

## 分支 (branch)

什麼是分支？

分支的運用

分支的切換

分支的合併

Topic分支和integration分支的運用實例

## 教學1 使用分支

0. 事前準備

1. 建立分支

2. 切換分支

3. 合併分支

4. 刪除分支

5. 平行操作

6. 解決合併的衝突

7. 用rebase合併

遠端數據庫

Pull

Fetch

Push

標籤

標籤

## 教學2 使用標籤

0. 事前準備

1. 添加輕量標籤

2. 添加標示標籤

3. 刪除標籤

改寫提交

修改最近的提交

取消過去的提交

放棄提交

提取提交

改寫提交的記錄

匯合分支上的提交一同合併

## 教學3 改寫提交

1. Commit --amend

2. Revert

## 教學1 使用分支

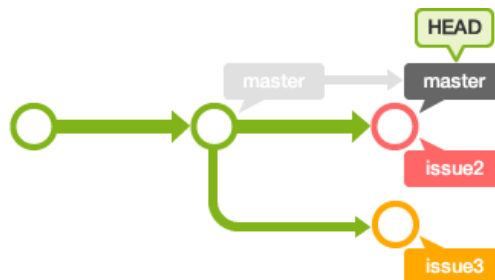
### 6. 解決合併的衝突

把 issue2 分支和 issue3 分支的修改合併到 master。

切換到 master 分支後，合併 issue2 分支。

```
$ git checkout master
Switched to branch 'master'
$ git merge issue2
Updating b2b23c4..8f7aa27
Fast-forward
 myfile.txt | 2 ++
 1 files changed, 2 insertions(+), 0 deletions(-)
```

執行 fast-forward (快轉) 合併。



接著合併 issue3 分支。

```
$ git merge issue3
Auto-merging myfile.txt
CONFLICT (content): Merge conflict in myfile.txt
Automatic merge failed; fix conflicts and then commit the result.
```

自動合併失敗。由於在同一行文字進行了修改，所以產生衝突。這時myfile.txt的內容如下：

```
連猴子都懂的Git命令
add 修改加入索引
<<<<<<< HEAD
commit 記錄索引的狀態
=====
pull 取得遠端數據庫的內容
>>>>>> issue3
```

在發生衝突的地方插入 Git 找到差異的部分，請做以下的修改：



```
連猴子都懂的Git命令
add 修改加入索引
```

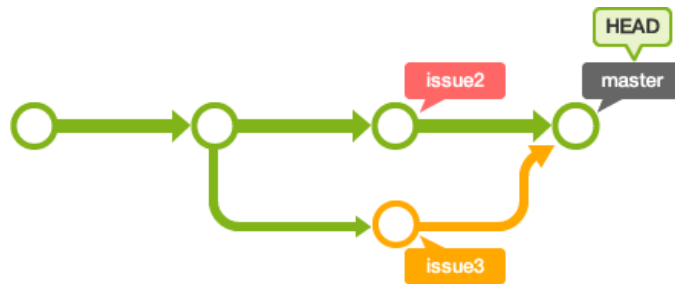
- 3. Reset
- 4. Cherry-pick
- 5. 使用 `rebase -i` 合併提交
- 6. 使用 `rebase -i` 修改提交
- 7. Merge --squash

`commit` 記錄索引的狀態  
`pull` 取得遠端數據庫的內容

衝突的部分已經修改，請再次提交。

```
$ git add myfile.txt
$ git commit -m "合併issue3"
# On branch master
nothing to commit (working directory clean)
```

歷史記錄如下圖所示。因為在這次的合併產生了衝突，必須修改衝突的部分，所以會建立新的提交記錄表示修改的合併提交。這樣，`master`的HEAD就移動到這裡了。這種合併不是`fast-forward`合併，而是`non fast-forward`合併。



[上一頁](#)

[下一頁](#)

Powered by  
**backlog**  
by nulab

繁體中文



Copyright © 2004-2019 Nulab Inc. All rights reserved.