GIT chapter 2

- git restore
- git branch
- git checkout/reset

陳岳洋(Jerry老師)

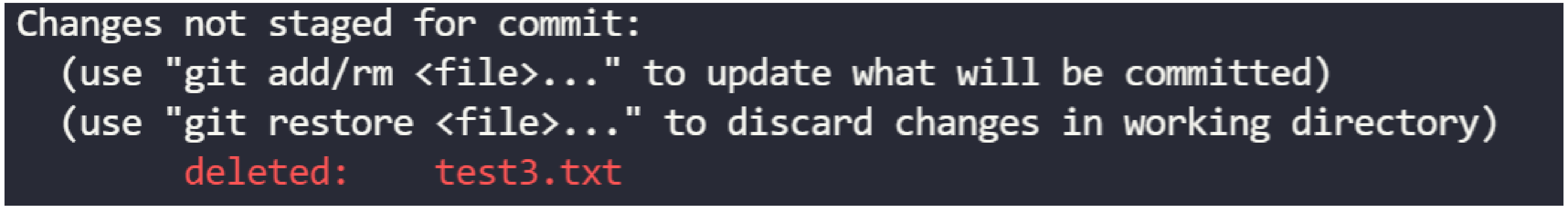
git restore

恢復檔案

git restore

git restore --staged <filename>

- 手動刪除檔案
  - 工作區
    - 沒控管
  - 暫存區/倉庫區
    - 轉成delete狀態
    - git add ➔確認刪除
    - git restore ➔恢復



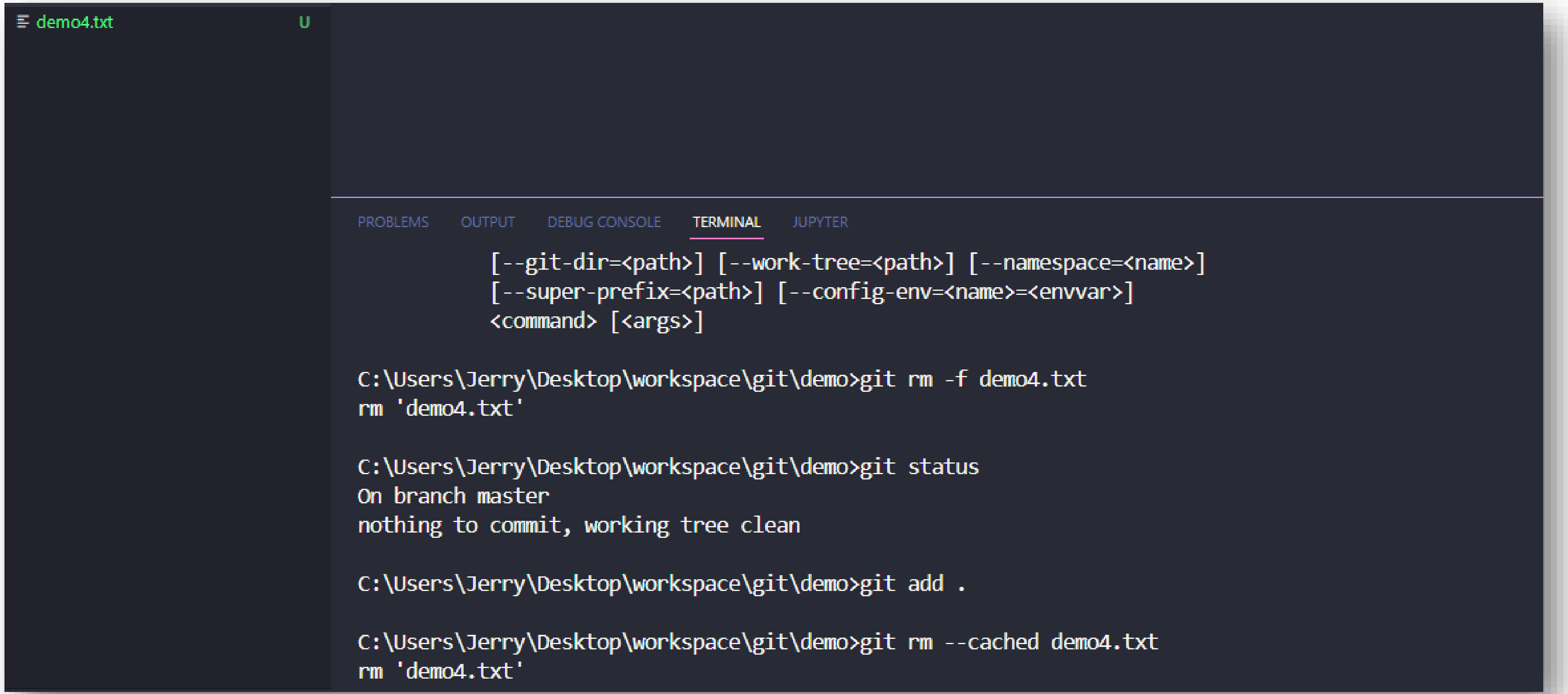
```
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    test3.txt
```

git rm

使用指令刪除檔案

```
git rm <filename>
git rm --cached <filename>
```

- 檔案在暫存區(未加入倉庫區)
  - 使用git rm －f 可以直接刪除(省略delete+git add確認)
  - 使用git rm －-cached
    - 將檔案從暫存區移到工作區



```
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
           <command> [<args>]

C:\Users\Jerry\Desktop\workspace\git\demo>git rm -f demo4.txt
rm 'demo4.txt'

C:\Users\Jerry\Desktop\workspace\git\demo>git status
On branch master
nothing to commit, working tree clean

C:\Users\Jerry\Desktop\workspace\git\demo>git add .

C:\Users\Jerry\Desktop\workspace\git\demo>git rm --cached demo4.txt
rm 'demo4.txt'
```

檔案在倉庫區

- 使用git rm [-f] <filename> 刪除
  - 將檔案從暫存區中移除
    - 確認儲存使用 git commit

使用-f 效果相同

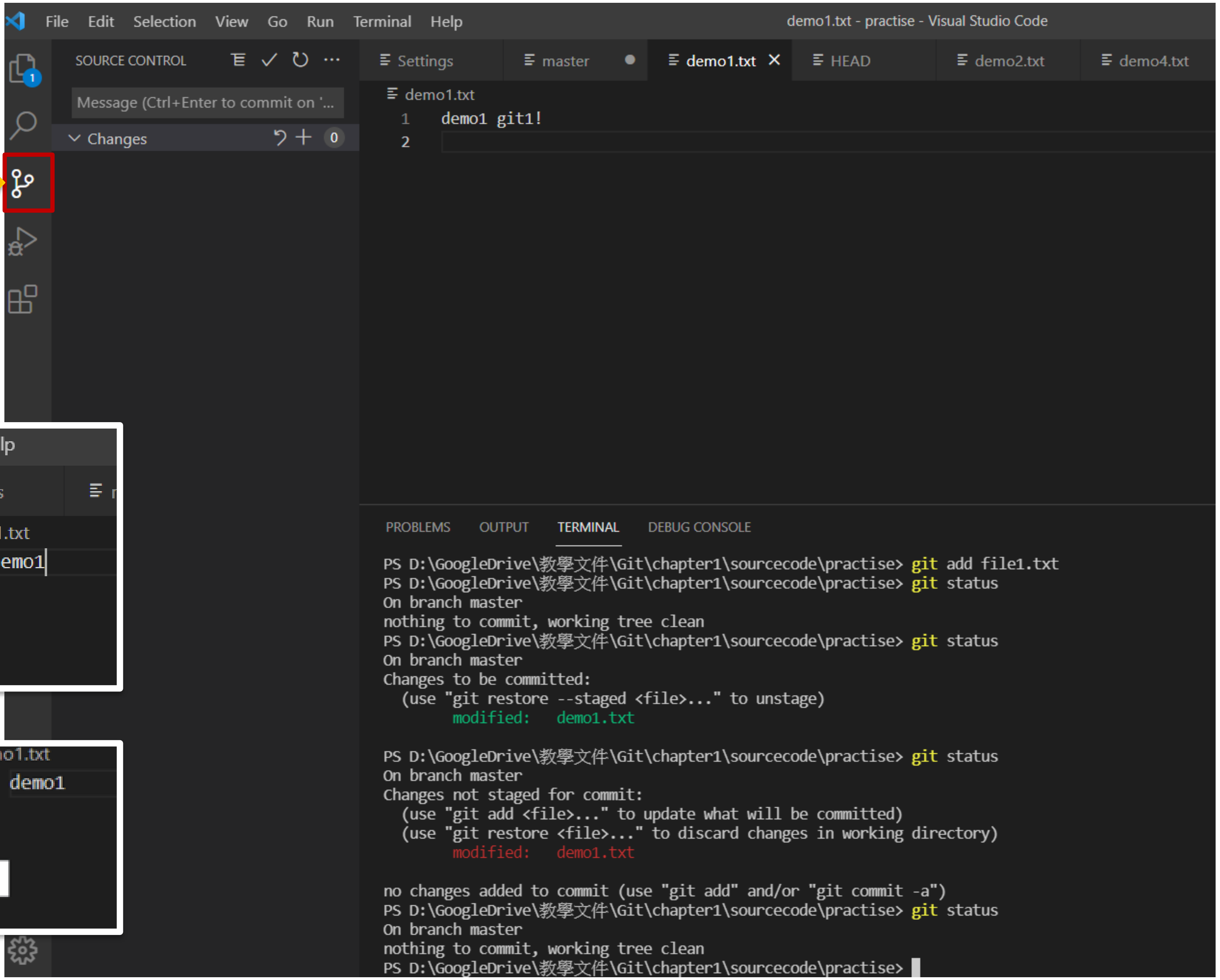
- git restore --staged <filename>
  - 將檔案狀態改為(deleted)
    - git add 確認刪除
    - git restore 恢復

使用rm刪除，恢復需要先unstage(類同手動刪除)

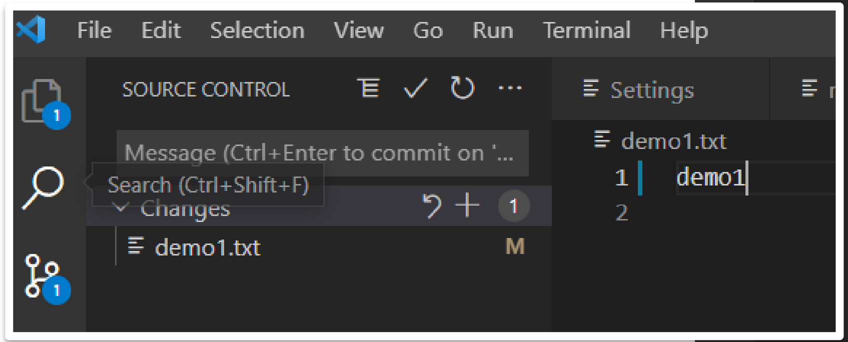
- 簡單來說➔
  - 在暫存區/倉庫區手動刪除的檔案
    - git restore 進行恢復
  - 在倉庫區git rm時需要恢復會比手動刪除時多一個
    - git restore --staged指令(unstage)
      - git restore 進行恢復
      - git add 確認
  - 使用git commit 確認更新儲存
  - git rm –-cached <file>
    - 從倉儲區/暫存區 中恢復檔案到untrack

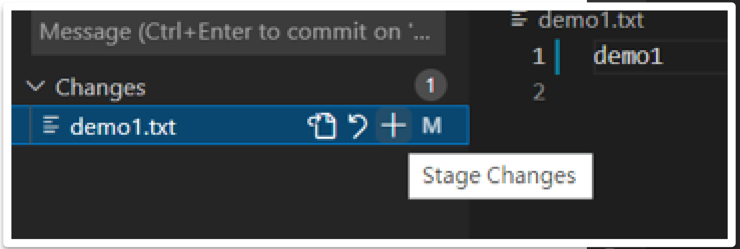
■ vscode也能進行操作

GIT狀態


File Edit Selection View Go Run Terminal Help
demo1.txt - practise - Visual Studio Code
SOURCE CONTROL
Settings
master
demo1.txt
HEAD
demo2.txt
demo4.txt
Message (Ctrl+Enter to commit on '...
Changes
0
demo1.txt
demo1 git1!
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE

```
PS D:\GoogleDrive\教學文件\Git\chapter1\sourcecode\practise> git add file1.txt
PS D:\GoogleDrive\教學文件\Git\chapter1\sourcecode\practise> git status
On branch master
nothing to commit, working tree clean
PS D:\GoogleDrive\教學文件\Git\chapter1\sourcecode\practise> git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   demo1.txt

PS D:\GoogleDrive\教學文件\Git\chapter1\sourcecode\practise> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   demo1.txt

no changes added to commit (use "git add" and/or "git commit -a")
PS D:\GoogleDrive\教學文件\Git\chapter1\sourcecode\practise> git status
On branch master
nothing to commit, working tree clean
PS D:\GoogleDrive\教學文件\Git\chapter1\sourcecode\practise> 
```


File Edit Selection View Go Run Terminal Help
SOURCE CONTROL
Settings
Message (Ctrl+Enter to commit on '...
Search (Ctrl+Shift+F)
Changes
1
demo1.txt
M
demo1



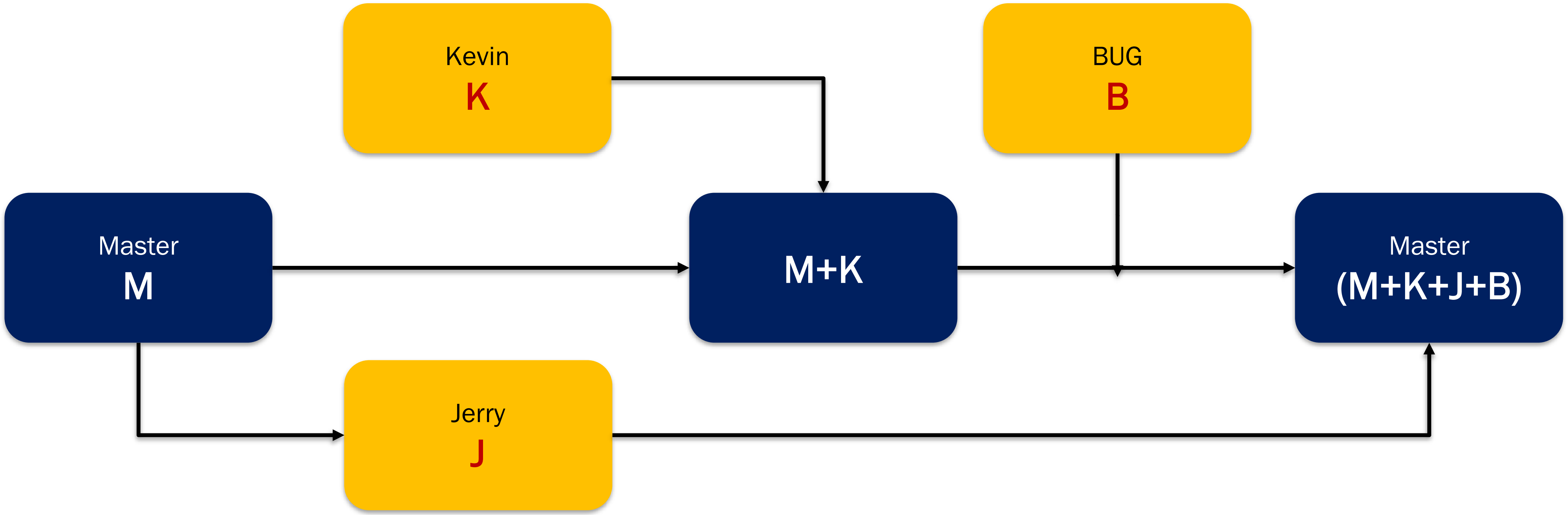
Git Basic Commands

- git init   (initialize an empty git repository)
- git status (show the working tree status)
- git add (add file contents to the index)
- git rm (Remove files from the working tree and from the index)
- git restore (Restore working tree files)
- git commit (Records the changes to repository)
- git log (show commit logs)
- git cat-file <SHA1>
  - -t (git object type)
  - -s (git object size)
  - -p (git object content)
- git ls-files
  - -s (Show staged contents' mode bits, object name and stage number in the output)

分支(branch)

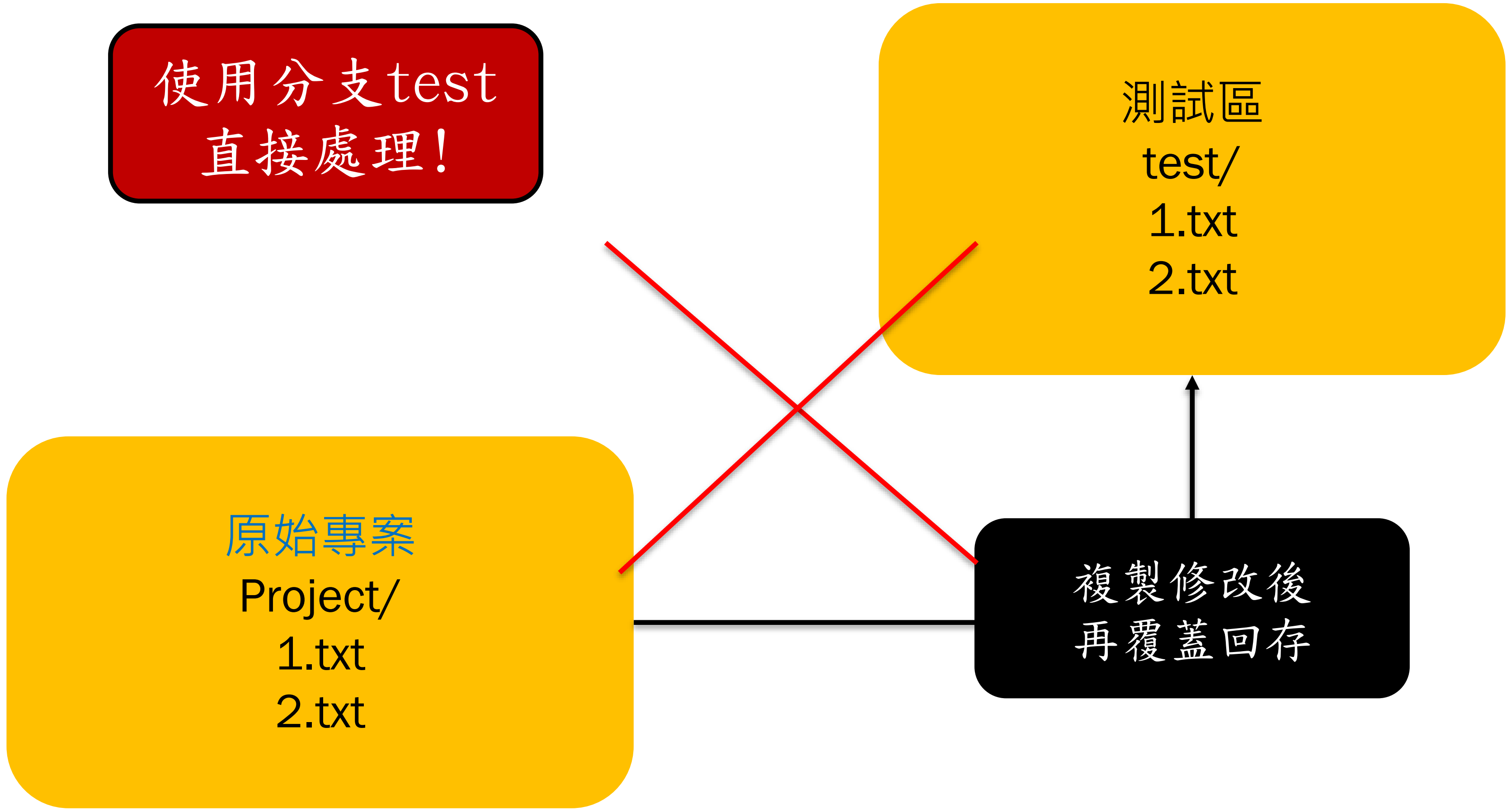
- 一個指向特定commit的指針
- git branch

- 為什麼需要分支?
  - 一個專案的完成通常需要多次測試或多方合作。
  - 為了不影響主要的工作分支(master)，故需要建立多個額外分支來進行測試跟功能製作或bug修復，最後在合併回主分支(master)

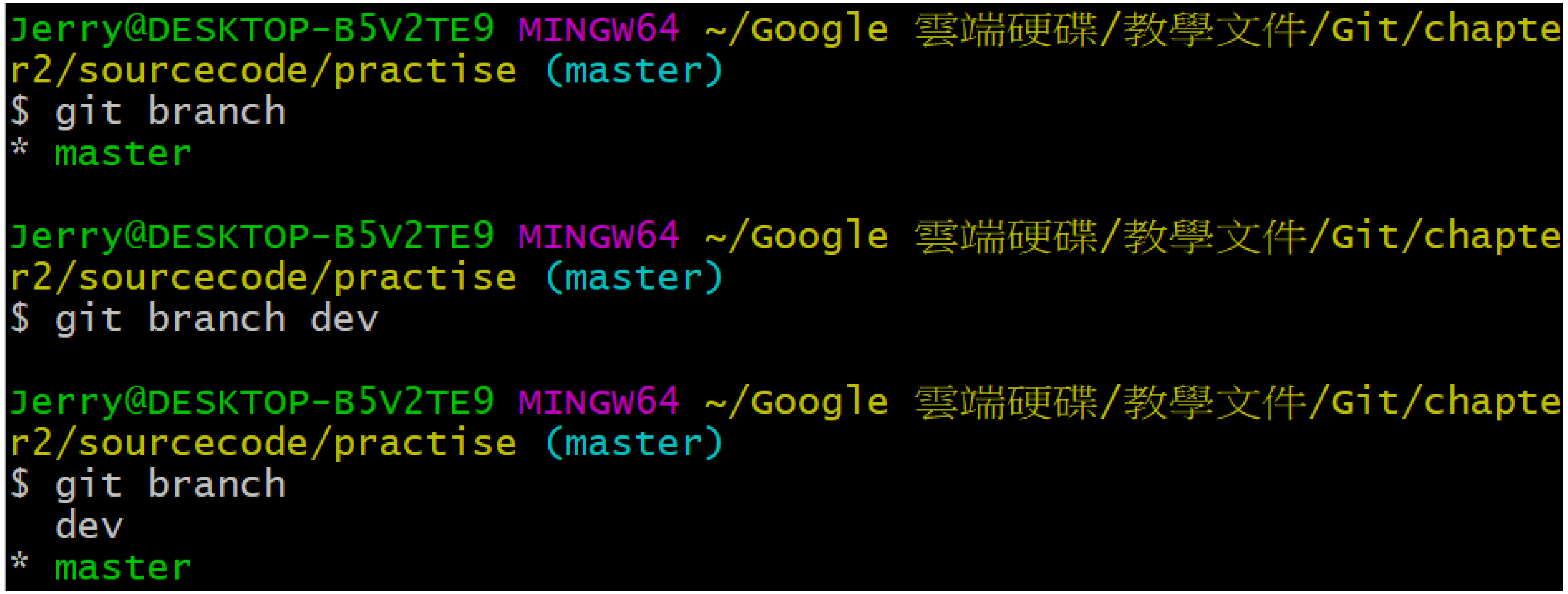


個人什麼時候該用分支?

- 不想影響到既有得程式碼
- 測試邏輯
- 修正bug
- …



- git branch
  - * master

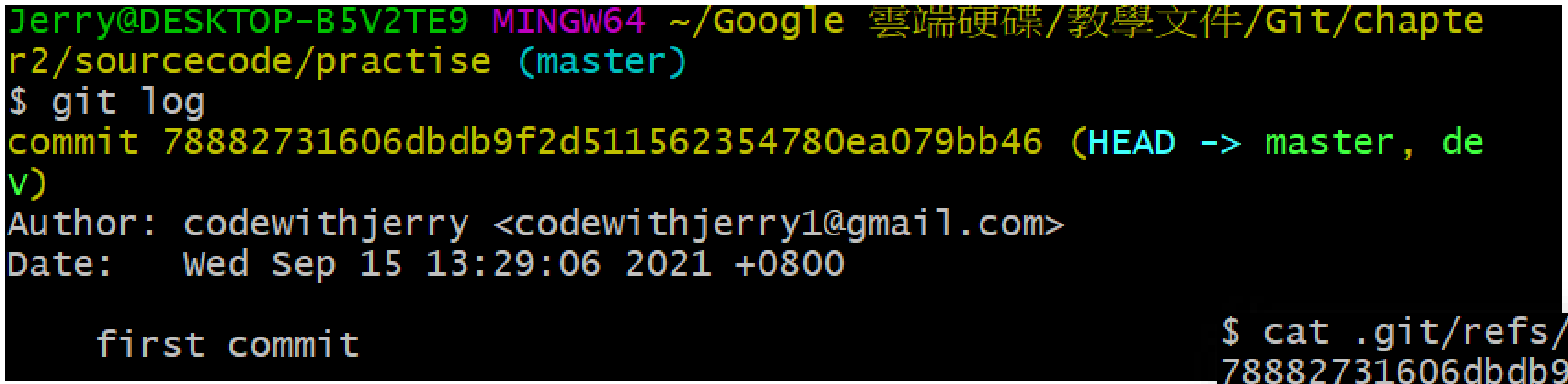


```
Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapte
r2/sourcecode/practise (master)
$ git branch
* master

Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapte
r2/sourcecode/practise (master)
$ git branch dev

Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapte
r2/sourcecode/practise (master)
$ git branch
  dev
* master
```

- git branch dev

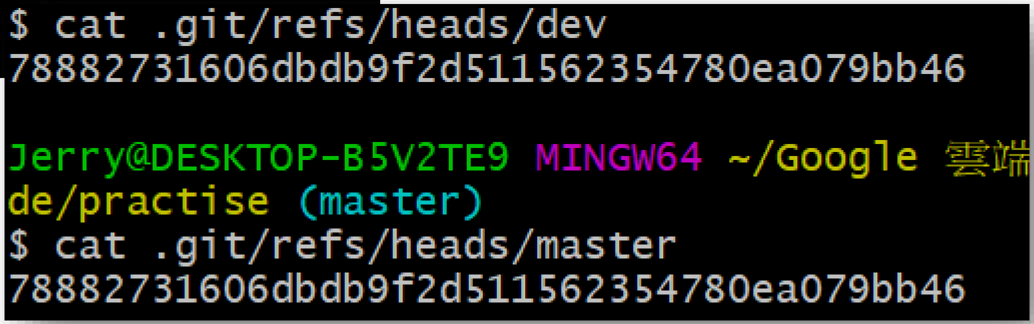
新增分支

git log

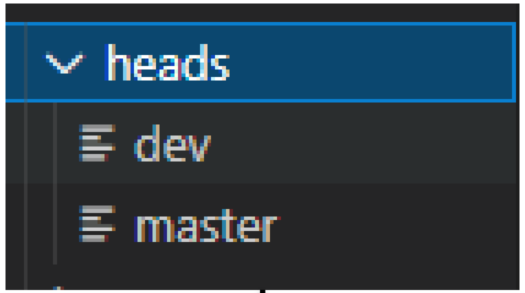


```
Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapte
r2/sourcecode/practise (master)
$ git log
commit 78882731606dbdb9f2d511562354780ea079bb46 (HEAD -> master, de
v)
Author: codewithjerry <codewithjerry1@gmail.com>
Date:   Wed Sep 15 13:29:06 2021 +0800

    first commit
```

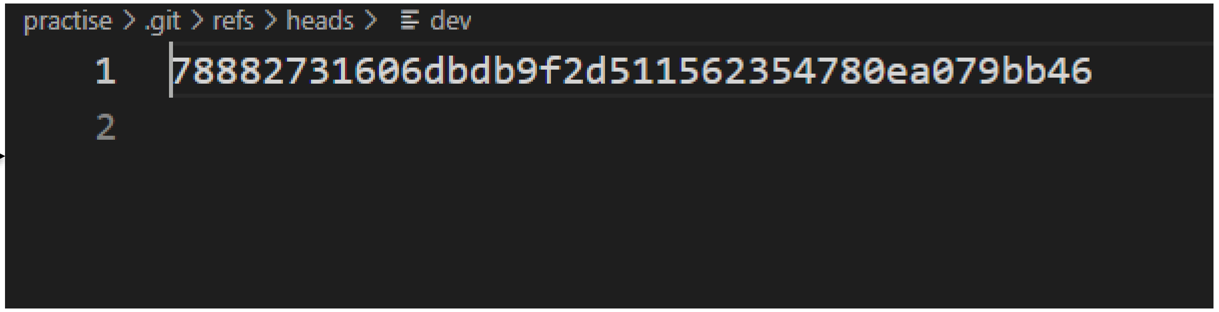
HEAD



```
$ cat .git/refs/heads/dev
78882731606dbdb9f2d511562354780ea079bb46

Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端
de/practise (master)
$ cat .git/refs/heads/master
78882731606dbdb9f2d511562354780ea079bb46
```



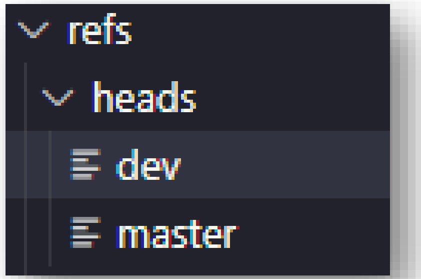
指向commit



```
practise > .git > refs > heads > dev
78882731606dbdb9f2d511562354780ea079bb46
```

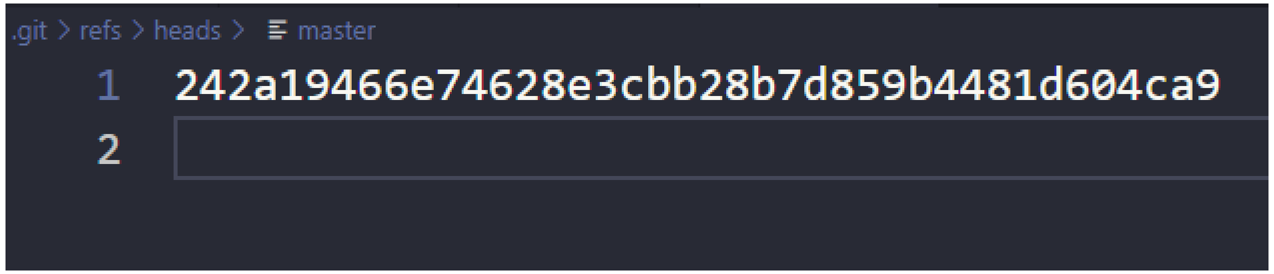
HEAD➔一直指向當前分支的最後commit

- 產生新分支紀錄
  - refs/heads/dev



refs
heads
dev
master

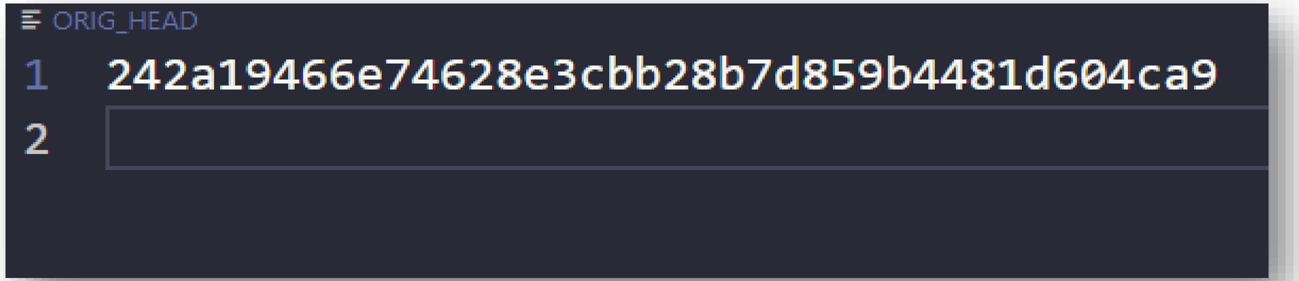
- HEAD
  - 指向目前運作的分支
  - ref: refs/heads/master



.git > refs > heads > master

```
242a19466e74628e3cbb28b7d859b4481d604ca9
```

- ORIG_HEAD
  - 指向上一個commit object

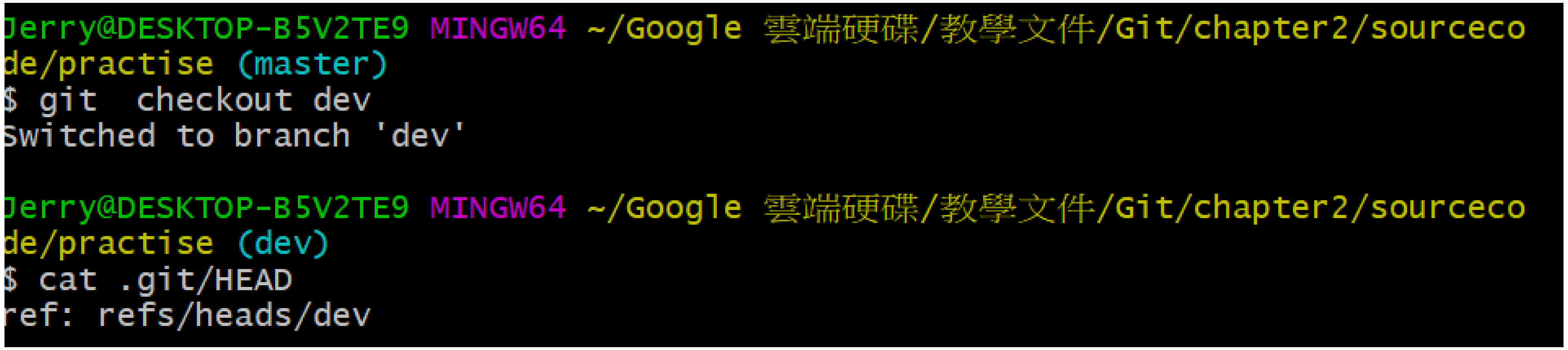


ORIG_HEAD

```
242a19466e74628e3cbb28b7d859b4481d604ca9
```

- git checkout 分支名稱
  - 切換分支

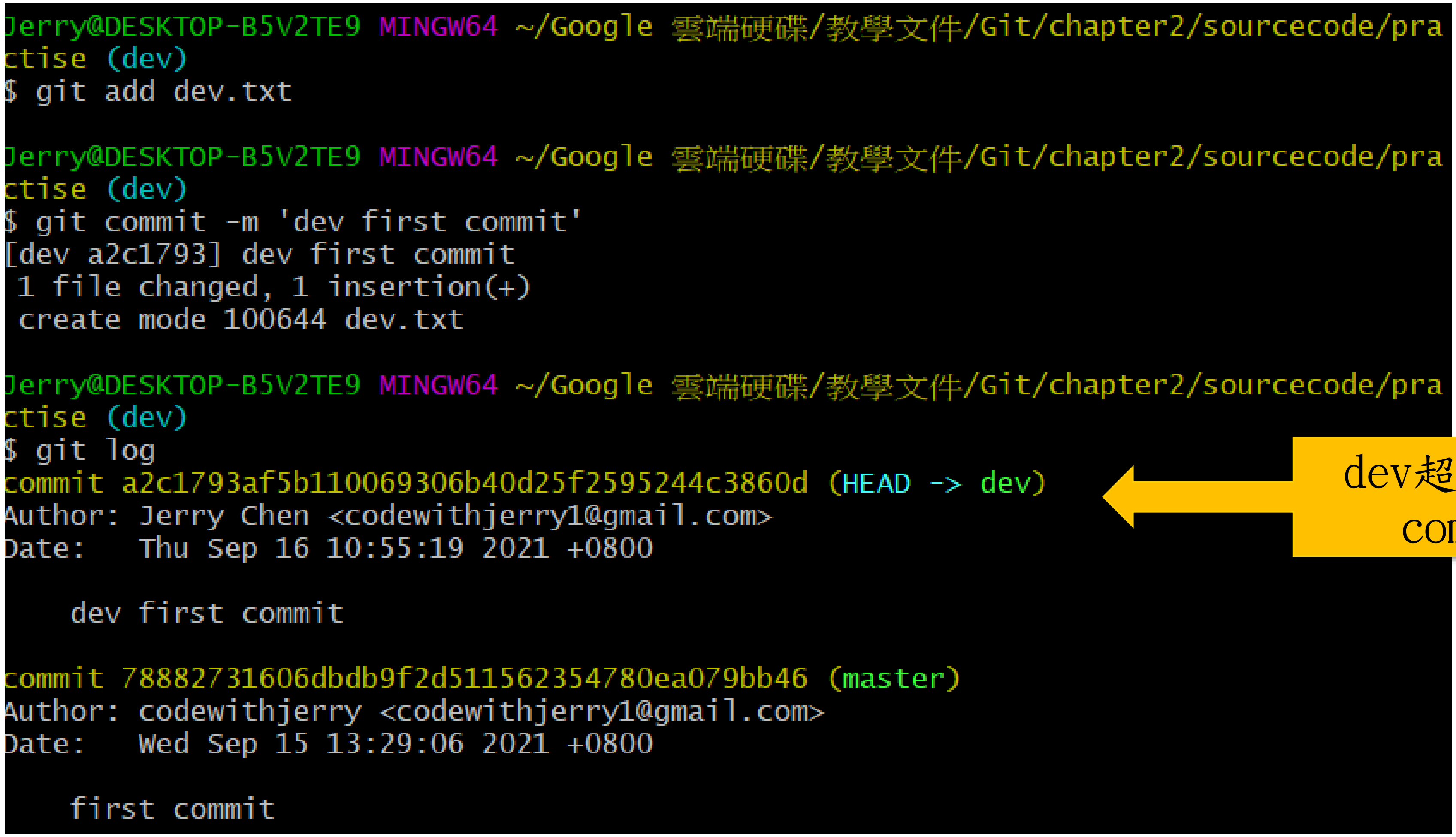
- git checkout dev



```
Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapter2/sourceco
de/practise (master)
$ git  checkout dev
Switched to branch 'dev'

Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapter2/sourceco
de/practise (dev)
$ cat .git/HEAD
ref: refs/heads/dev
```

分支新增commit

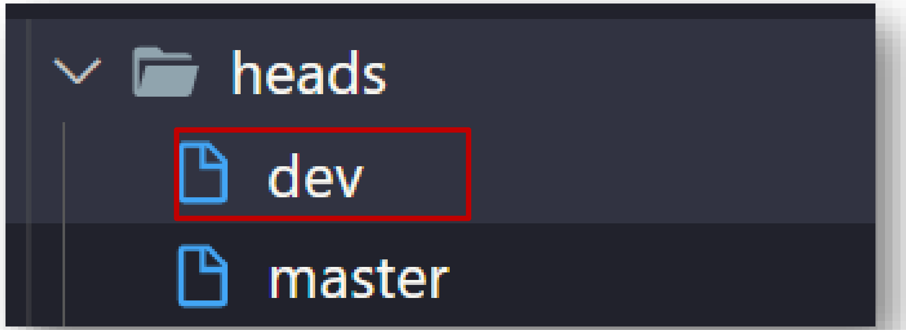
- 新增dev.txt檔案➔commit

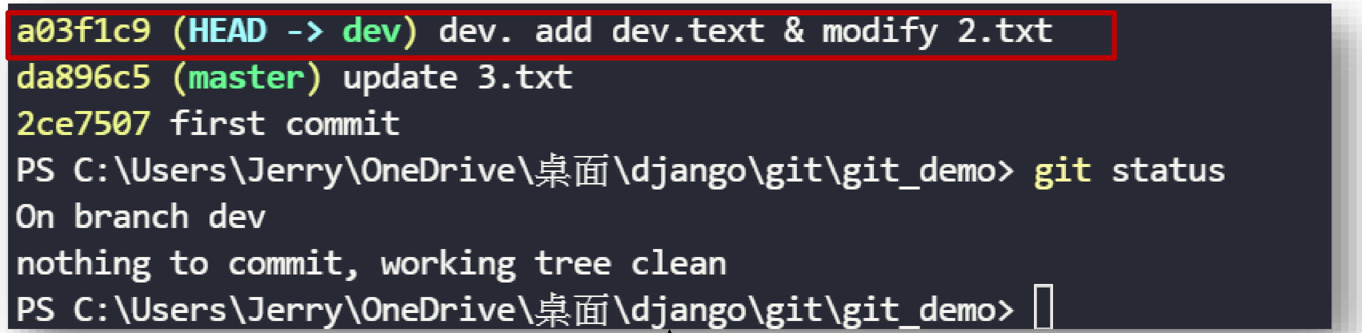


```
Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapter2/sourcecode/pra
ctise (dev)
$ git add dev.txt

Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapter2/sourcecode/pra
ctise (dev)
$ git commit -m 'dev first commit'
[dev a2c1793] dev first commit
 1 file changed, 1 insertion(+)
 create mode 100644 dev.txt

Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapter2/sourcecode/pra
ctise (dev)
$ git log
commit a2c1793af5b110069306b40d25f2595244c3860d (HEAD -> dev)
Author: Jerry Chen <codewithjerry1@gmail.com>
Date:   Thu Sep 16 10:55:19 2021 +0800

    dev first commit

commit 78882731606dbdb9f2d511562354780ea079bb46 (master)
Author: codewithjerry <codewithjerry1@gmail.com>
Date:   Wed Sep 15 13:29:06 2021 +0800

    first commit
```

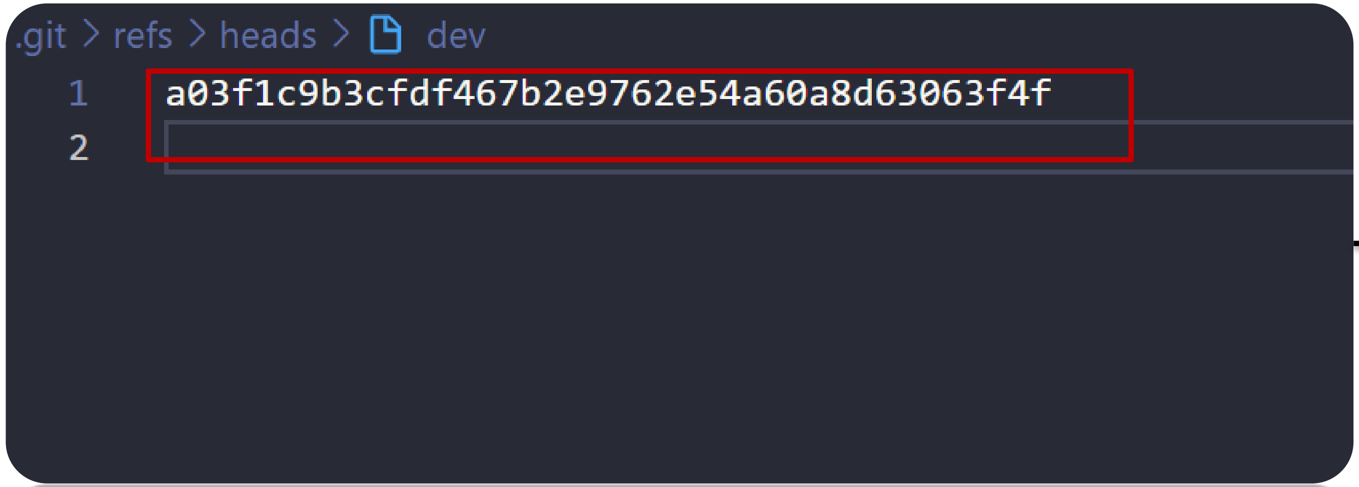
dev超前一個 commit

- HEAD移動到dev
  - 最新的commit





```
a03f1c9 (HEAD -> dev) dev. add dev.text & modify 2.txt
da896c5 (master) update 3.txt
2ce7507 first commit
PS C:\Users\Jerry\OneDrive\桌面\django\git\git_demo> git status
On branch dev
nothing to commit, working tree clean
PS C:\Users\Jerry\OneDrive\桌面\django\git\git_demo> 
```

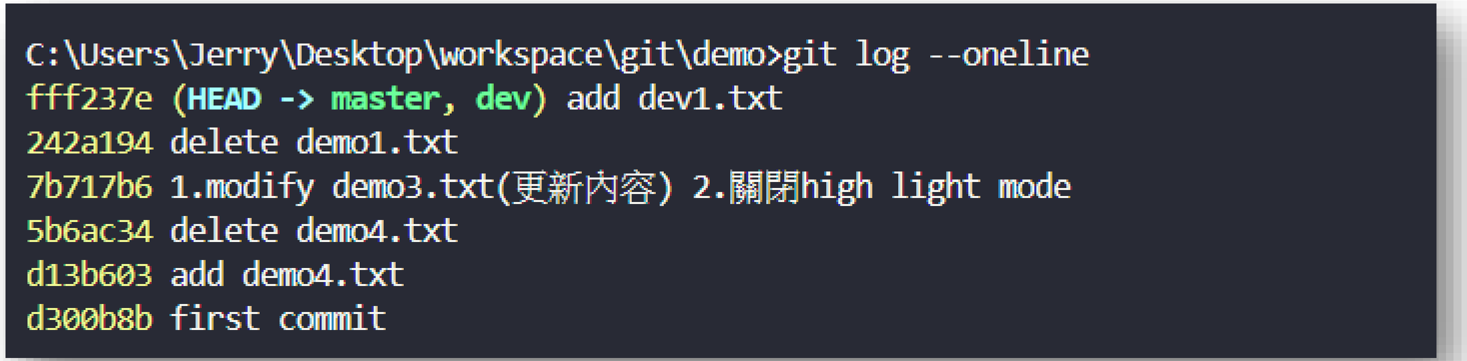


.git > refs > heads > dev

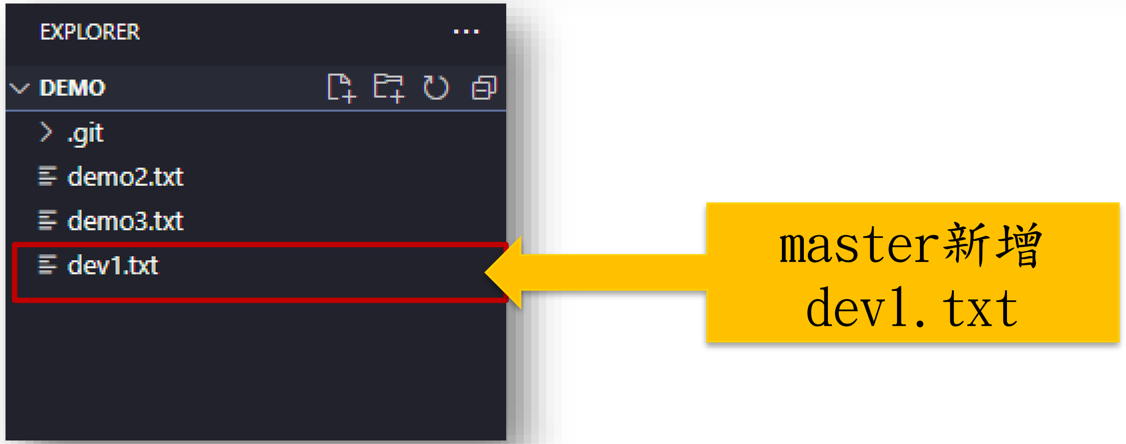
```
a03f1c9b3cfdf467b2e9762e54a60a8d63063f4f

```

簡易分支合併

- git merge <branch-name>
- 需先切換欲合併的主分支(master)



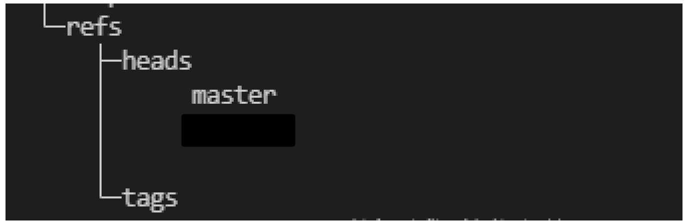
```
C:\Users\Jerry\Desktop\workspace\git\demo>git log --oneline
fff237e (HEAD -> master, dev) add dev1.txt
242a194 delete demo1.txt
7b717b6 1.modify demo3.txt(更新內容) 2.關閉high light mode
5b6ac34 delete demo4.txt
d13b603 add demo4.txt
d300b8b first commit
```



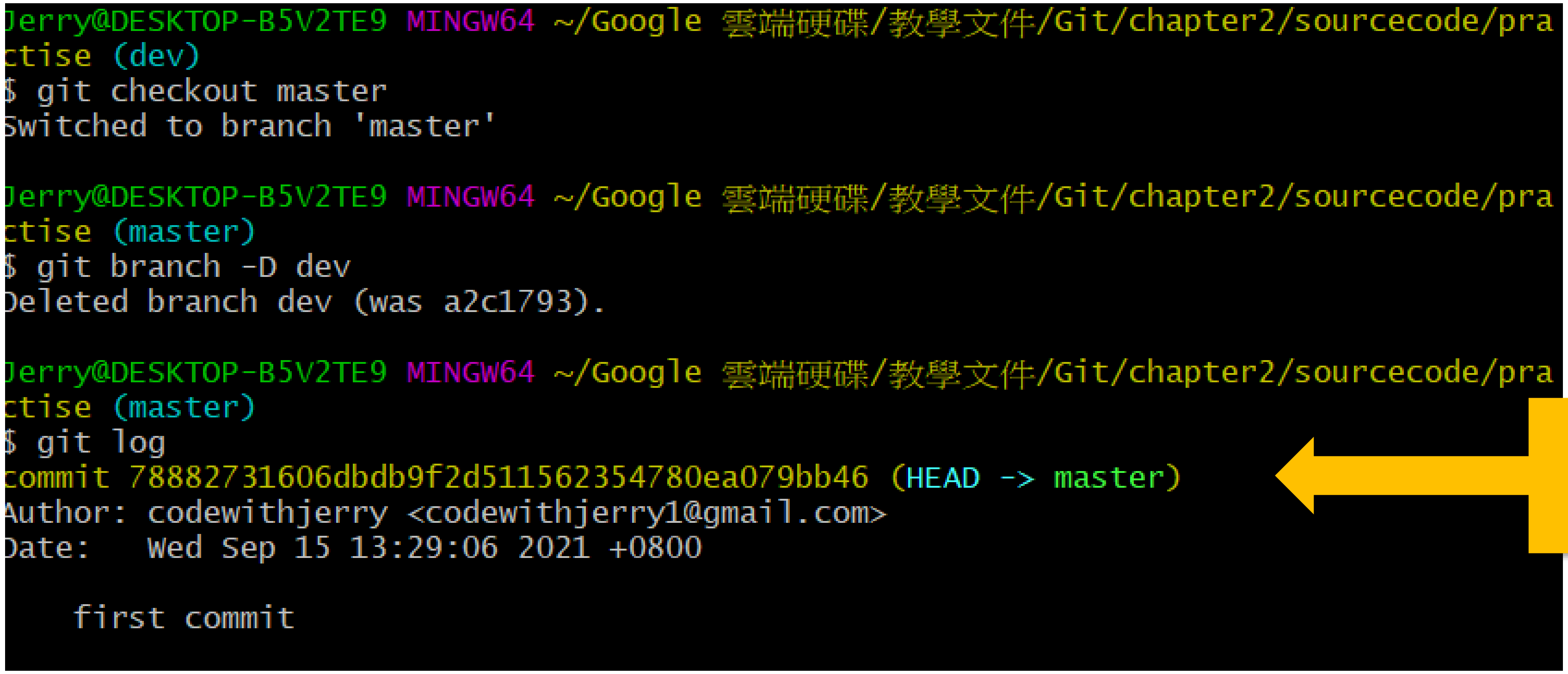
```
EXPLORER ...
DEMO
 > .git
 demo2.txt
 demo3.txt
 dev1.txt
```

master新增
dev1.txt

- 刪除分支
  - git branch –D dev
  - 在該分支上無法進行刪除
  - 需要切換回其他分支(ex. master)



```
└refs
   ├heads
   │     master
   │     [redacted]
   │
   └tags
```



```
Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapter2/sourcecode/pra
ctise (dev)
$ git checkout master
Switched to branch 'master'

Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapter2/sourcecode/pra
ctise (master)
$ git branch -D dev
Deleted branch dev (was a2c1793).

Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapter2/sourcecode/pra
ctise (master)
$ git log
commit 78882731606dbdb9f2d511562354780ea079bb46 (HEAD -> master)
Author: codewithjerry <codewithjerry1@gmail.com>
Date:   Wed Sep 15 13:29:06 2021 +0800

    first commit
```

dev分支消失

- git branch (list and branches we have)
- git branch <branch_name> (create a branch with branch_name, if there already have branch with the name you want to create, it will return error)
- git branch –D <branch_name> (delete branch, can't delete current active branch or branch not existing)
- git checkout (change the current active branch)
- git checkout –b <branch_name> (checkout a branch, will create that branch if it doesn't exsit)
- git branch –m <old_name> <new_name> (rename branch with new name)

建立跟切換分支一次完成

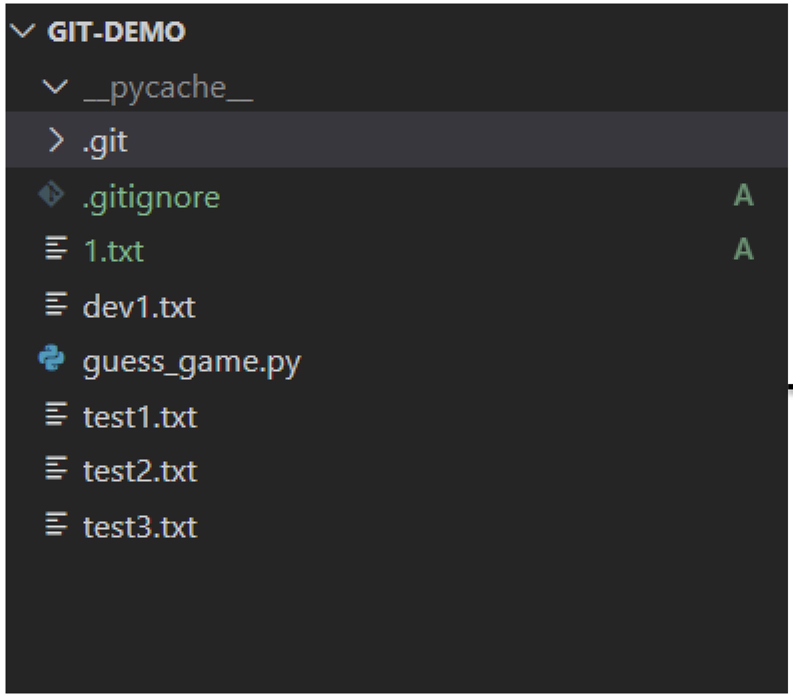
git checkout

```
git checkout commit-object
```

切换到任何commit之上

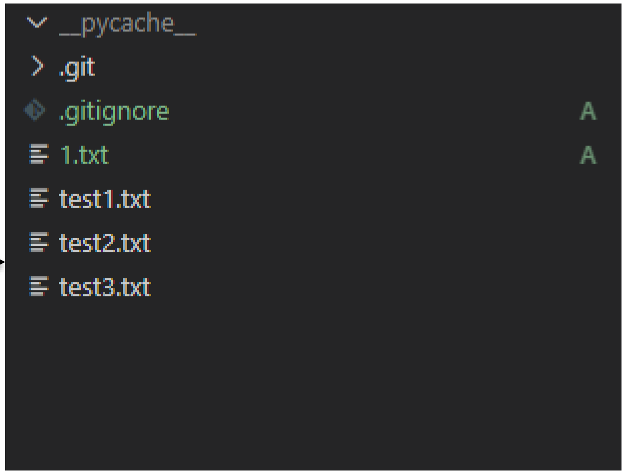
為什麼需要切換 commit-object?

- 回到過去看當時程式碼
- 回到過去修改程式碼(需建立分支)



```
GIT-DEMO
  __pycache__
  .git
  .gitignore        A
  1.txt             A
  dev1.txt
  guess_game.py
  test1.txt
  test2.txt
  test3.txt
```

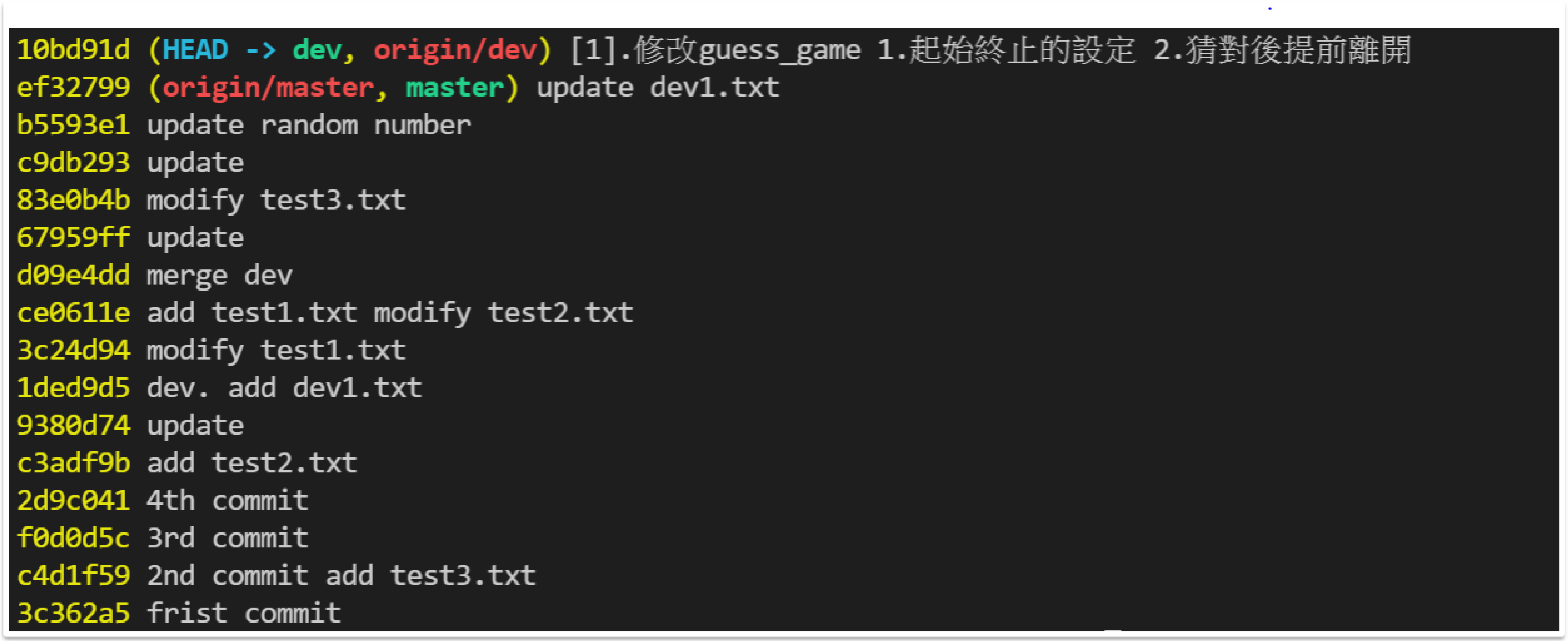
過去



```
  __pycache__
  .git
  .gitignore        A
  1.txt             A
  test1.txt
  test2.txt
  test3.txt
```

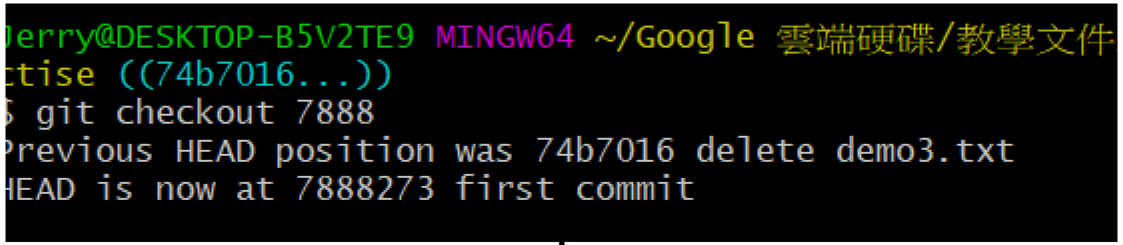
git checkout commit-object

- 切換commit



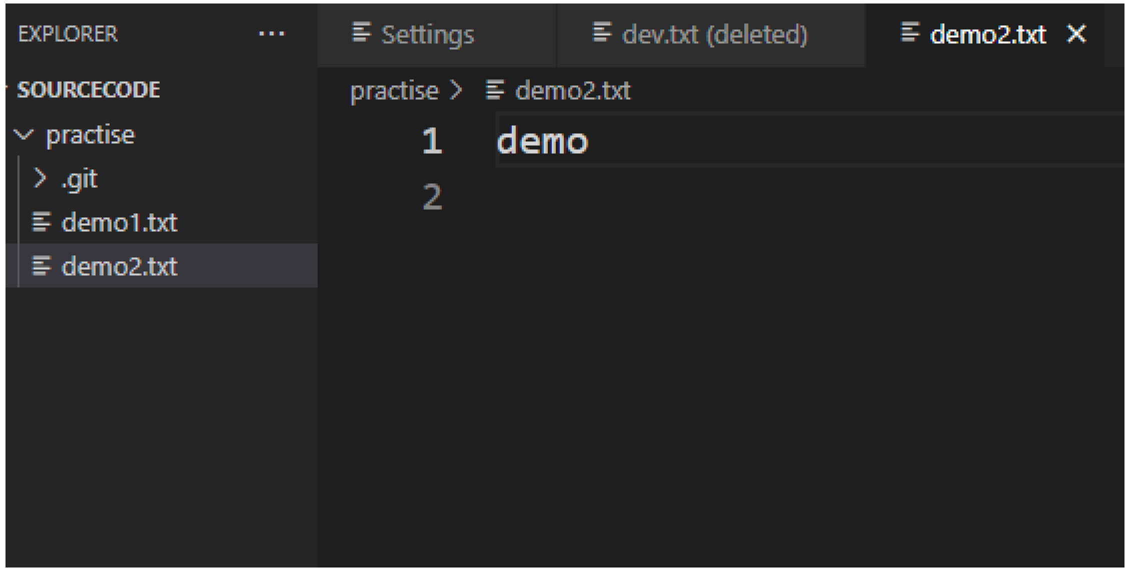
```
10bd91d (HEAD -> dev, origin/dev) [1].修改guess_game 1.起始終止的設定 2.猜對後提前離開
ef32799 (origin/master, master) update dev1.txt
b5593e1 update random number
c9db293 update
83e0b4b modify test3.txt
67959ff update
d09e4dd merge dev
ce0611e add test1.txt modify test2.txt
3c24d94 modify test1.txt
1ded9d5 dev. add dev1.txt
9380d74 update
c3adf9b add test2.txt
2d9c041 4th commit
f0d0d5c 3rd commit
c4d1f59 2nd commit add test3.txt
3c362a5 frist commit
```

▪ git checkout 7888

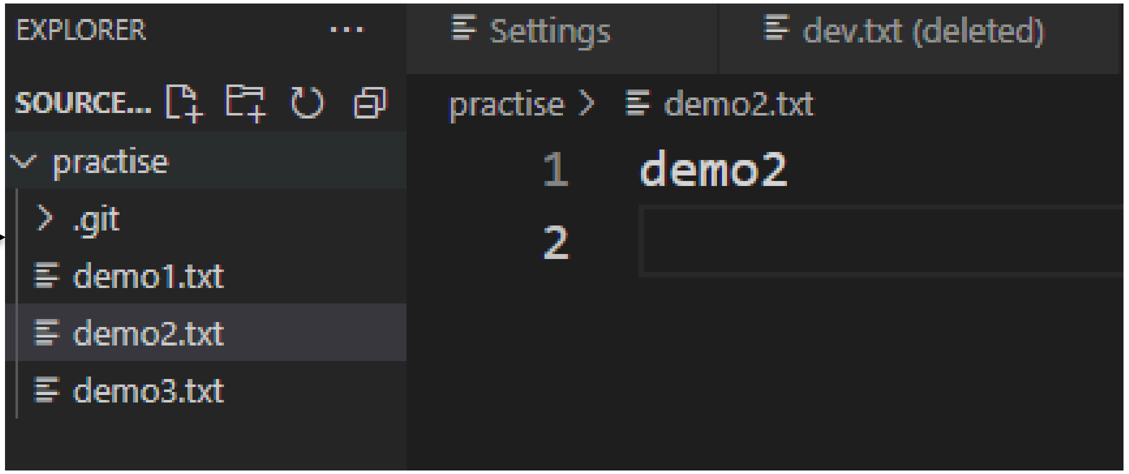


```
Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件
ctise ((74b7016...))
$ git checkout 7888
Previous HEAD position was 74b7016 delete demo3.txt
HEAD is now at 7888273 first commit
```

回到第一個
commit



```
demo
```



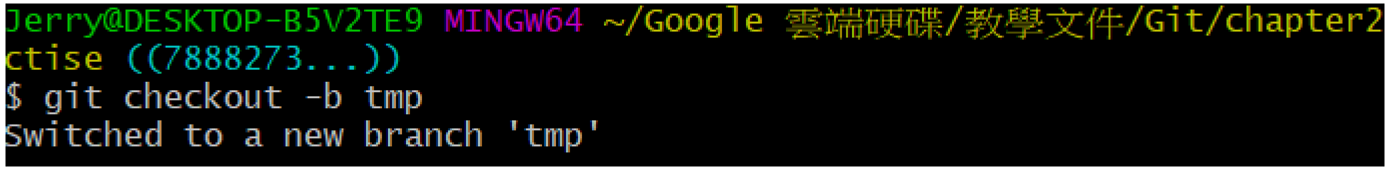
```
demo2
```

回到過去的修改跟新增?

- commit下建立分支
  - git checkout –b tmp
  - 進行檔案修改或新增

直接建立分支跟切換

否則修改不會成立!



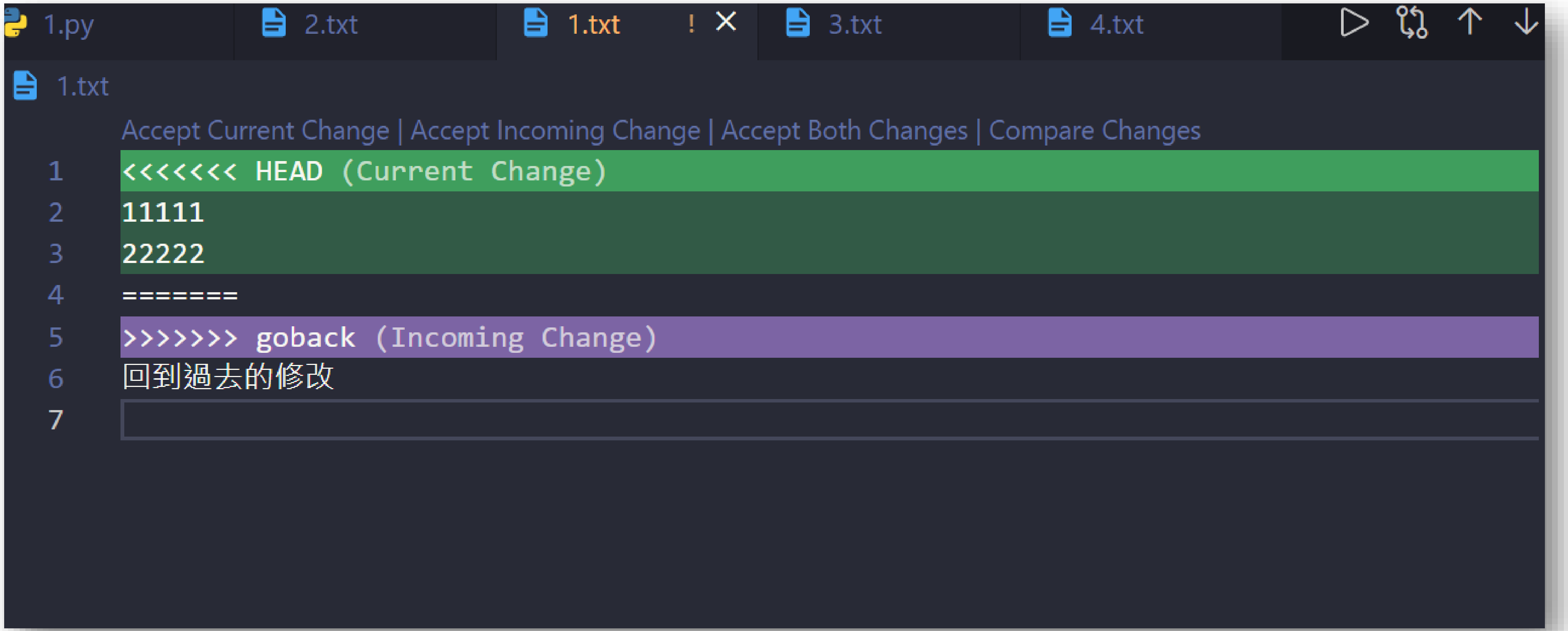
```
Jerry@DESKTOP-B5V2TE9 MINGW64 ~/Google 雲端硬碟/教學文件/Git/chapter2
ctise ((7888273...))
$ git checkout -b tmp
Switched to a new branch 'tmp'
```

- git checkout master
- git merge tmp
- git branch －D tmp

進行分支合併

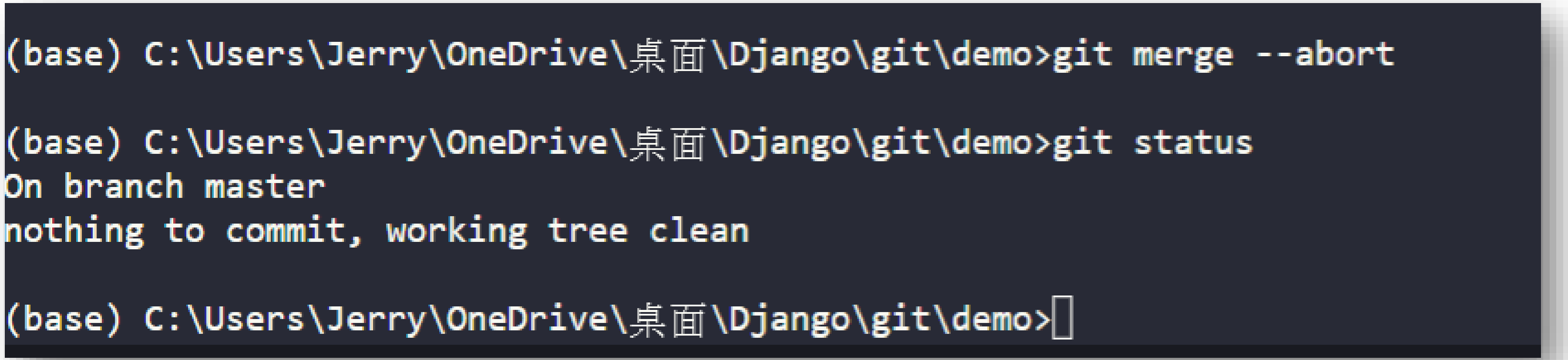
- 回到過去的新增跟修改
  - 修改會發生衝突，需要處理
  - 新增的部分，需要merge

merge 後再 commit


1.py
2.txt
1.txt
3.txt
4.txt
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes

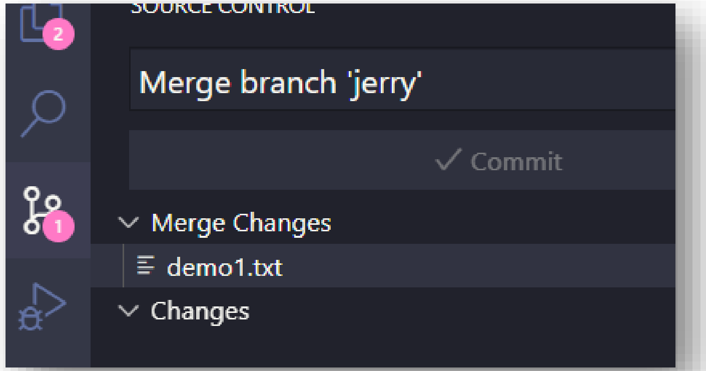
```
<<<<<<< HEAD (Current Change)
11111
22222
=======
>>>>>>> goback (Incoming Change)
回到過去的修改

```

- 反悔合併
  - git merge --abort



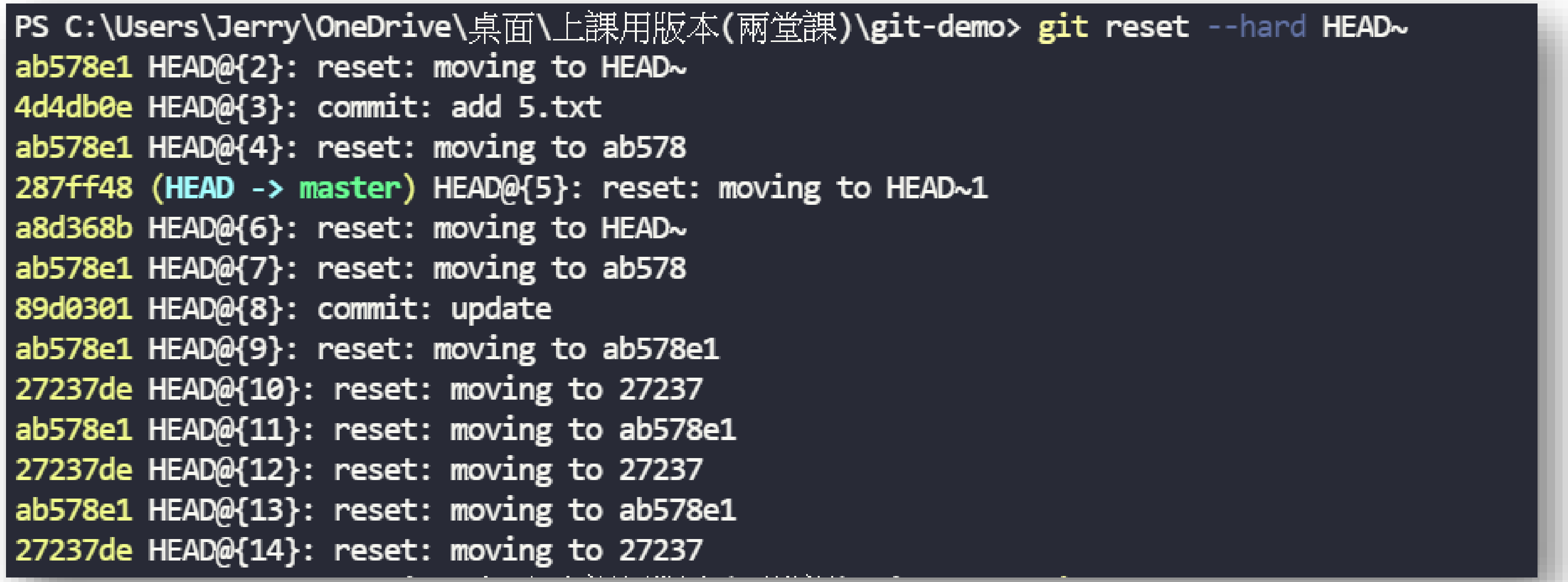
```
(base) C:\Users\Jerry\OneDrive\桌面\Django\git\demo>git merge --abort

(base) C:\Users\Jerry\OneDrive\桌面\Django\git\demo>git status
On branch master
nothing to commit, working tree clean

(base) C:\Users\Jerry\OneDrive\桌面\Django\git\demo>
```

- 使用vscode Source Control
  - 選擇合併項目



SOURCE CONTROL
Merge branch 'jerry'
✓ Commit
Merge Changes
demo1.txt
Changes

- 問題
  - 切換commit迷路怎辦?
    - 使用git checkout master
  - 切換commit後，其他的commit-object?
    - 使用git reflog
- git checkout
  - 恢復修改
  - 切換分支
  - 切換commit



```
PS C:\Users\Jerry\OneDrive\桌面\上課用版本(兩堂課)\git-demo> git reset --hard HEAD~
ab578e1 HEAD@{2}: reset: moving to HEAD~
4d4db0e HEAD@{3}: commit: add 5.txt
ab578e1 HEAD@{4}: reset: moving to ab578
287ff48 (HEAD -> master) HEAD@{5}: reset: moving to HEAD~1
a8d368b HEAD@{6}: reset: moving to HEAD~
ab578e1 HEAD@{7}: reset: moving to ab578
89d0301 HEAD@{8}: commit: update
ab578e1 HEAD@{9}: reset: moving to ab578e1
27237de HEAD@{10}: reset: moving to 27237
ab578e1 HEAD@{11}: reset: moving to ab578e1
27237de HEAD@{12}: reset: moving to 27237
ab578e1 HEAD@{13}: reset: moving to ab578e1
27237de HEAD@{14}: reset: moving to 27237
```

git reset

真正恢復(回到)過去的commit

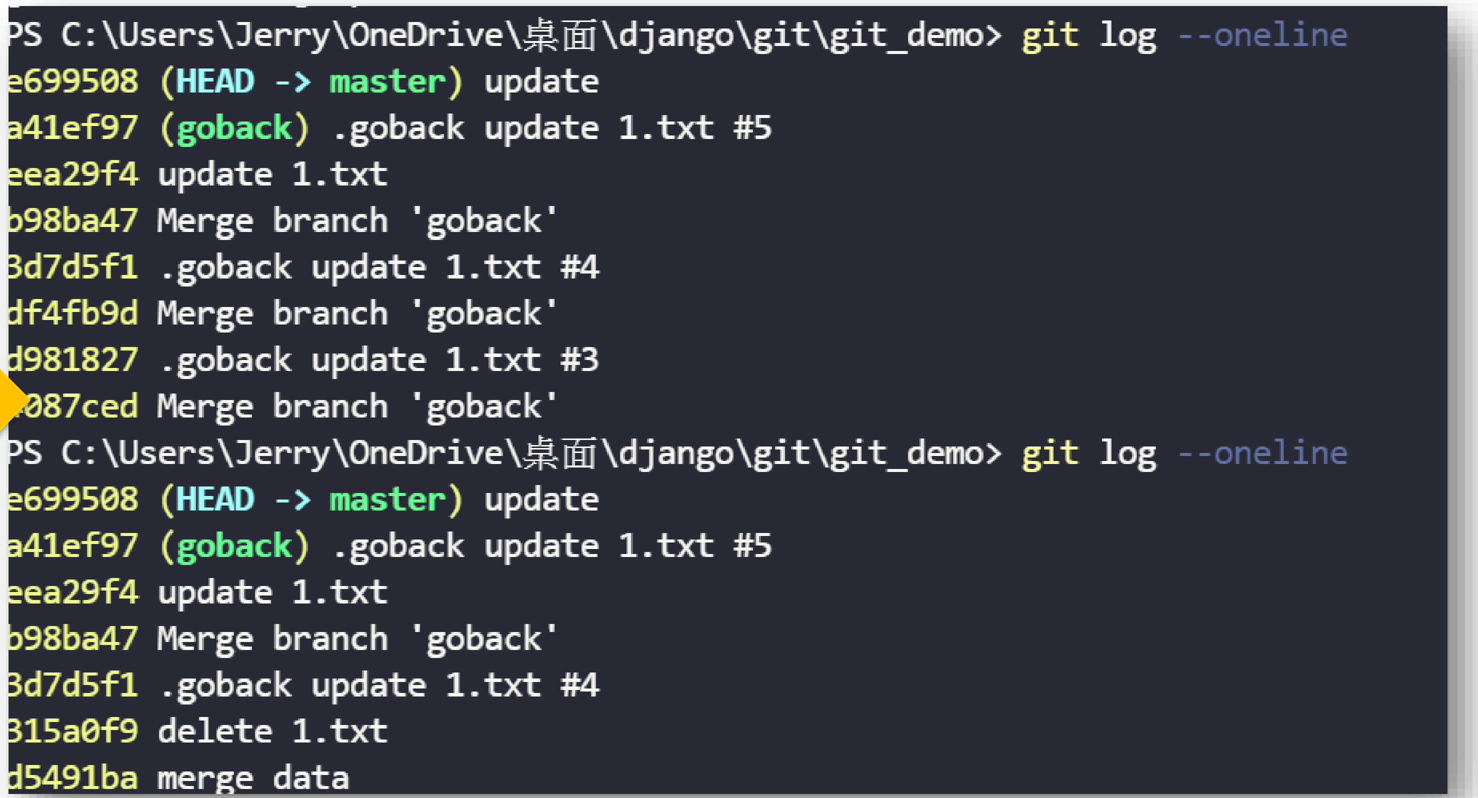
git reset <commit-object>

mixed(預設)/soft/hard

為何需要恢復到某一個commit?

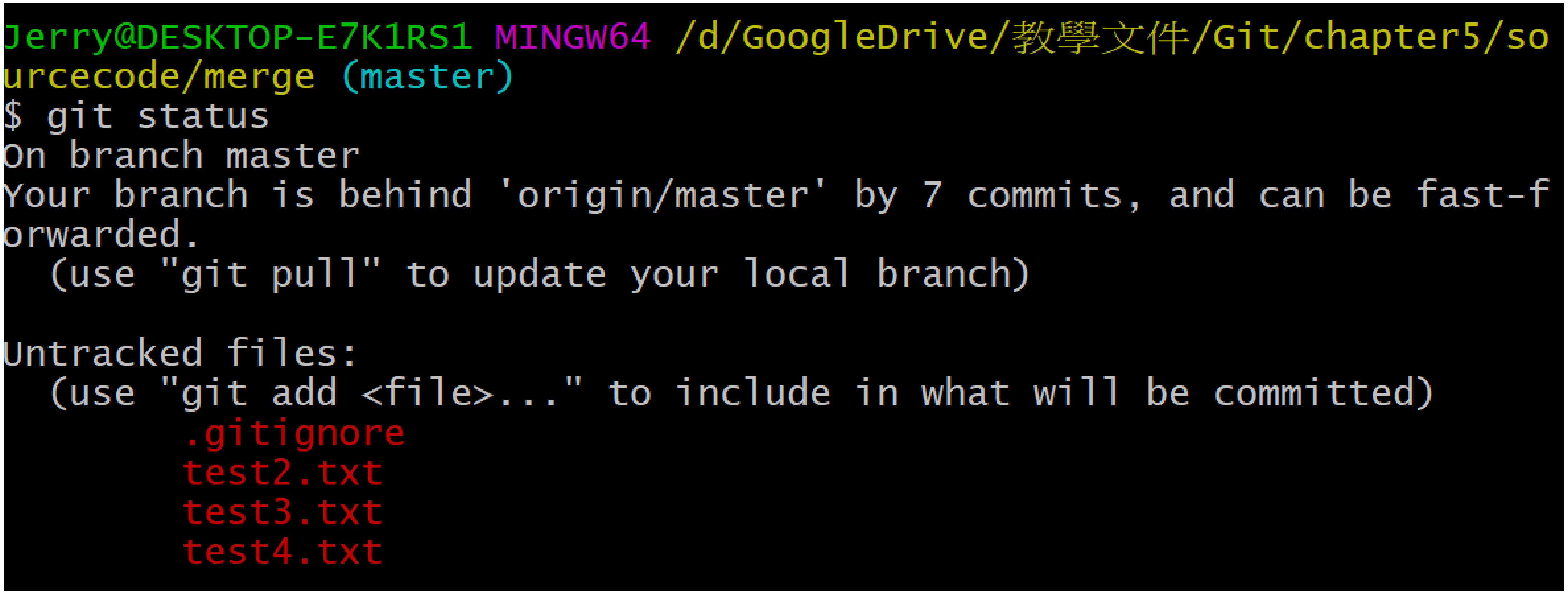
- commit太多需要整理
- 某一個commit是比較合適的版本
- 修改太多但都沒有最新commit版本好
- …

commit太多...



```
PS C:\Users\Jerry\OneDrive\桌面\django\git\git_demo> git log --oneline
e699508 (HEAD -> master) update
a41ef97 (goback) .goback update 1.txt #5
eea29f4 update 1.txt
b98ba47 Merge branch 'goback'
3d7d5f1 .goback update 1.txt #4
df4fb9d Merge branch 'goback'
d981827 .goback update 1.txt #3
087ced Merge branch 'goback'
PS C:\Users\Jerry\OneDrive\桌面\django\git\git_demo> git log --oneline
e699508 (HEAD -> master) update
a41ef97 (goback) .goback update 1.txt #5
eea29f4 update 1.txt
b98ba47 Merge branch 'goback'
3d7d5f1 .goback update 1.txt #4
315a0f9 delete 1.txt
d5491ba merge data
```

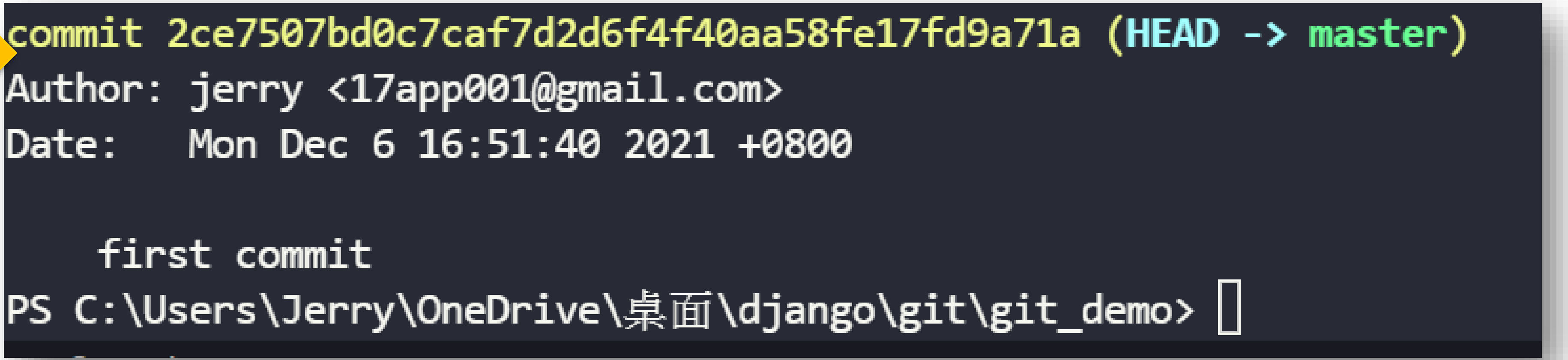
- git reset commit-object （預設是mixed）
  - 後面新增檔案不會消失
    - 該commit後面新增檔案會untracked



```
Jerry@DESKTOP-E7K1RS1 MINGW64 /d/GoogleDrive/教學文件/Git/chapter5/so
urcecode/merge (master)
$ git status
On branch master
Your branch is behind 'origin/master' by 7 commits, and can be fast-f
orwarded.
  (use "git pull" to update your local branch)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        test2.txt
        test3.txt
        test4.txt
```

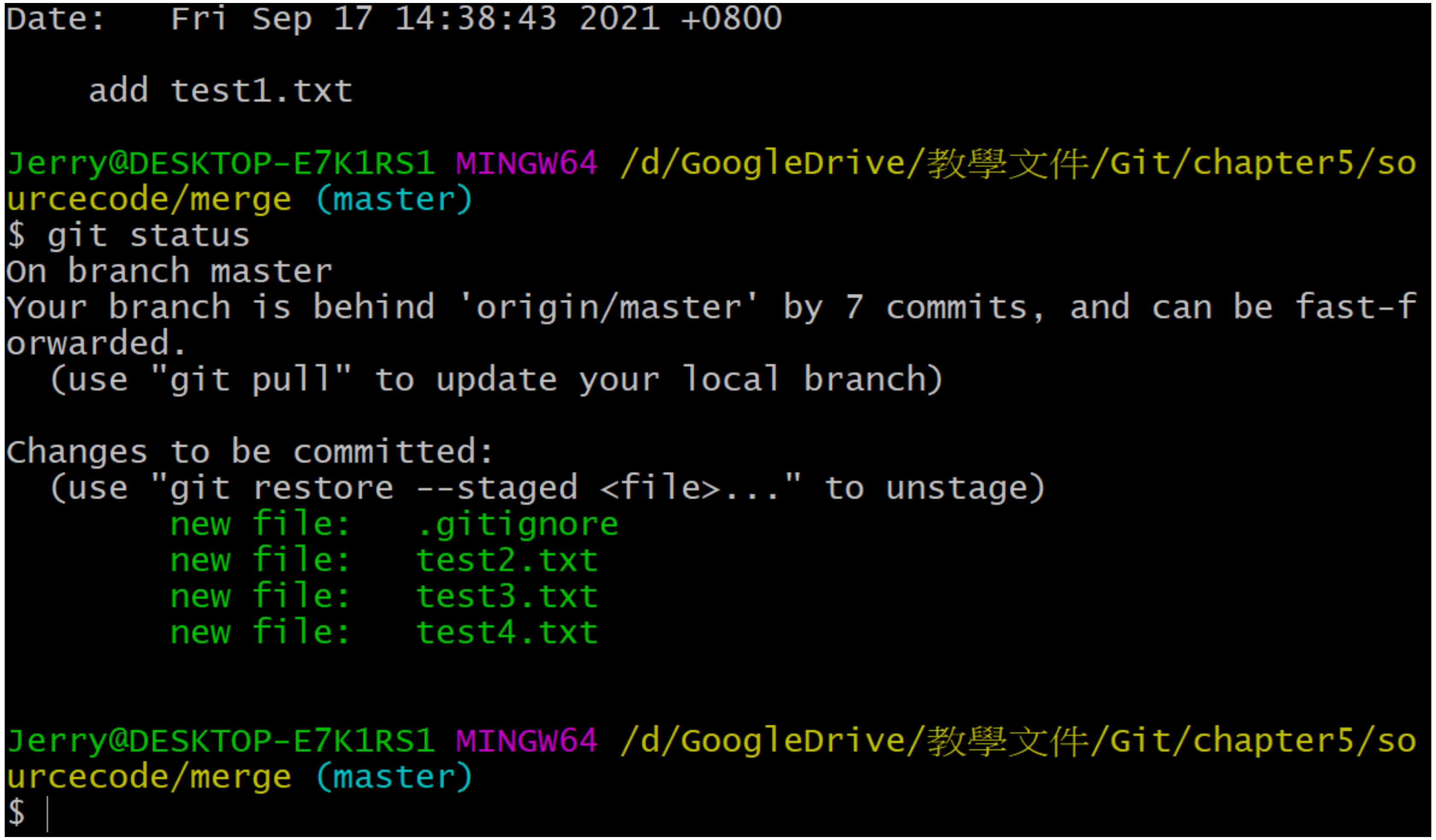
HEAD/master會切換到該commit之上

commit剩下一個



```
commit 2ce7507bd0c7caf7d2d6f4f40aa58fe17fd9a71a (HEAD -> master)
Author: jerry <17app001@gmail.com>
Date:   Mon Dec 6 16:51:40 2021 +0800

    first commit
PS C:\Users\Jerry\OneDrive\桌面\django\git\git_demo> 
```

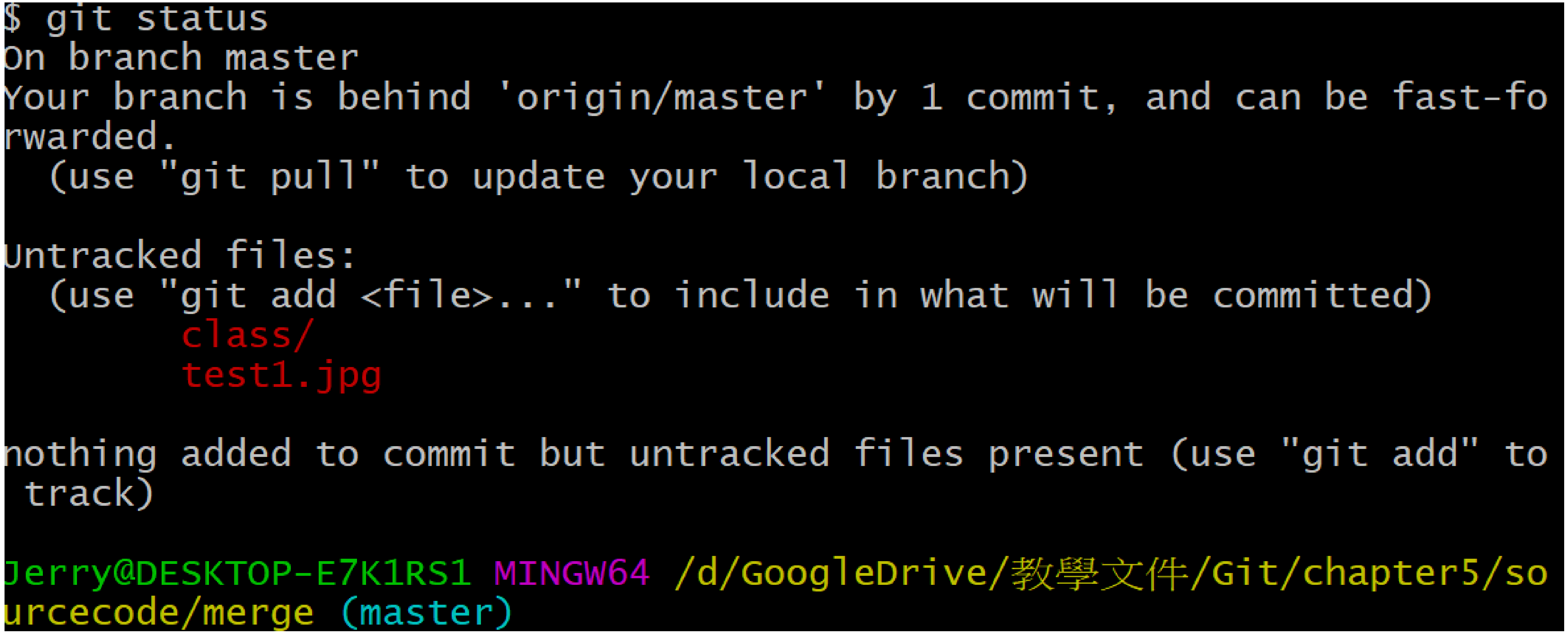
- git reset --soft commit-object
  - 恢復在暫存區，等待commit
  - 可以使用git commit －am “更新後提交”



```
Date:   Fri Sep 17 14:38:43 2021 +0800

    add test1.txt

Jerry@DESKTOP-E7K1RS1 MINGW64 /d/GoogleDrive/教學文件/Git/chapter5/so
urcecode/merge (master)
$ git status
On branch master
Your branch is behind 'origin/master' by 7 commits, and can be fast-f
orwarded.
  (use "git pull" to update your local branch)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   .gitignore
        new file:   test2.txt
        new file:   test3.txt
        new file:   test4.txt


Jerry@DESKTOP-E7K1RS1 MINGW64 /d/GoogleDrive/教學文件/Git/chapter5/so
urcecode/merge (master)
$ 
```

git reset --hard commit-object

- 會移除commit-object後面所有新增的內容
- 要注意使用



```
$ git status
On branch master
Your branch is behind 'origin/master' by 1 commit, and can be fast-fo
rwarded.
  (use "git pull" to update your local branch)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        class/
        test1.jpg

nothing added to commit but untracked files present (use "git add" to
 track)

Jerry@DESKTOP-E7K1RS1 MINGW64 /d/GoogleDrive/教學文件/Git/chapter5/so
urcecode/merge (master)
```

- git reset --hard HEAD
  - 恢復到最新的commit

- git reset --hard HEAD~1
  - 恢復到上一次的commit

~1
上一個

- git reset ORIG_HEAD
  - 反悔reset
    - 恢復到reset前的commit-object
    - 一般就是指原master