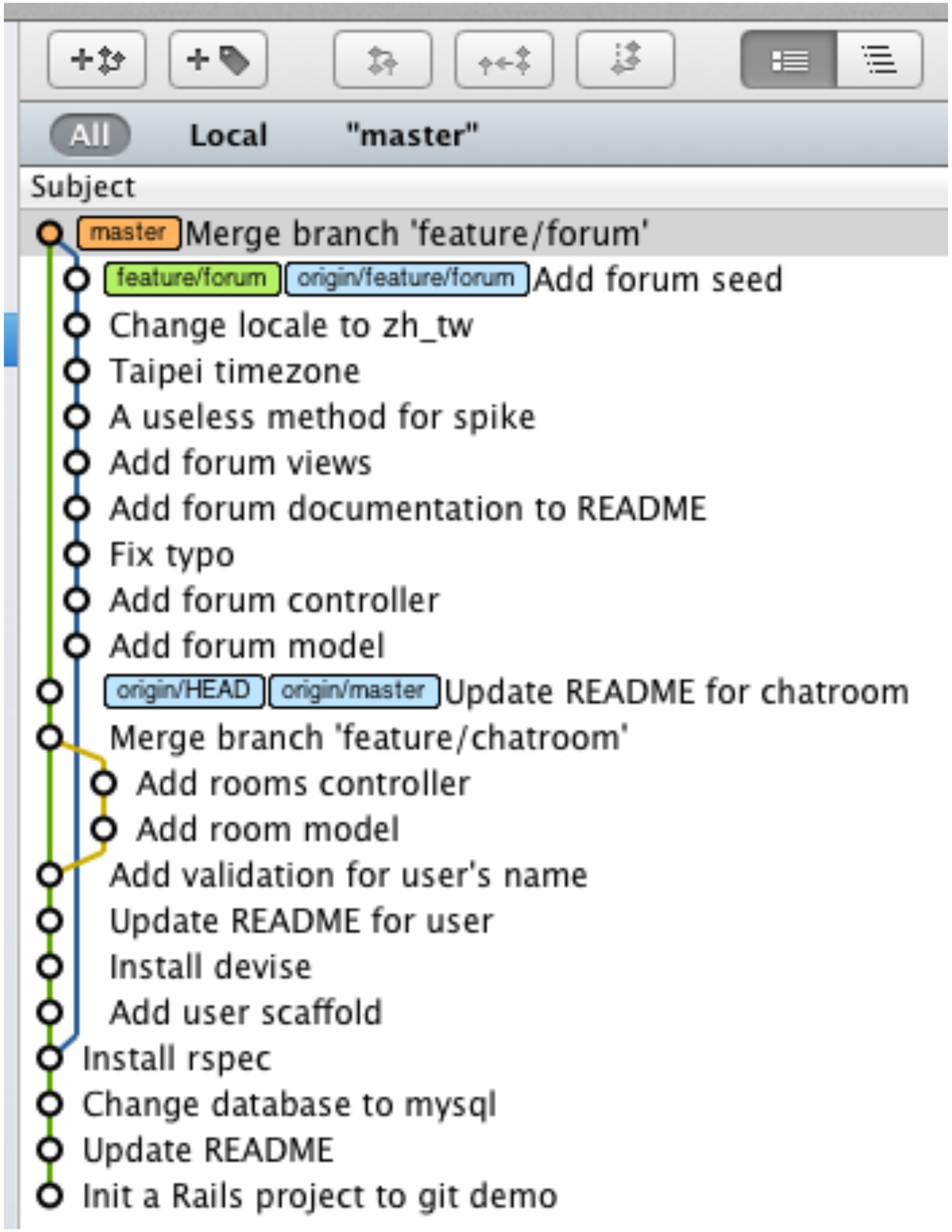
Demo (before merge)



feature/forum origin/feature/forum Add forum seed
Change locale to zh_tw
Taipei timezone
A useless method for spike
Add forum views
Add forum documentation to README
Fix typo
Add forum controller
Add forum model
master origin/HEAD origin/master Update README for chatroom
Merge branch 'feature/chatroom'
Add rooms controller
Add room model
Add validation for user's name
Update README for user
Install devise
Add user scaffold
Install rspec
Change database to mysql
Update README
Init a Rails project to git demo

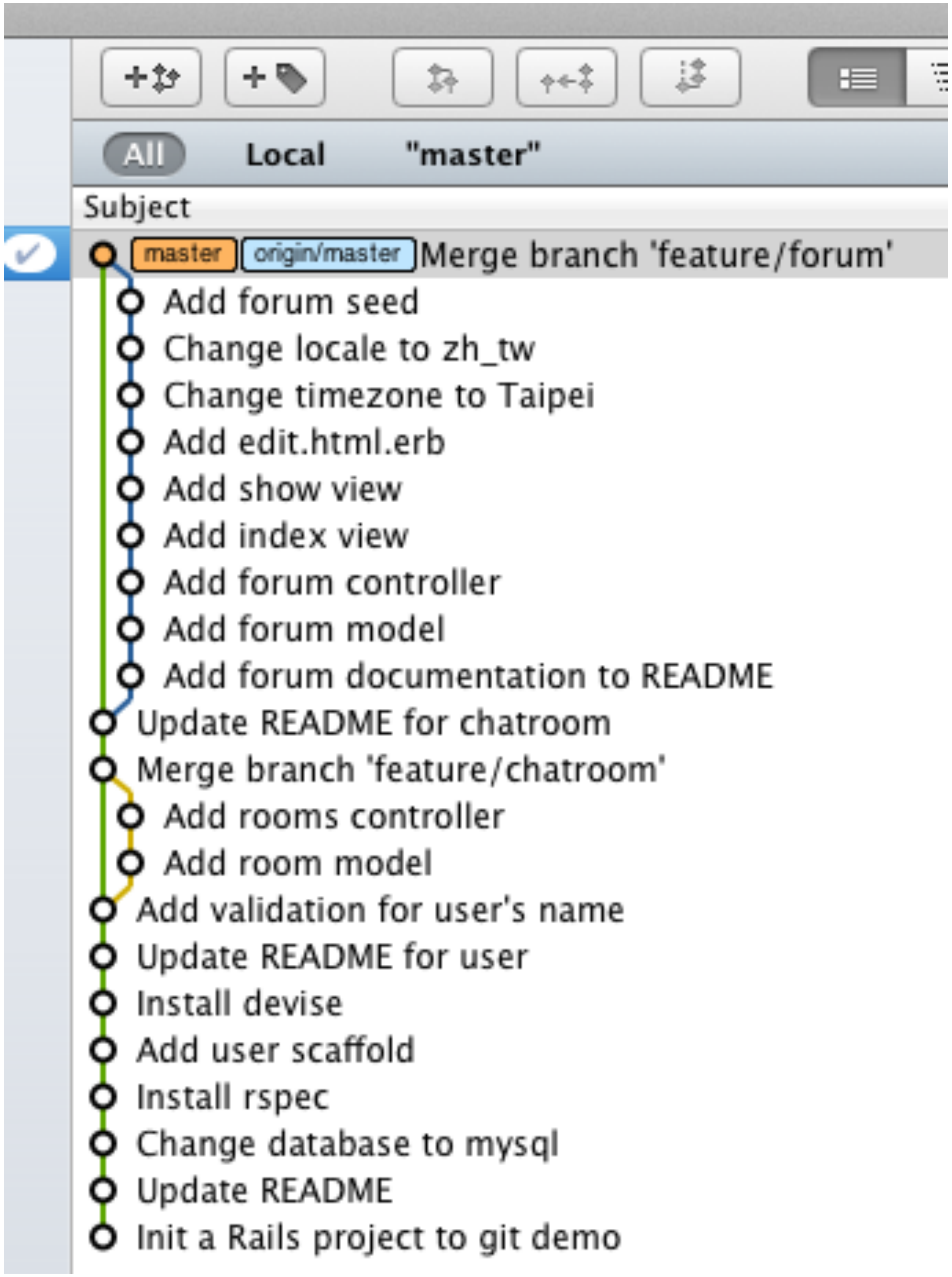
http://ihower.tw/blog/archives/6704/

Demo (normal merge)



master Merge branch 'feature/forum'
feature/forum origin/feature/forum Add forum seed
Change locale to zh_tw
Taipei timezone
A useless method for spike
Add forum views
Add forum documentation to README
Fix typo
Add forum controller
Add forum model
origin/HEAD origin/master Update README for chatroom
Merge branch 'feature/chatroom'
Add rooms controller
Add room model
Add validation for user's name
Update README for user
Install devise
Add user scaffold
Install rspec
Change database to mysql
Update README
Init a Rails project to git demo

Demo (rebase + merge)



Merge branch 'feature/forum'
Add forum seed
Change locale to zh_tw
Change timezone to Taipei
Add edit.html.erb
Add show view
Add index view
Add forum controller
Add forum model
Add forum documentation to README
Update README for chatroom
Merge branch 'feature/chatroom'
Add rooms controller
Add room model
Add validation for user's name
Update README for user
Install devise
Add user scaffold
Install rspec
Change database to mysql
Update README
Init a Rails project to git demo

注意事項 (1)

- 必須要加 --no-ff 才會有 merge commit。不然會是 fast-forward。
- rebase 之後的 feature branch 就不要再 push 出去了
- 如果有遠端的 feature branch，合併完也砍掉

注意事項 (2)

- 不求一次 rebase 到完美，不然中間的 conflict 會搞混。
- 可以一次改點東西就 rebase 一次，然後開個臨時的 branch 存檔起來，再繼續 rebase 自己直到滿意為止。

Rebase demo screencast

- http://ihower.tw/blog/archives/6704

2. Feature branch issue

- 根據 Feature 或 User Story 來建立 Branch 開發，直到 Branch 驗收完成才合併回主幹
- 可讓主幹總是 releasable
- 缺點是與主幹的分歧造成合併問題和不利於 CI
  - 主幹如果有更新，Feature Branch 必須經常去合併回來
  - Feature Branch 週期越短越好，幾天到不超過一個開發週期 (Iteration)
  - 不要同時開太多 Feature Branch (避免半成品)
  - 需要一個 Tech Lead 來負責主幹的合併

Feature Branch (cont.)

- 對 Open Source Project 來說非常有效率
  - Small core team 可以主導要不要接受 patch
  - Release date 不固定，可以慢慢考慮 patch
- 但對大型商業團隊來說，可能變成 Anti-pattern (超長 branch 合併不回來)。團隊需要良好紀律：
  - codebase 需要良好模組化
  - 大家都乖乖定期更新主幹的 code，並且經常 commit 程式到主幹
  - Delivery team 不能因為時程壓力而輕率 merge

反思 Develop on Mainline ?

from Continuous Delivery ch.14

- 以唯一的 Mainline branch 作為開發用途
  - 還是可以開以不合併為前提的 branch，例如 release branch 或 spike 實驗
- Continuous Integration 最好做，程式碼總是 integrated 的狀態
- 開發者總是拿到最新的 code
- 避免開發後期 merge hell 和 integration hell 的問題
- 缺點：releasable 程度可能降低，必須在架構設計上有增量式開發的能力和技巧。

在單一Branch 上做增量式開發的技巧

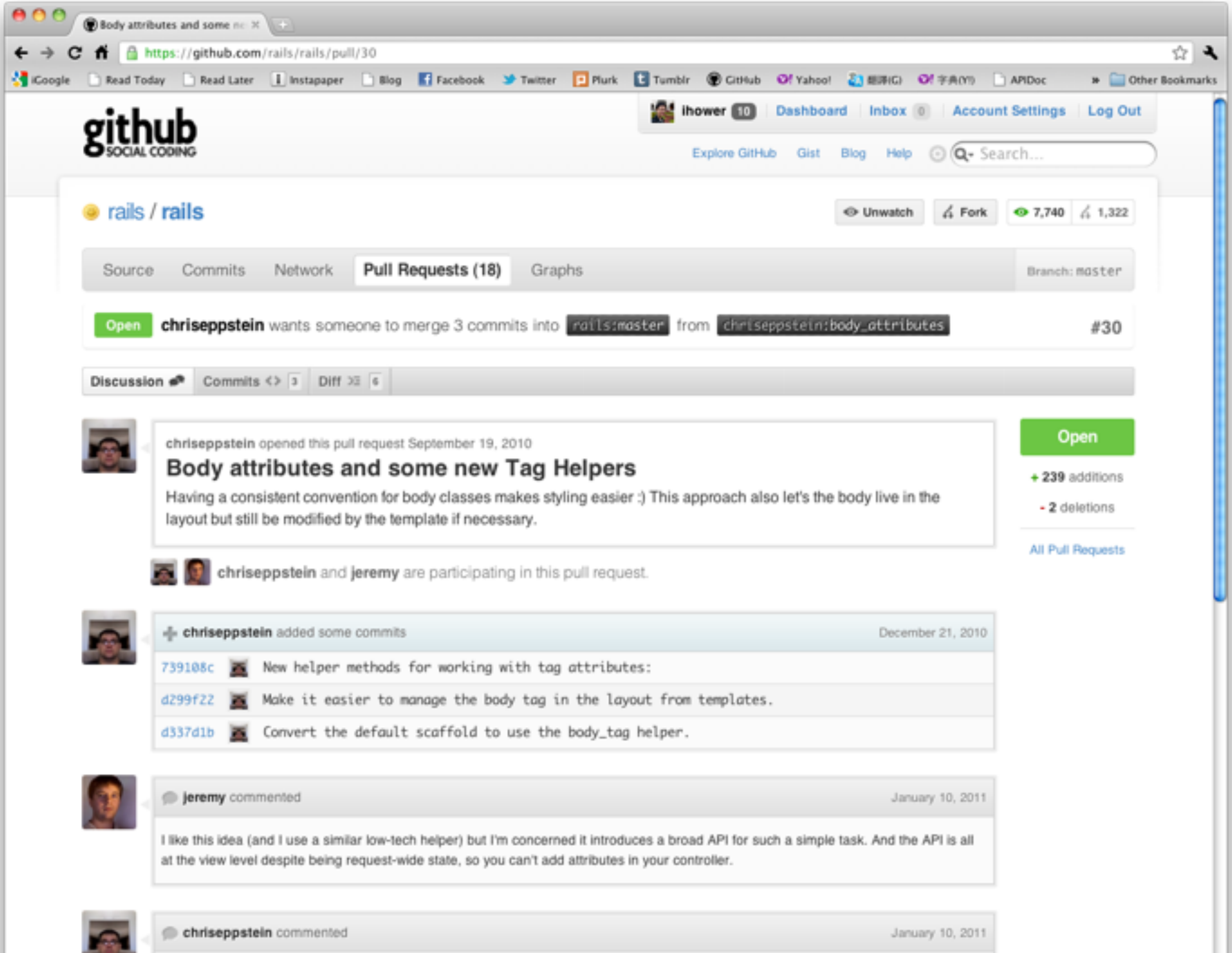
- Feature Toggle
  http://martinfowler.com/bliki/FeatureToggle.html
- Branch By Abstraction
  http://continuousdelivery.com/2011/05/make-large-scale-changes-incrementally-with-branch-by-abstraction/

Feature Toggle

- 功能完成，不代表業務面需要馬上上線
- 擔心 Feature branch 放太久臭掉?
- 建議先合併進主幹! 但是 UI 先藏起來即可!
  - 甚至是設計權限，讓 admin 可以提早看到做 production 的線上測試

3. Code Review Policy

- 規定只有 project leaders 可以 commit/merge 進 develop branch。
- 規定只有 release team 可以管理 master branch。
- 其他開發者都開 topic branches，完成後發 pull request，做完 code review 後，沒問題才 merge 進 develop。
- 例如：GitHub 的 pull request 可以作，但沒有強制性



Open **chriseppstein** wants someone to merge 3 commits into `rails:master` from `chriseppstein:body_attributes` #30

chriseppstein opened this pull request September 19, 2010

Body attributes and some new Tag Helpers

Having a consistent convention for body classes makes styling easier :) This approach also let's the body live in the layout but still be modified by the template if necessary.

+ **239** additions

- **2** deletions

chriseppstein and **jeremy** are participating in this pull request.

chriseppstein added some commits December 21, 2010

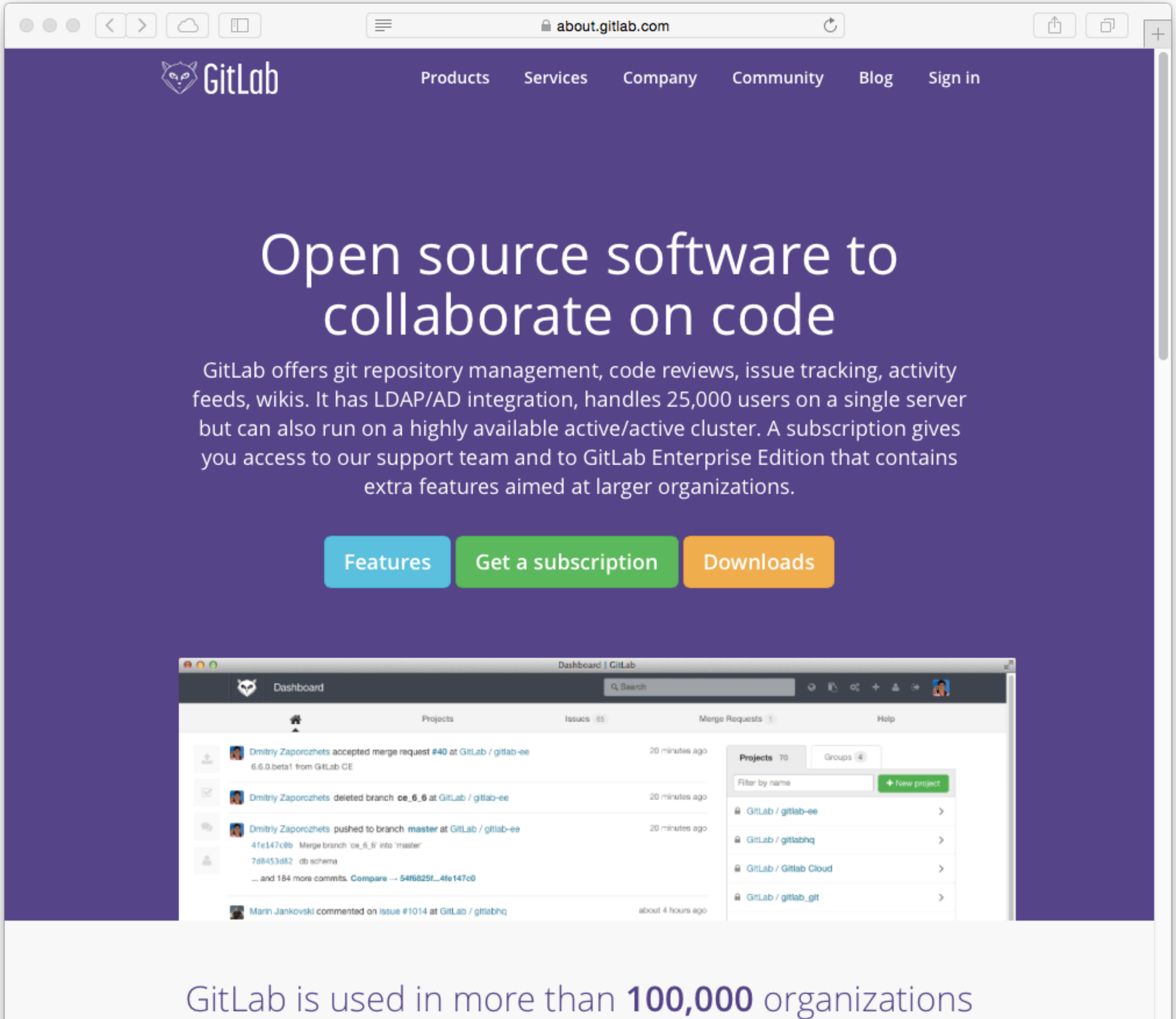
739108c	New helper methods for working with tag attributes:
d299f22	Make it easier to manage the body tag in the layout from templates.
d337d1b	Convert the default scaffold to use the body_tag helper.

jeremy commented January 10, 2011

I like this idea (and I use a similar low-tech helper) but I'm concerned it introduces a broad API for such a simple task. And the API is all at the view level despite being request-wide state, so you can't add attributes in your controller.

chriseppstein commented January 10, 2011

GitLab



Open source software to collaborate on code

GitLab offers git repository management, code reviews, issue tracking, activity feeds, wikis. It has LDAP/AD integration, handles 25,000 users on a single server but can also run on a highly available active/active cluster. A subscription gives you access to our support team and to GitLab Enterprise Edition that contains extra features aimed at larger organizations.

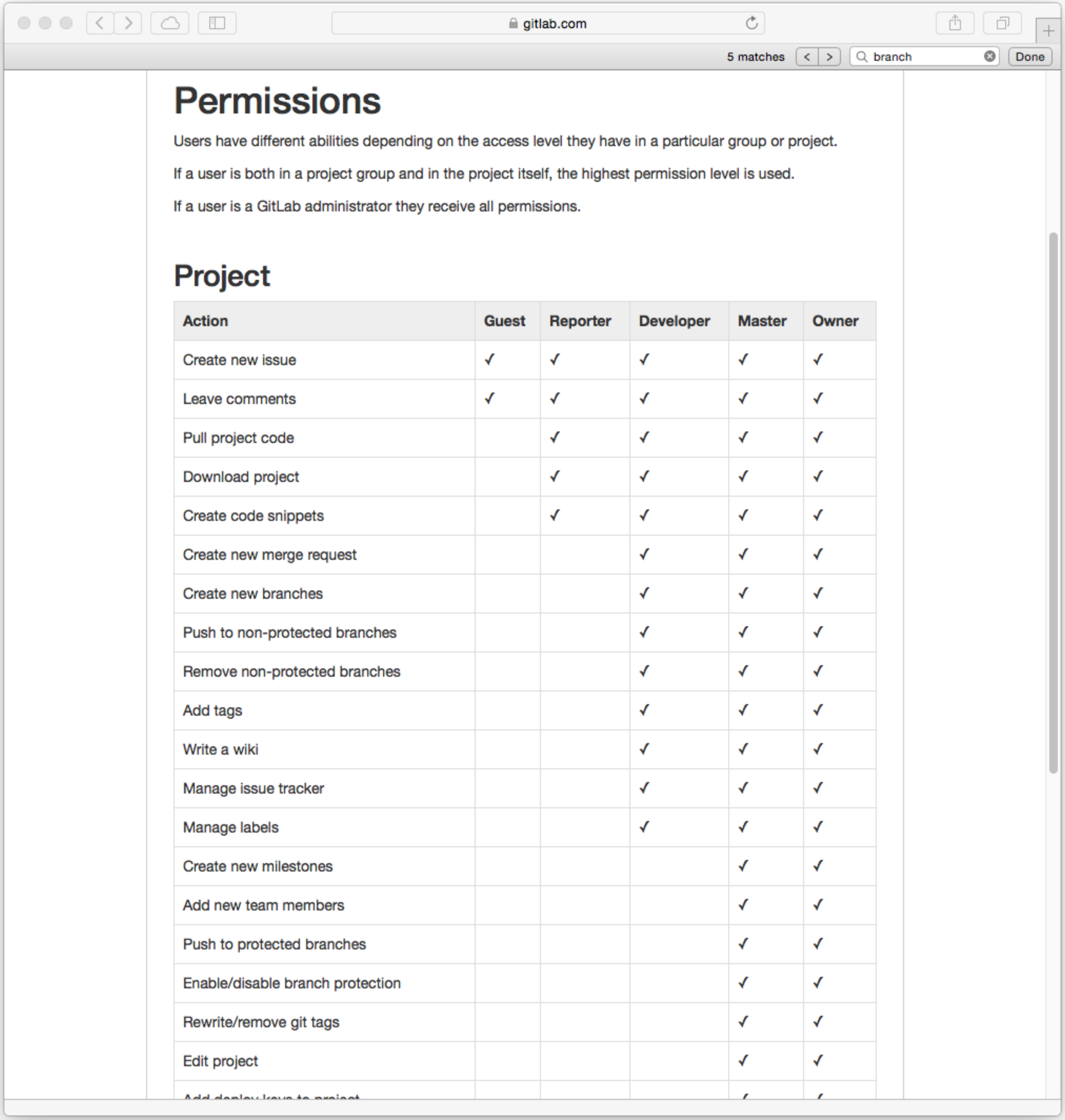
Dmitriy Zaporozhets accepted merge request #40 at GitLab / gitlab-ee 20 minutes ago
6.6.0.beta1 from GitLab CE

Dmitriy Zaporozhets deleted branch ce_6_6 at GitLab / gitlab-ee 20 minutes ago

Dmitriy Zaporozhets pushed to branch master at GitLab / gitlab-ee 20 minutes ago
4fe147c0b Merge branch 'ce_6_6' into 'master'
7d8453d82 db schema
... and 184 more commits. Compare → 54f6825f...4fe147c0

Marin Jankovski commented on issue #1014 at GitLab / gitlabhq about 4 hours ago

GitLab is used in more than **100,000** organizations



Permissions

Users have different abilities depending on the access level they have in a particular group or project.

If a user is both in a project group and in the project itself, the highest permission level is used.

If a user is a GitLab administrator they receive all permissions.

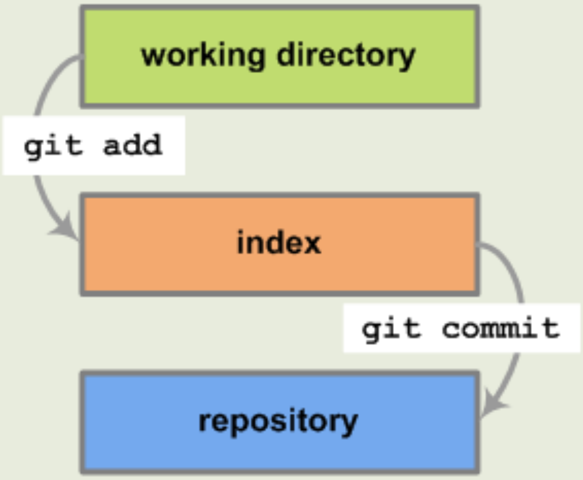
Project

Action	Guest	Reporter	Developer	Master	Owner
Create new issue	✓	✓	✓	✓	✓
Leave comments	✓	✓	✓	✓	✓
Pull project code		✓	✓	✓	✓
Download project		✓	✓	✓	✓
Create code snippets		✓	✓	✓	✓
Create new merge request			✓	✓	✓
Create new branches			✓	✓	✓
Push to non-protected branches			✓	✓	✓
Remove non-protected branches			✓	✓	✓
Add tags			✓	✓	✓
Write a wiki			✓	✓	✓
Manage issue tracker			✓	✓	✓
Manage labels			✓	✓	✓
Create new milestones				✓	✓
Add new team members				✓	✓
Push to protected branches				✓	✓
Enable/disable branch protection				✓	✓
Rewrite/remove git tags				✓	✓
Edit project				✓	✓
Add deploy keys to project				✓	✓

Android

- Repo is a repository management tool that we built on top of Git
- Gerrit is a web-based code review system for projects that use git

https://source.android.com/source/developing.html



Edit the files using vim / emacs / etc.,

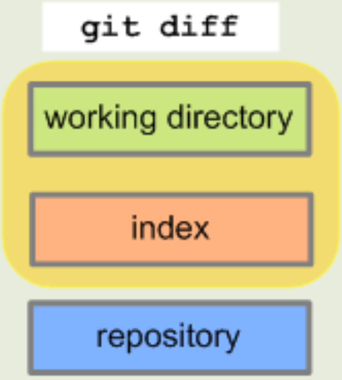
Stage the changes using `git add`

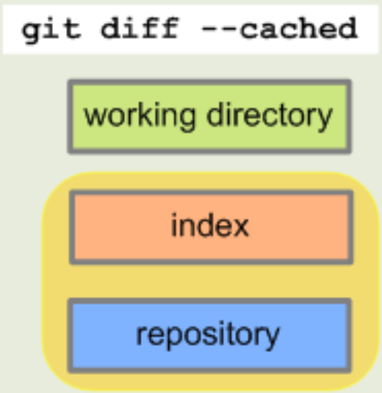
Review the changes using `repo status` .

Commit the changes using `git commit`

Common Commands

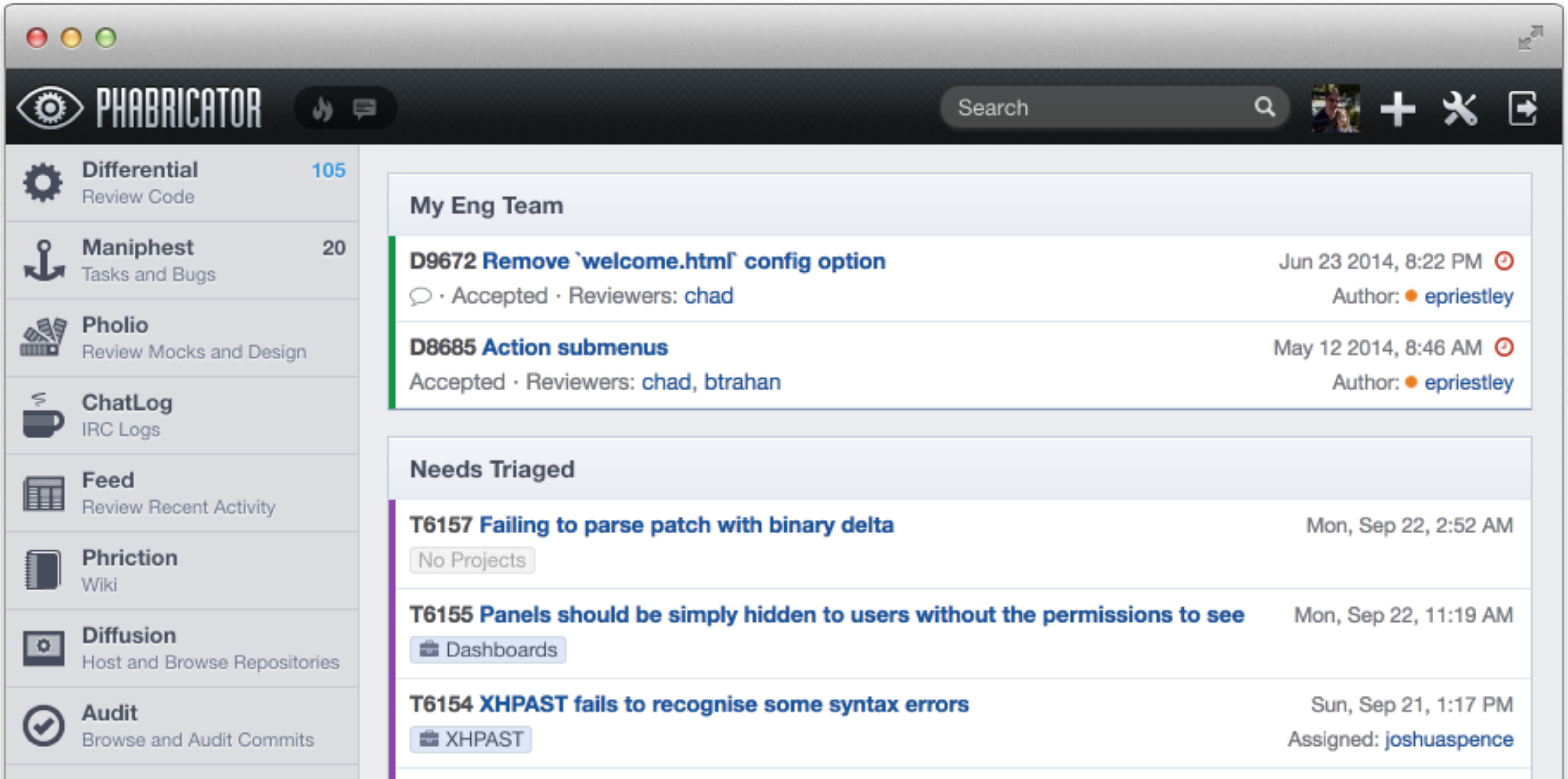
`repo init`	initializes a new client
`repo sync`	syncs client to repositories
`repo start`	starts a new branch
`git add`	stages files
`repo status`	shows status of current branch
`git commit`	commits staged files
`git branch`	shows current branches
`git branch [branch]`	creates new topic branch
`git checkout [branch]`	switches HEAD to specified branch
`git merge [branch]`	merges [branch] into current branch
`git diff`	shows diff of unstaged changes
`git diff --cached`	shows diff of staged changes
`git log`	shows history of current branch
`git log m/[codeline]..`	shows commits that are not pushed
`repo upload`	uploads changes to review server





Facebook

- http://phabricator.org/



Differential 105
Review Code

Maniphest 20
Tasks and Bugs

Pholio
Review Mocks and Design

ChatLog
IRC Logs

Feed
Review Recent Activity

Phriction
Wiki

Diffusion
Host and Browse Repositories

Audit
Browse and Audit Commits

My Eng Team

D9672 Remove `welcome.html` config option — Jun 23 2014, 8:22 PM
· Accepted · Reviewers: chad — Author: epriestley

D8685 Action submenus — May 12 2014, 8:46 AM
Accepted · Reviewers: chad, btrahan — Author: epriestley

Needs Triaged

T6157 Failing to parse patch with binary delta — Mon, Sep 22, 2:52 AM
No Projects

T6155 Panels should be simply hidden to users without the permissions to see — Mon, Sep 22, 11:19 AM
Dashboards

T6154 XHPAST fails to recognise some syntax errors — Sun, Sep 21, 1:17 PM
XHPAST — Assigned: joshuaspence

4. Release branch?

釋出週期越短
每天或 Continuous Deployment

釋出週期越長
數週或需要等 app store 審核

可以不需要 Release branch
直接主幹當 production 版

需要較穩定的
Release branches

開發與佈署流程

http://ihower.tw/blog/archives/7798

- 分支流程不只與開發相關，也與測試和軟體部署(釋出)流程相關
- 什麼時候，哪個分支跑 CI 自動測試?
- 什麼時候，哪個分支佈署到 Staging Server 進行人工測試?
  - 使用 Github flow 的話，需要讓每個 feature branch 都可以上 CI 和 staging server 環境
  - 或是採用 CMake 解法，有一個 next(staging) branch 專門用來整合跑 CI，這個 branch 不合併回 master

與專案管理的搭配

- Scrum (有 iteration 開發週期)
  - 比較適合搭配有 release branch 的 Git flow
- Kanban (沒有 iteration 開發週期)
  - 比較適合 Github flow 流程

Part2
TL;DR 小結

- Github flow 或 Gitflow 二選一
    - desktop/mobile software：用 Gitflow
    - 想要嚴謹的流程：用 Gitflow
    - 頻繁釋出的 Web app: 用 Github flow

謝謝，請多指教

http://ihower.tw

ALPHAcamp

參考資料

- http://ihower.tw/blog/category/git
- http://pragprog.com/screencasts/v-jwsceasy/source-control-made-easy
- http://www.youtube.com/watch?v=4XpnKHJAok8 Linux 的演講
- http://www.softdevtube.com/2013/02/05/advanced-git/
- http://git-scm.com/book
- Git from the bottom up
  http://ftp.newartisans.com/pub/git.from.bottom.up.pdf
- Version Control with Git, O'Reilly
- http://nfarina.com/post/9868516270/git-is-simpler
- http://think-like-a-git.net/sections/graph-theory.html

- Git in Practice, Manning
- https://peepcode.com/products/git
- https://peepcode.com/products/advanced-git
- Git Internals, Peepcode
- Pragmatic Version Control Using Git, Pragmatic
- Pragmatic Guide to Git, Pragmatic
- Continuous Delivery Ch.14
- https://www.atlassian.com/git/tutorials/comparing-workflows
- https://guides.github.com/introduction/flow/index.html