

Git tutorial for windows user

cloudtu

<http://cloudtu.blogspot.tw>



Agenda

- 版控系統發展史
- Git設計目標
- 安裝Git
- Git指令教學
- 常用的Git指令
- Git GUI工具教學(TortoiseGit)
- Git基本守則
- 常用協同開發流程
- 參考資料

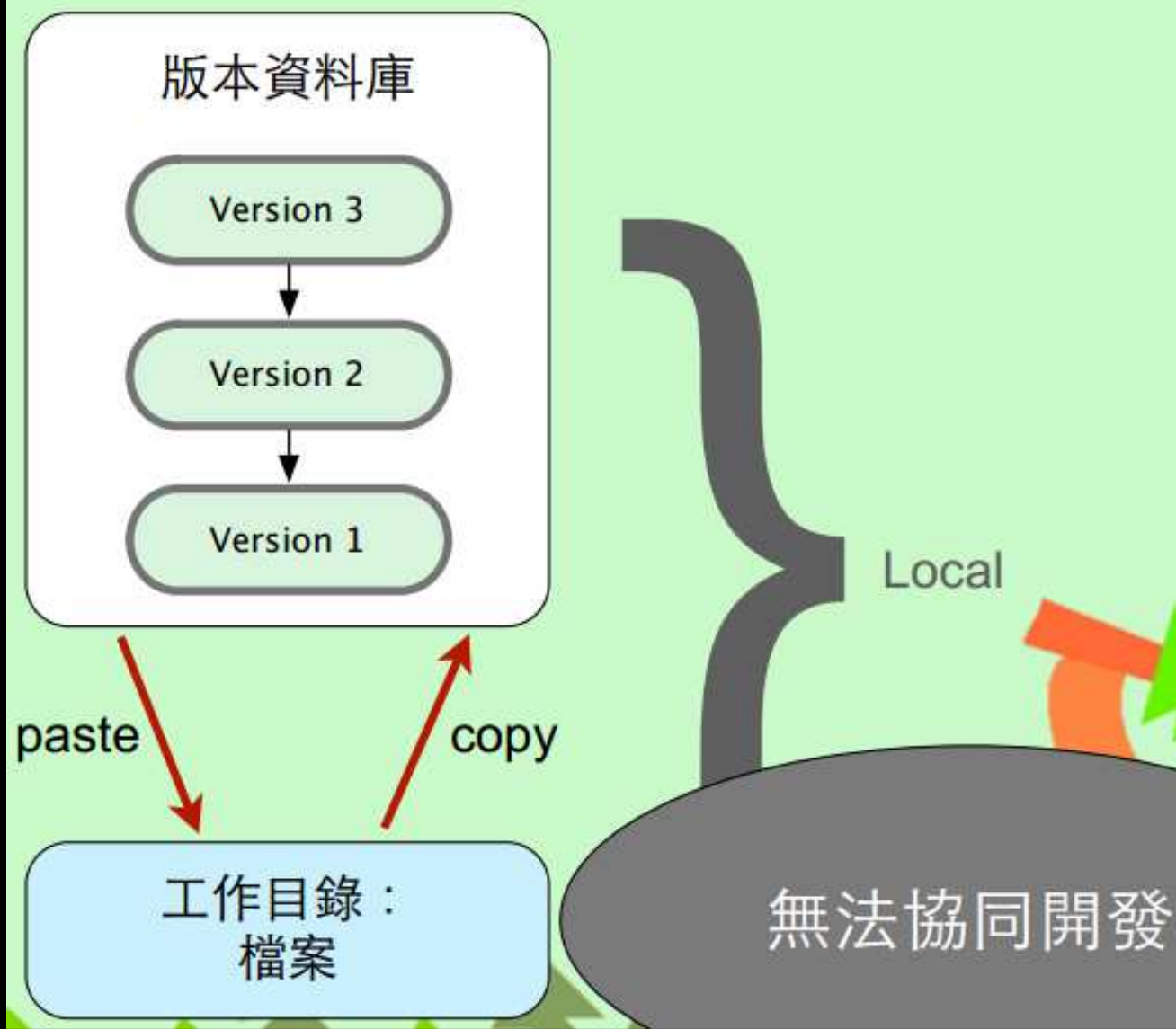
Agenda

- 版控系統發展史
- Git設計目標
- 安裝Git
- Git指令教學
- 常用的Git指令
- Git GUI工具教學(TortoiseGit)
- Git基本守則
- 常用協同開發流程
- 參考資料

版控系統發展史

Local VCS

copy paste 資料夾管理, rcs

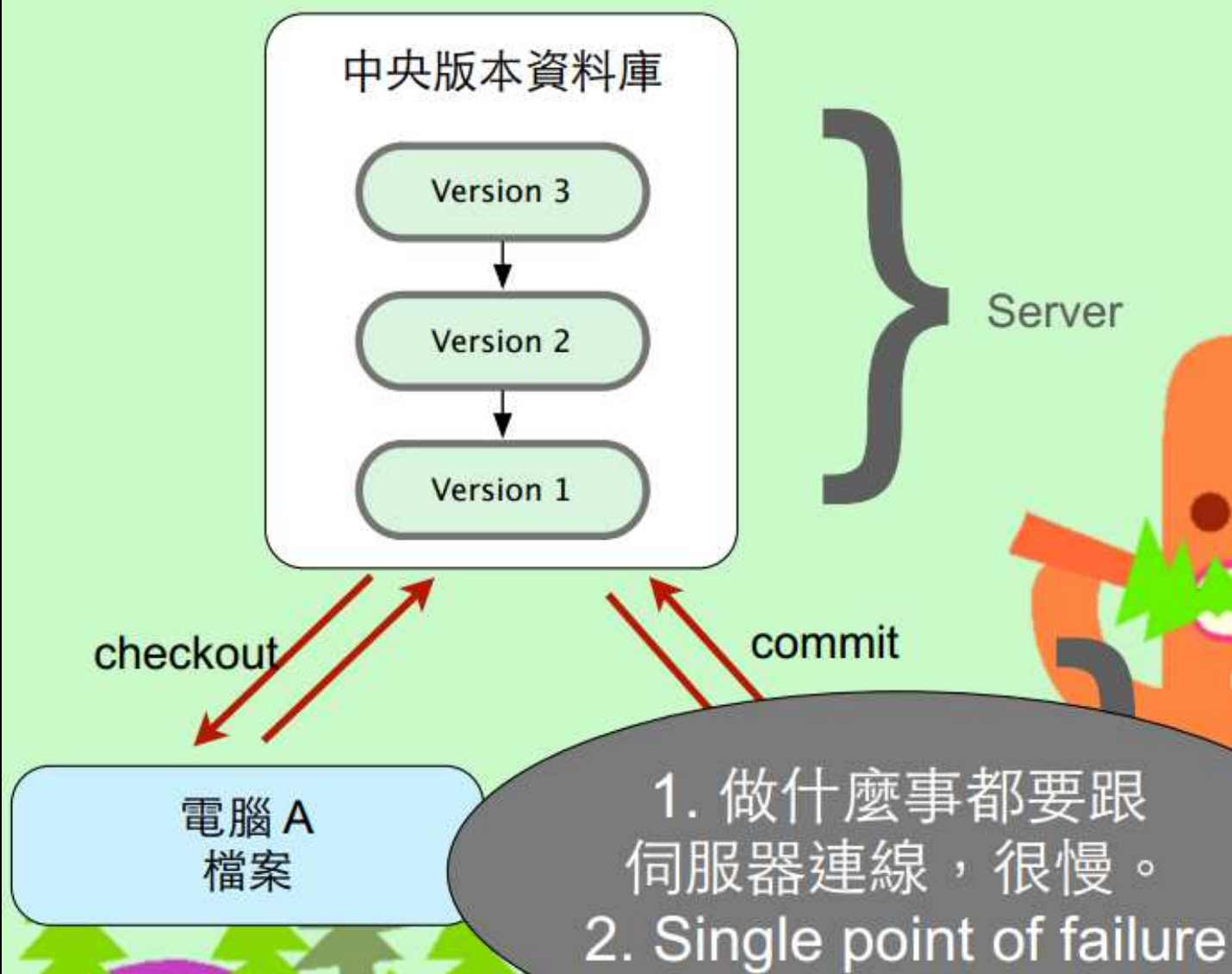


* 取自「Git Tutorial」

版控系統發展史

Centralized VCS

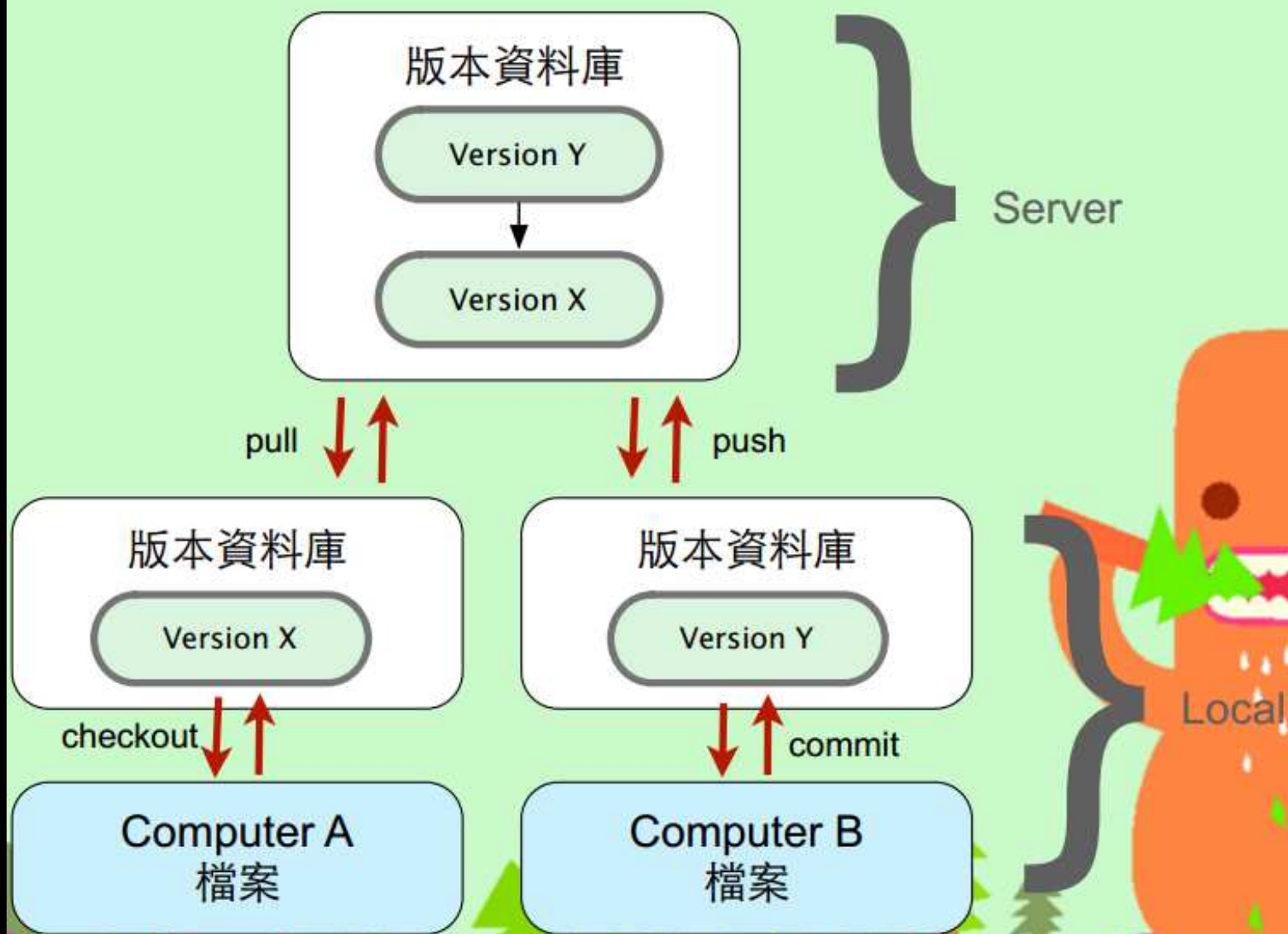
CVS, Subversion, Perforce



版控系統發展史

Distributed VCS

Git, Mercurial(Hg), Bazaar



* 取自「Git Tutorial」

Agenda

- 版控系統發展史
- **Git設計目標**
- 安裝Git
- Git指令教學
- 常用的Git指令
- Git GUI工具教學(TortoiseGit)
- Git基本守則
- 常用協同開發流程
- 參考資料

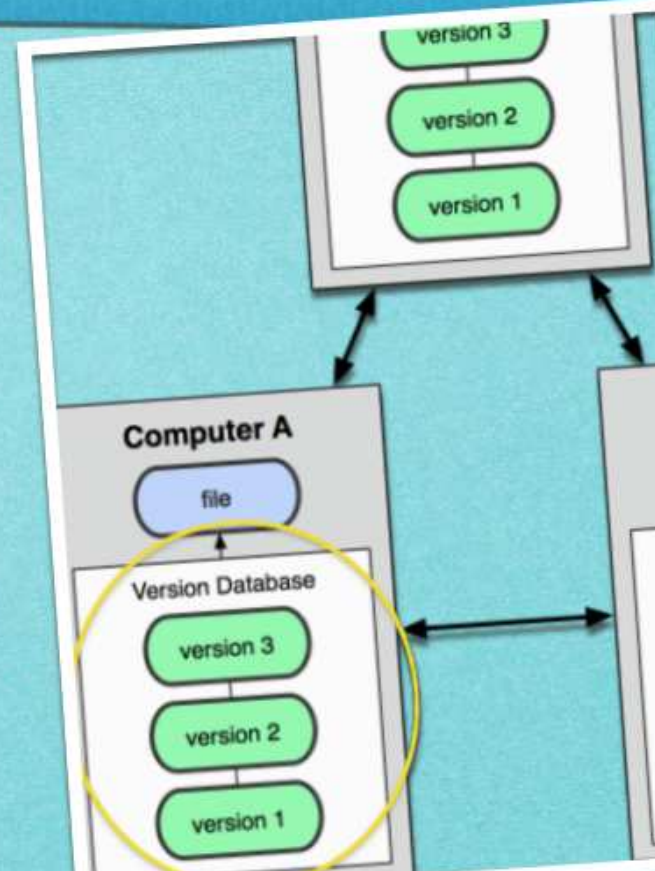
Git設計目標

- 快！非常的快
- 簡單！簡單到只需要 command-line
- 非線性開發！
不要每個人修改什麼就馬上發佈
- 分散式！
沒有網路的時候還可以工作

Git設計目標

Git 幾乎什麼事都在本機進行！

- ▶ 你不只有一份完整的資料庫.....
- ▶ 閱讀版本歷史、提交變更這些動作都可以在本機進行
- ▶ 不需要網路連線也能單獨工作！



Agenda

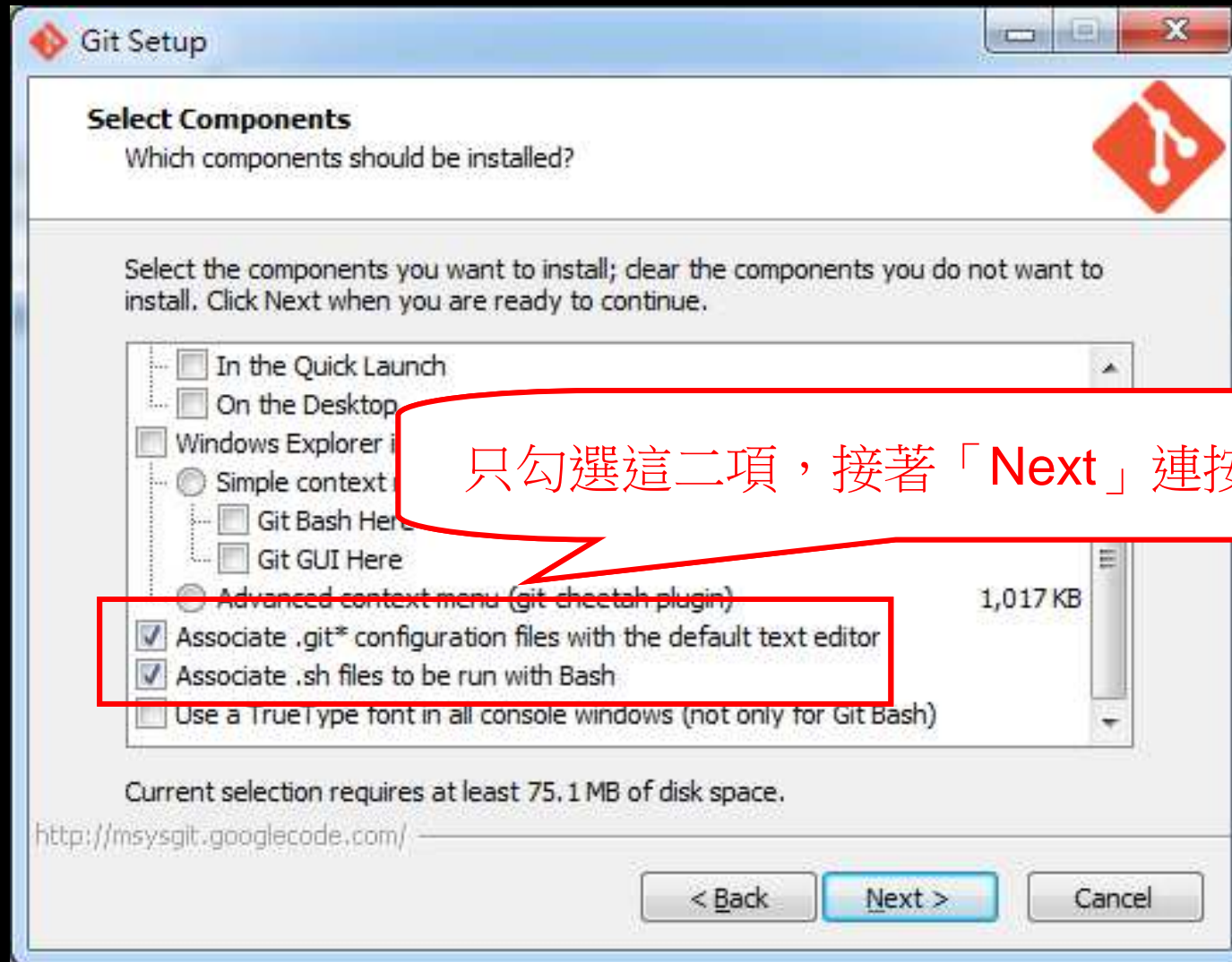
- 版控系統發展史
- Git設計目標
- 安裝Git
- Git指令教學
- 常用的Git指令
- Git GUI工具教學(TortoiseGit)
- Git基本守則
- 常用協同開發流程
- 參考資料

安裝Git

- 安裝流程
 - Step1.安裝msysgit
 - Git-1.7.11-preview20120710.exe
 - <http://msysgit.googlecode.com/files/Git-1.7.11-preview20120710.exe>
 - Step2.安裝tortoisegit
 - TortoiseGit 1.7.12.0(32-bit)
 - <http://tortoisegit.googlecode.com/files/TortoiseGit-1.7.12.0-32bit.msi>
 - TortoiseGit 1.7.12.0(64-bit)
 - <http://tortoisegit.googlecode.com/files/TortoiseGit-1.7.12.0-64bit.msi>
- 補充說明
 - TortoiseGit官方網址
 - <http://code.google.com/p/tortoisegit/>

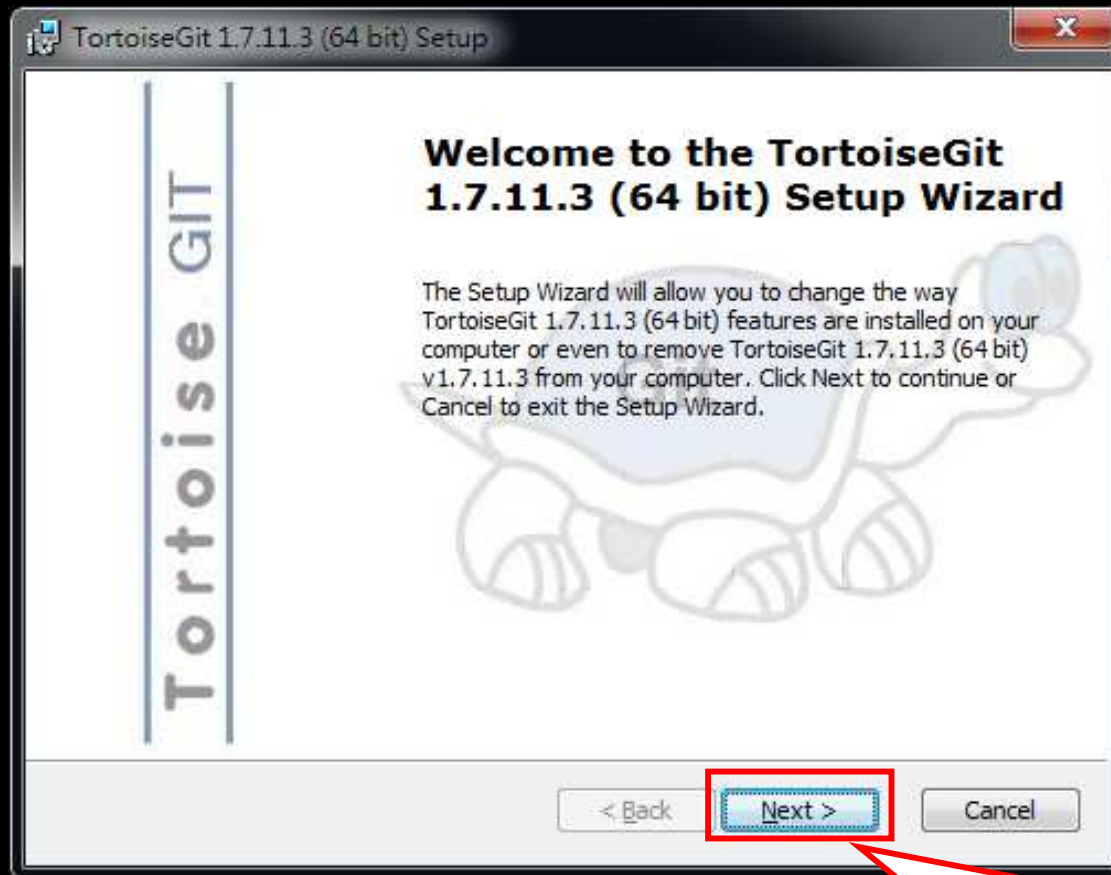
安裝Git

- 安裝msysgit



安裝Git

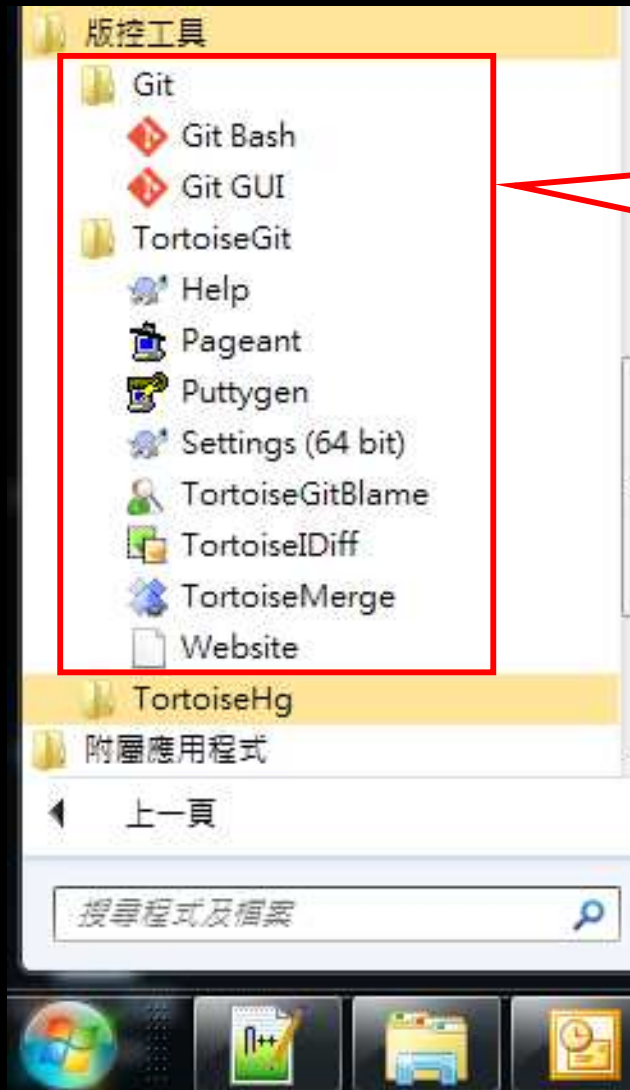
- 安裝tortoisegit



「Next」連接完成安裝，裝完後記的重開機。

安裝Git

- 安裝完成後



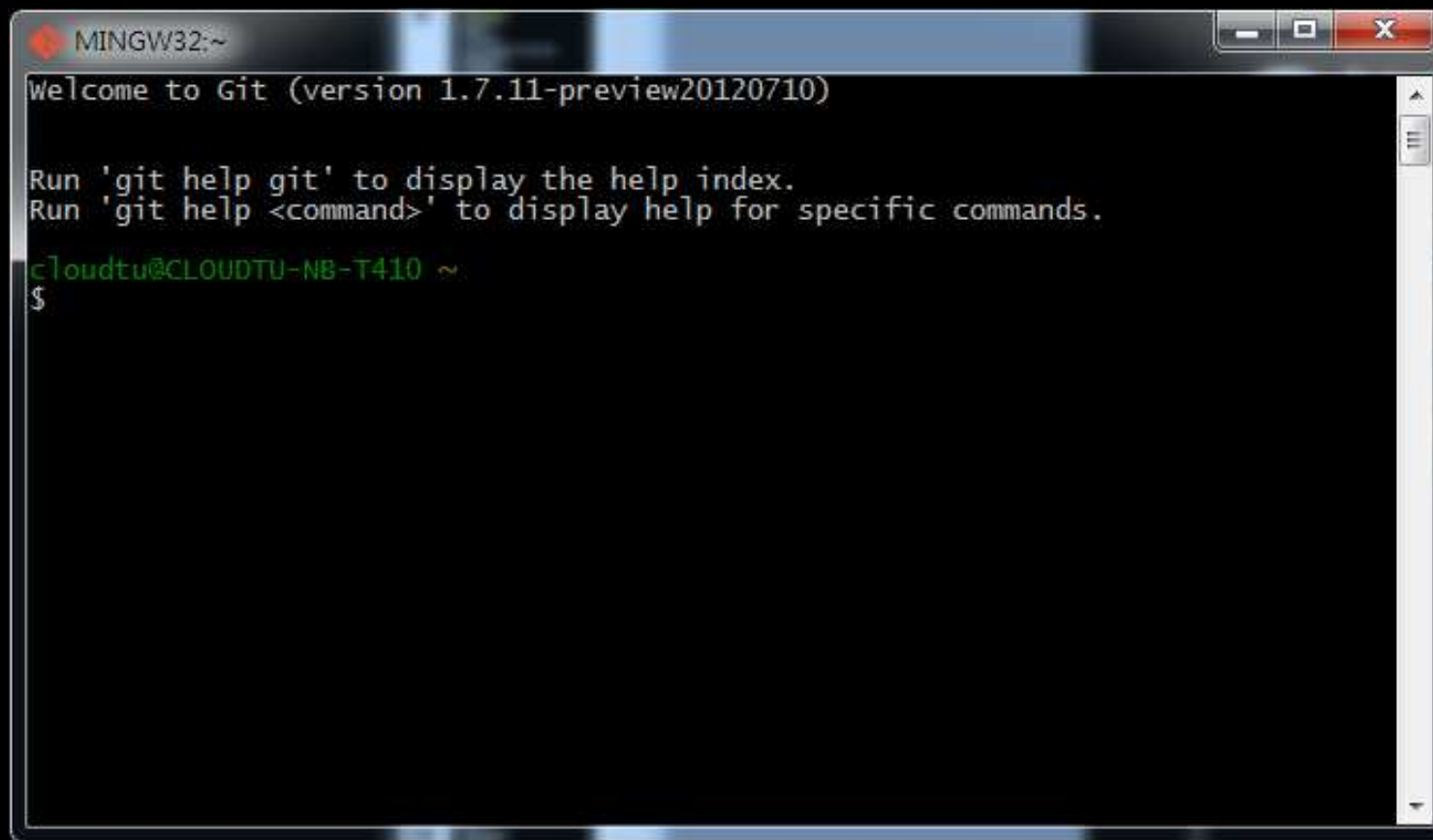
安裝成功的話，程式集中會出現「Git」、「TortoiseGit」二個目錄。

Agenda

- 版控系統發展史
- Git設計目標
- 安裝Git
- **Git指令教學**
- 常用的Git指令
- Git GUI工具教學(TortoiseGit)
- Git基本守則
- 常用協同開發流程
- 參考資料

Git指令教學

執行程式集中的「**Git Bash**」就可以用指令模式操作**Git**了... :)



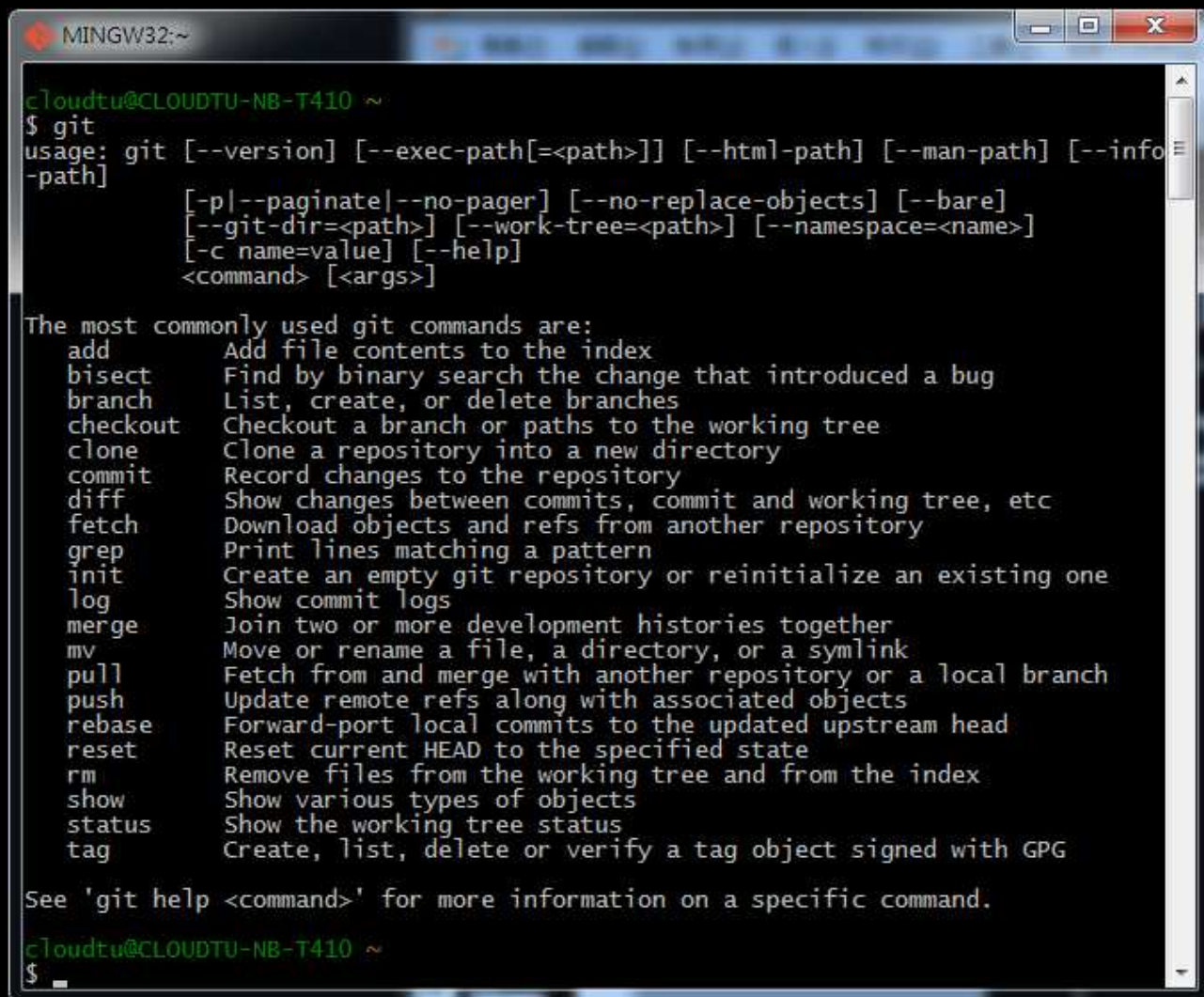
```
MINGW32:~
Welcome to Git (version 1.7.11-preview20120710)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

cloudtu@CLOUDTU-NB-T410 ~
$
```


Git指令教學

最簡單的指令：`git`



```
MINGW32:~
c\loudtu@CLOUDTU-NB-T410 ~
$ git
usage: git [--version] [--exec-path[=<path>]] [--html-path] [--man-path] [--info-
-path]
        [-p|--paginate|--no-pager] [--no-replace-objects] [--bare]
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
        [-c name=value] [--help]
        <command> [<args>]

The most commonly used git commands are:
add          Add file contents to the index
bisect       Find by binary search the change that introduced a bug
branch       List, create, or delete branches
checkout     Checkout a branch or paths to the working tree
clone        Clone a repository into a new directory
commit       Record changes to the repository
diff         Show changes between commits, commit and working tree, etc
fetch        Download objects and refs from another repository
grep         Print lines matching a pattern
init         Create an empty git repository or reinitialize an existing one
log          Show commit logs
merge        Join two or more development histories together
mv           Move or rename a file, a directory, or a symlink
pull         Fetch from and merge with another repository or a local branch
push         Update remote refs along with associated objects
rebase       Forward-port local commits to the updated upstream head
reset        Reset current HEAD to the specified state
rm           Remove files from the working tree and from the index
show         Show various types of objects
status       Show the working tree status
tag          Create, list, delete or verify a tag object signed with GPG

See 'git help <command>' for more information on a specific command.
c\loudtu@CLOUDTU-NB-T410 ~
$
```

Git指令教學(help)

查指令怎麼用：`git help [cmd]`



```
MINGW32:~
c\loudtu@CLOUDTU-NB-T410 ~
$ git help log
Launching default browser to display HTML ...
c\loudtu@CLOUDTU-NB-T410 ~
$
```



Git指令教學(config)

- 設定Git系統參數
 - 設定自己在Git上的名字跟mail (用來區別自己和他人)
`git config --global user.name "my_name"`
`git config --global user.email "my_name@email.com"`
 - 查詢系統設定值：`git config --global -l`

```
cloudtu@CLOUDTU-NB-T410 ~  
$ git config --global -l  
user.name=cloudtu  
user.email=cloud.tu@gmail.com
```

Git指令教學(init)

建立repository : `git init .`

```
cloudtu@CLOUDTU-NB-T410 ~  
$ cd /z
```

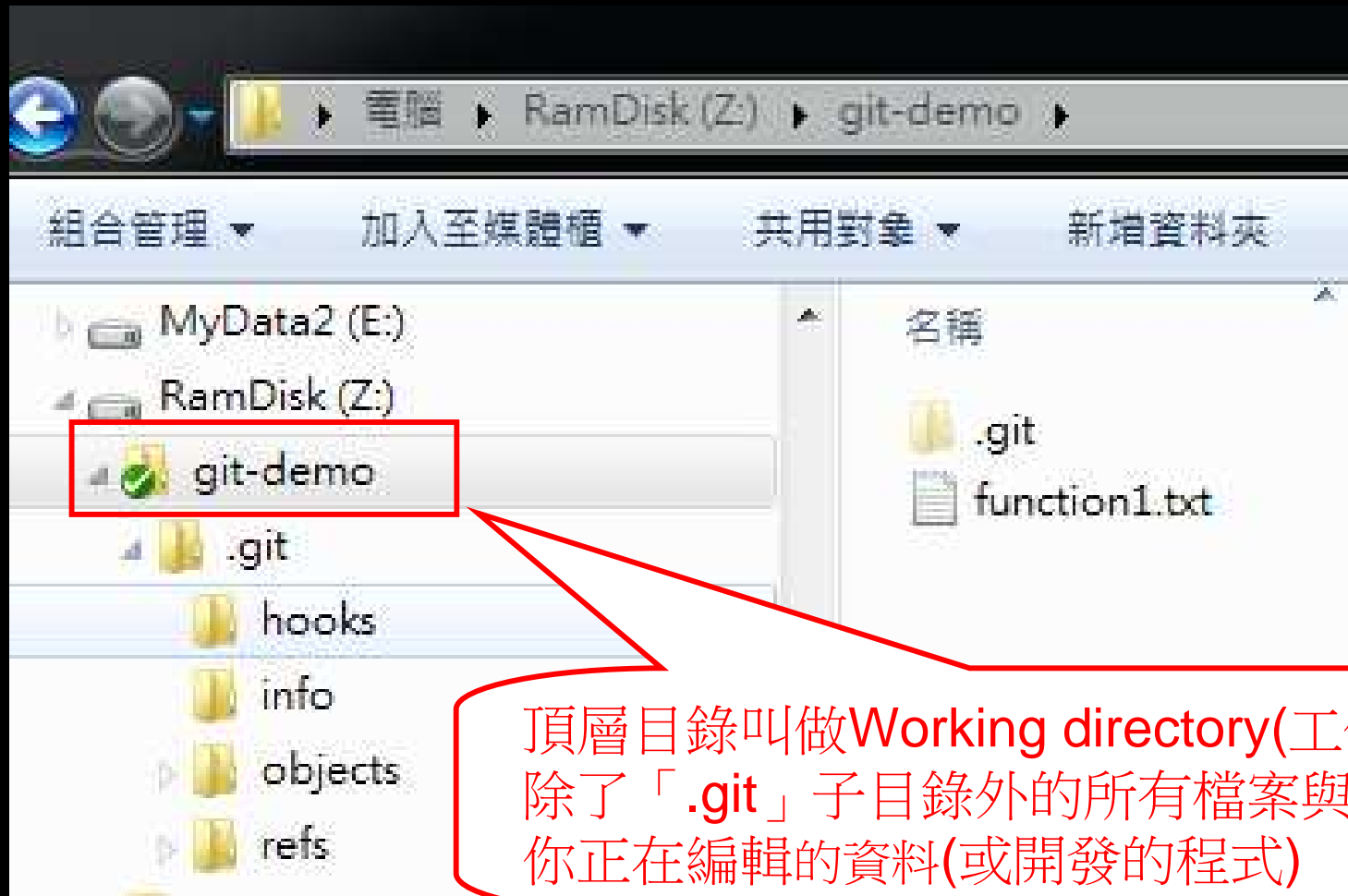
```
cloudtu@CLOUDTU-NB-T410 /z  
$ cd git-demo/
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo  
$ git init .  
Initialized empty Git repository in z:/git-demo/.git/
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)  
$
```

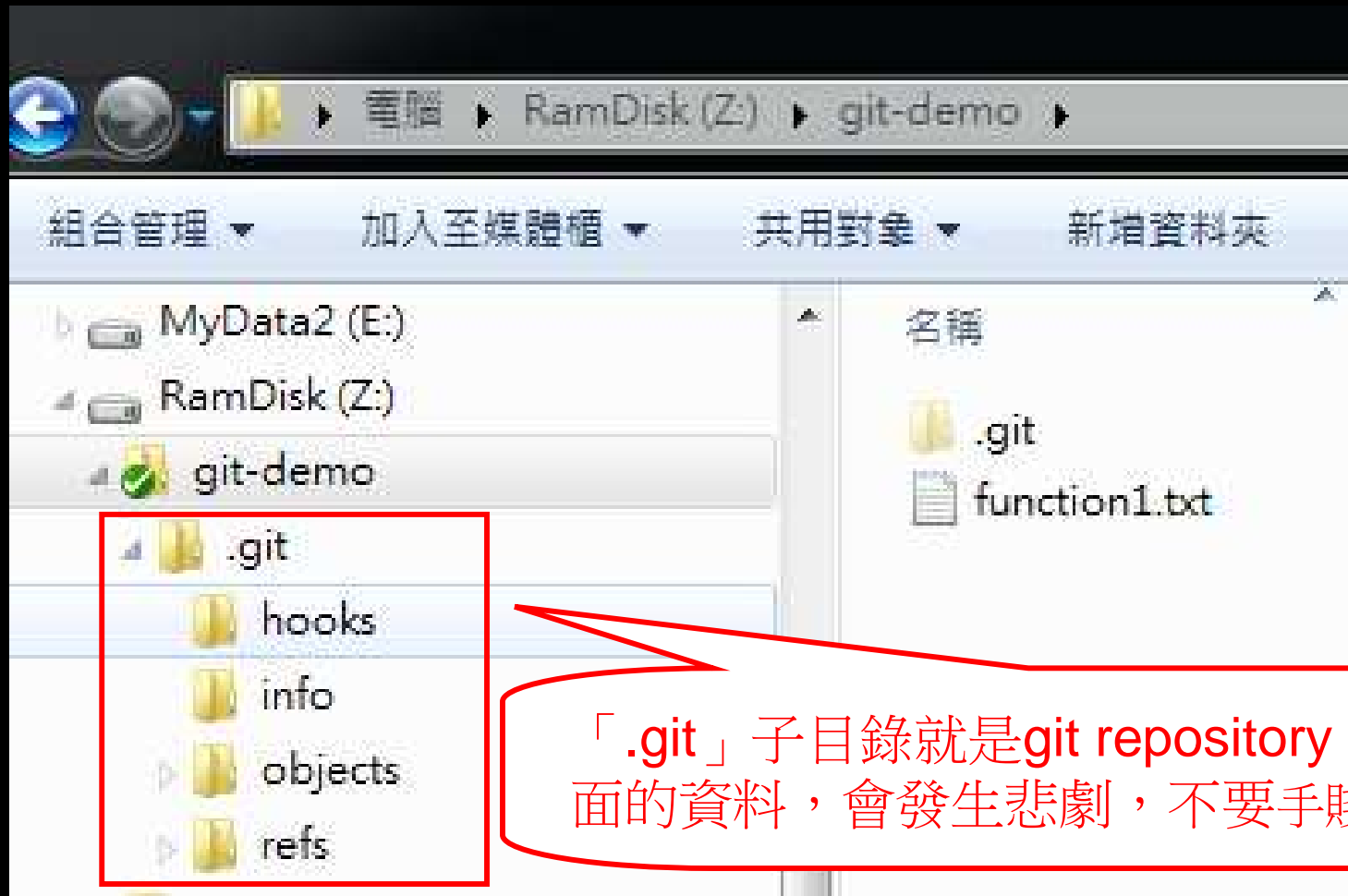
Git指令教學(init)

Repository結構



Git指令教學(init)

Repository結構



Git指令教學(add/commit)

- 將working directory資料送進repository

`git add .`

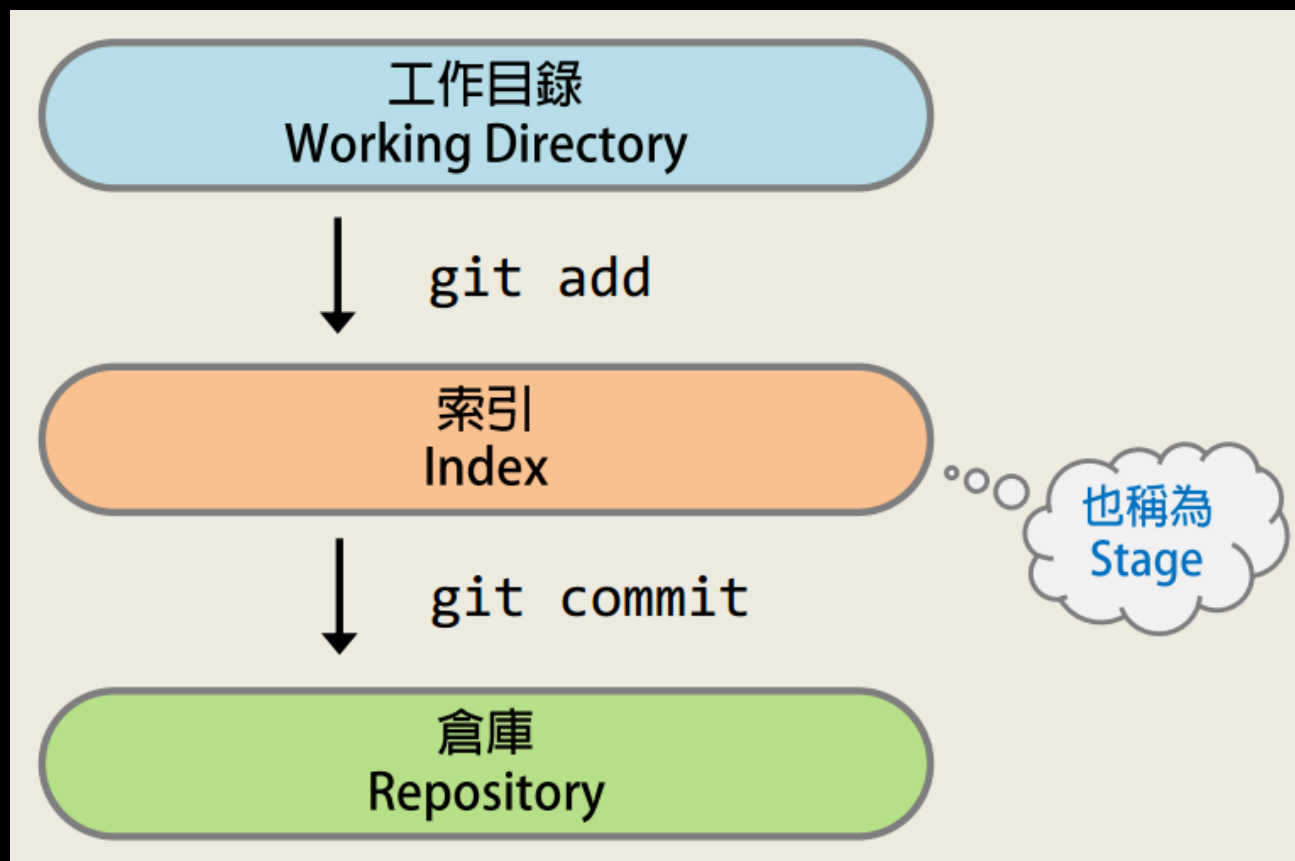
`git commit -a -m "my comment"`

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git add .
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git commit -a -m "add function1"
[master (root-commit) 3d2c234] add function1
1 file changed, 1 insertion(+)
create mode 100644 function1.txt
```

Git指令教學(add/commit)

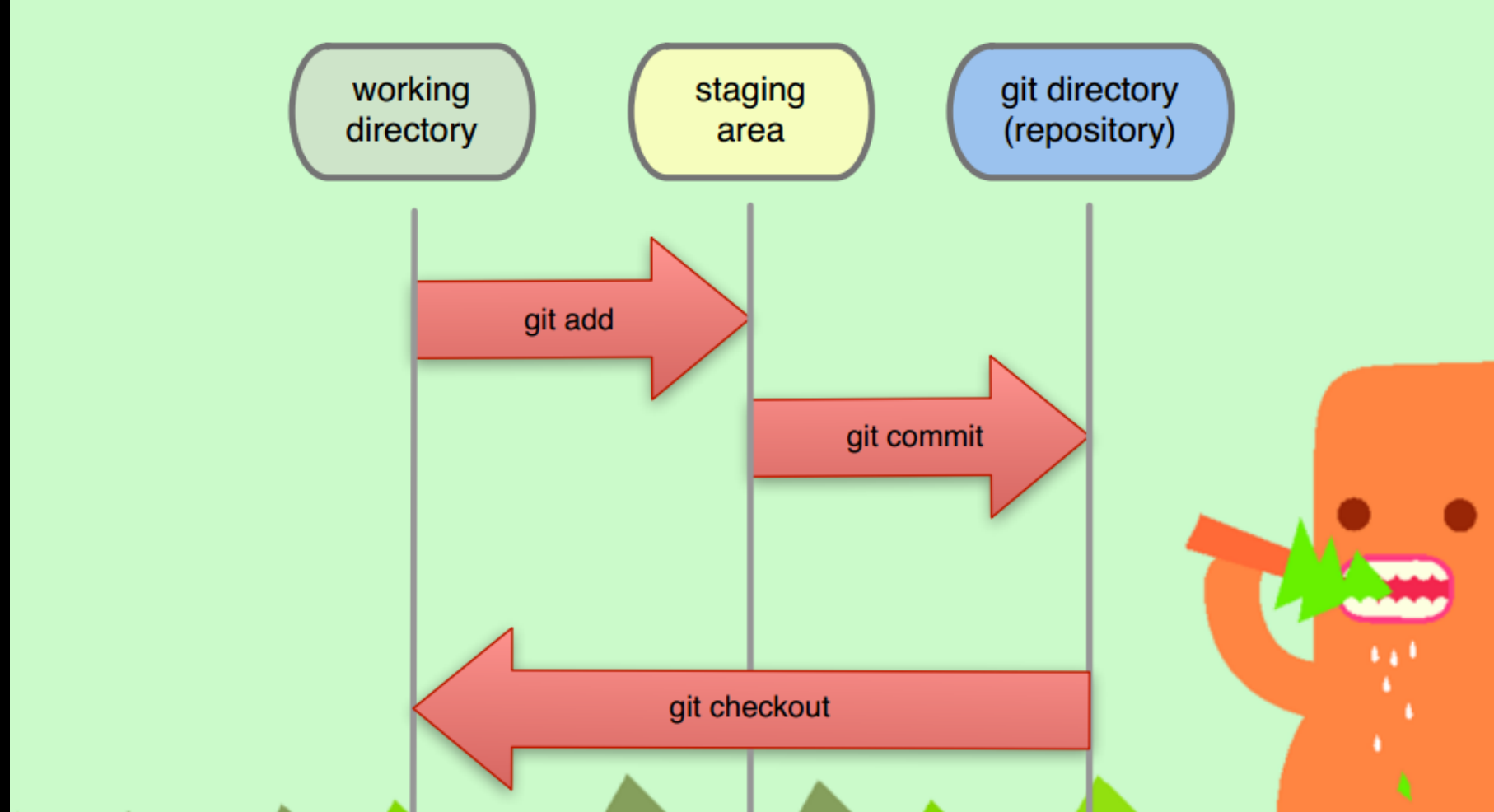
將working directory資料送進repository



Git指令教學(add/commit)

將working directory資料送進repository

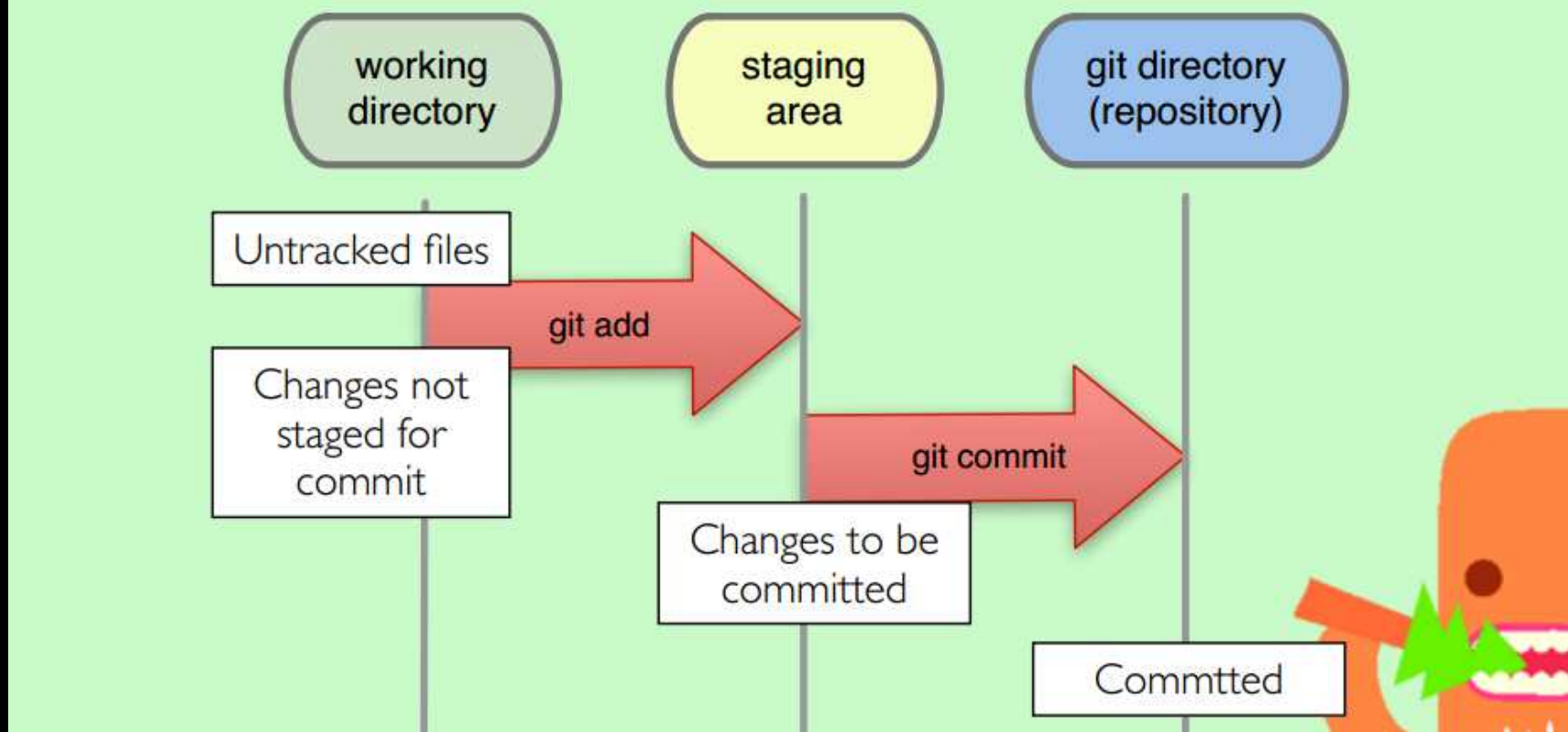
三種區域: Working tree/Staging Area/Repository



Git指令教學(add/commit)

將working directory資料送進repository

Working tree 裡的檔案有四種狀態



Git指令教學(add/commit)

- 將working directory資料送進repository
 - `git add .`
 - 將working directory裡「新增」與「修改」的檔案加到staging area
 - `git commit -a -m "my comment"`
 - 將working directory裡「修改」與「刪除」的檔案加到staging area，接著把所有staging area的資料給commit到repository
 - 上面二個指令並用就能確保每次的commit都包含working directory裡所有「新增」、「修改」與「刪除」的檔案

Git指令教學(add/commit)

- **commit**基本原則
 - 適當的粒度/相關性/獨立性
 - 以一個小功能、小改進或一個bug fixed為單位
 - 對應的unit test程式在同一個commit
 - 無相關的修改不在同一個commit
 - 語法錯誤的半成品程式不能commit
 - **commit**訊息很重要
 - 第一行寫摘要
 - 有需要寫實作細節的話，放第二行之後

Git指令教學(status)

查看檔案狀態：`git status`

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
```

```
$ ls -l
```

```
total 1
```

```
-rw-r--r--  1 cloudtu  Administ    10 Aug 20 16:48 function1.txt
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
```

```
$ git status
```

```
# On branch master
```

```
#
```

```
# Initial commit
```

```
#
```

```
# Untracked files:
```

```
#   (use "git add <file>..." to include in what will be committed)
```

```
#
```

```
#       function1.txt
```

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

新增function1.txt檔案

Git指令教學(status)

查看檔案狀態：`git status`

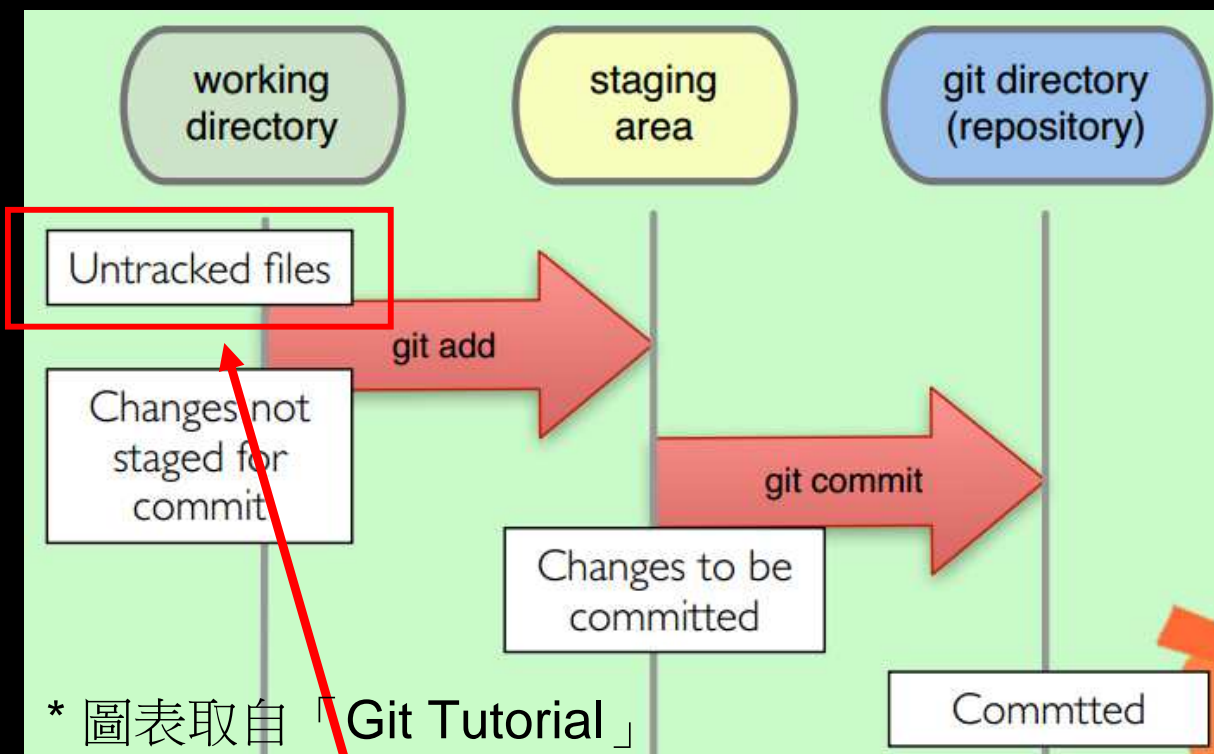
```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ ls -l
total 1
-rw-r--r--  1 cloudtu  Administ  10 Aug 20 16:48 function1.txt

cloudtu@CLOUDTU-NB-T410 /z/
$ git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       function1.txt
nothing added to commit but untracked files present (use "git add" to track)
```

檔案狀態是"Untracked files"，這是什麼
東東？

Git指令教學(status)

查看檔案狀態：**git status**

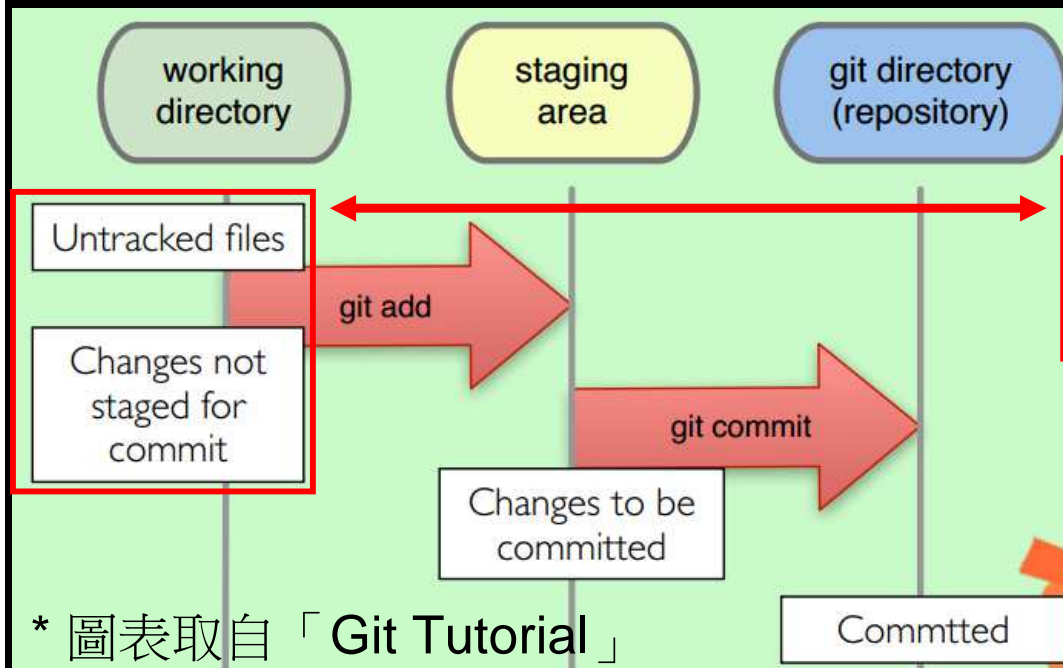


```
cloudtu@CLOUDTU-1B-T410 /z/git-demo (master)
$ git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       function1.txt
nothing added to commit but untracked files present (use "git add" to track)
```

想起來了嗎？

Git指令教學(status)

查看檔案狀態：**git status**



```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       function1.txt
nothing added to commit but untracked files present
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git add .

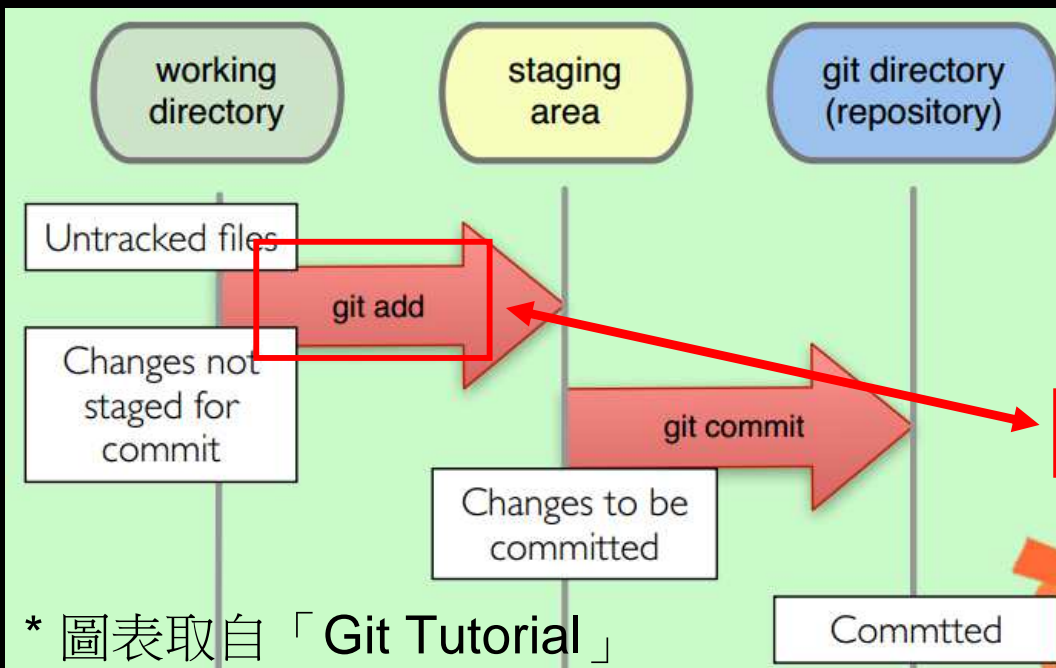
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   function1.txt
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git commit -a -m "add function1"
[master (root-commit) d175788] add function1
1 file changed, 1 insertion(+)
create mode 100644 function1.txt
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
nothing to commit (working directory clean)
```


Git指令教學(status)

查看檔案狀態：**git status**



```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       function1.txt
nothing added to commit but untracked files present
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git add .
```

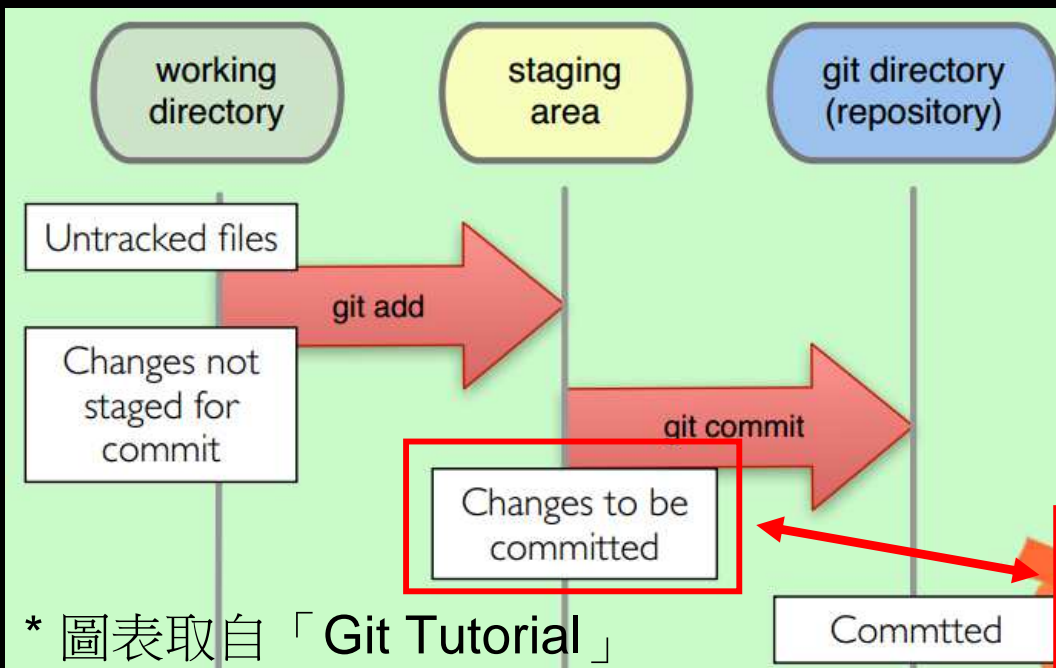
```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   function1.txt
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git commit -a -m "add function1"
[master (root-commit) d175788] add function1
1 file changed, 1 insertion(+)
create mode 100644 function1.txt
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
nothing to commit (working directory clean)
```

Git指令教學(status)

查看檔案狀態：**git status**



```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       function1.txt
nothing added to commit but untracked files present (use "git add" to track)
```

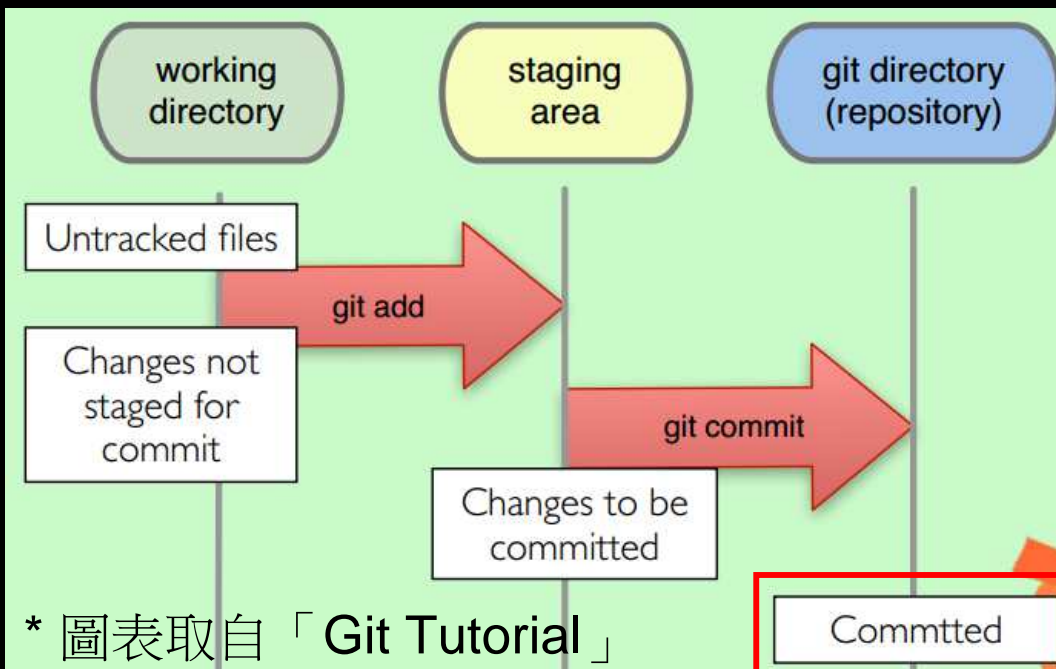
```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   function1.txt
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git commit -a -m "add function1"
[master (root-commit) d175788] add function1
1 file changed, 1 insertion(+)
create mode 100644 function1.txt
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
nothing to commit (working directory clean)
```

Git指令教學(status)

查看檔案狀態：**git status**



```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       function1.txt
nothing added to commit but untracked files present
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git add .
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   function1.txt
```

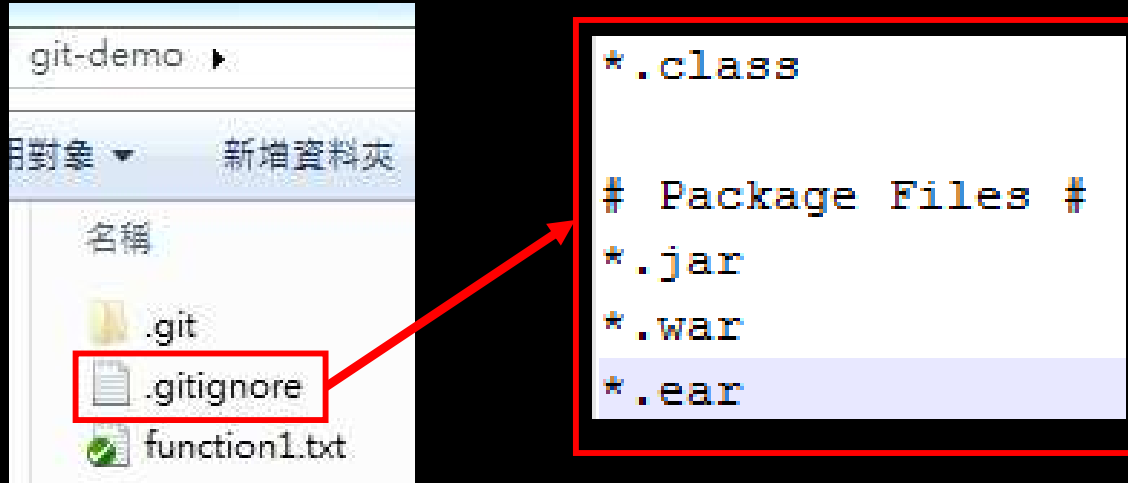
```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git commit -a -m "add function1"
[master (root-commit) d175788] add function1
1 file changed, 1 insertion(+)
create mode 100644 function1.txt
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git status
# On branch master
nothing to commit (working directory clean)
```

Git指令教學(gitignore)

- .gitignore

– 告訴Git哪些檔案類型不用進版控



– .gitignore大集合

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

Git指令教學(diff/show)

- commit之前看檔案內容差異
 - `git diff`
 - `git diff [file_name]`
- commit之後看檔案內容差異
 - `git show`
 - `git show HEAD^`
 - `git show [file_name]`

Git指令教學(diff/show)

- commit之前看檔案內容差異
 - `git diff`
 - `git diff [file_name]`
- commit之後看檔案內容差異
 - `git show`
 - `git show HEAD^`
 - `git show [file_name]`

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ ls -l
total 1
-rw-r--r--  1 ccloudtu  Administ      10 Aug 21 13:52 function1.txt
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git diff
diff --git a/function1.txt b/function1.txt
index 4119a72..dccd23f 100644
--- a/function1.txt
+++ b/function1.txt
@@ -1,1 @@
-<U+FEFF>change0
\ No newline at end of file
+<U+FEFF>change1
\ No newline at end of file
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git commit -a -m "change function1"
[master 246924f] change function1
1 file changed, 1 insertion(+), 1 deletion(-)
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git show
commit 246924f18f45bef4cee3c97de023edeb9468cd05
Author: ccloudtu <ccloud.tu@gmail.com>
Date: Tue Aug 21 13:52:51 2012 +0800
```

change function1

```
diff --git a/function1.txt b/function1.txt
index 4119a72..dccd23f 100644
--- a/function1.txt
+++ b/function1.txt
@@ -1,1 @@
-<U+FEFF>change0
\ No newline at end of file
+<U+FEFF>change1
\ No newline at end of file
```

Git指令教學(diff/show)

- commit之前看檔案內容差異
 - git diff
 - git diff [file_name]
- commit之後看檔案內容差異
 - git show
 - git show HEAD^
 - git show [file_name]

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ ls -l
total 1
-rw-r--r--  1 ccloudtu  Administ      10 Aug 21 13:52 function1.txt
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git diff
diff --git a/function1.txt b/function1.txt
index 4119a72..dccd23f 100644
--- a/function1.txt
+++ b/function1.txt
@@ -1,1 @@
-<U+FEFF>change0
\ No newline at end of file
+<U+FEFF>change1
\ No newline at end of file
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git commit -a -m "change function1"
[master 246924f] change function1
1 file changed, 1 insertion(+), 1 deletion(-)
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git show
commit 246924f18f45bef4cee3c97de023edeb9468cd05
Author: ccloudtu <ccloud.tu@gmail.com>
Date: Tue Aug 21 13:52:51 2012 +0800

    change function1

diff --git a/function1.txt b/function1.txt
index 4119a72..dccd23f 100644
--- a/function1.txt
+++ b/function1.txt
@@ -1,1 @@
-<U+FEFF>change0
\ No newline at end of file
+<U+FEFF>change1
\ No newline at end of file
```

Git指令教學(log)

查看commit的歷史記錄

- `git log`
- `git log [file_name]`
- `git log --graph --decorate --all`

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git log --graph --decorate --all
* commit 246924f18f45bef4cee3c97de023edeb9468cd05 (HEAD, master)
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Tue Aug 21 13:52:51 2012 +0800

       change function1

* commit dabd7200767e21b10d183a18e5e3c563b546ca7a
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Tue Aug 21 13:43:00 2012 +0800

       add .gitignore

* commit d1757885c68d18156c69cae36894fc81779d2b66
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Tue Aug 21 11:38:56 2012 +0800

       add function1
```


Git指令教學(reset)

- 檔案內容改爛了!怎麼辦?
 - commit之前改爛了，恢復到修改前的內容
 - `git checkout [file_name]`
 - `git reset --hard`
 - commit之後後悔了，恢復到前一版的commit
 - `git reset --hard HEAD^`

Git指令教學(reset)

- 檔案內容改爛了!怎麼辦?
 - 未commit前改爛了，恢復到修改前的內容
 - git checkout [file_name]
 - git reset --hard
 - commit之後後悔了，恢復到前一版的commit
 - git reset --hard HEAD^

執行後，前一版的commit(246924f18f45bef4cee3c97de023edeb9468cd05)砍掉。這招起手無回，砍掉後救不回來，使用前請三思。

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git log --graph --decorate --all
* commit 246924f18f45bef4cee3c97de023edeb9468cd05 (HEAD, master)
  Author: ccloudtu <ccloud.tu@gmail.com>
  Date:   Tue Aug 21 13:52:51 2012 +0800

      change function1

* commit dabd7200767e21b10d183a18e5e3c563b546ca7a
  Author: ccloudtu <ccloud.tu@gmail.com>
  Date:   Tue Aug 21 13:43:00 2012 +0800

      add .gitignore

* commit d1757885c68d18156c69cae36894fc81779d2b66
  Author: ccloudtu <ccloud.tu@gmail.com>
  Date:   Tue Aug 21 11:38:56 2012 +0800

      add function1

ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git reset --hard HEAD^
HEAD is now at dabd720 add .gitignore

ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git log --graph --decorate --all
* commit dabd7200767e21b10d183a18e5e3c563b546ca7a (HEAD, master)
  Author: ccloudtu <ccloud.tu@gmail.com>
  Date:   Tue Aug 21 13:43:00 2012 +0800

      add .gitignore

* commit d1757885c68d18156c69cae36894fc81779d2b66
  Author: ccloudtu <ccloud.tu@gmail.com>
  Date:   Tue Aug 21 11:38:56 2012 +0800

      add function1
```

Git指令教學(branch)

為何要使用branch(分支)?

- 不使用 branch 你簡直浪費 Git !
 - 在 Git 裡 branch 可以開很大、開不用錢
- 使用 branch 的時機點
 - 進行重構 (refactor)
 - 開發新功能
 - 修復臭蟲 (bugs)
- 好處：不會影響上線的產品 (production)

Git指令教學(branch)

使用branch(分支)



* 取自「Git in a nutshell」

Git指令教學(branch)

使用branch(分支)

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (new_feature1)
$ git log --graph --decorate --all
* commit fd167176ca5948a9b589c443c0e6f79ab4e3fbb7 (master)
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 15:05:43 2012 +0800
    change function1

* commit a8af617aa3653d1bdf1cbab863f05b99dbff1f1f (HEAD, new_feature1)
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 15:03:41 2012 +0800
    add function2

* commit dabd7200767e21b10d183a18e5e3c563b546ca7a
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 13:43:00 2012 +0800
    add .gitignore

* commit d1757885c68d18156c69cae36894fc81779d2b66
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 11:38:56 2012 +0800
    add function1
```

「master」 branch

「new_feature1」 branch

「HEAD」代表目前所在位置。所以目前在
「new_feature1」 branch的最後一個commit

Git指令教學(branch)

- 列出所有branch

```
git branch -a
```

- 產生新branch

```
git branch [branch_name]
```

- 切換branch

```
git checkout [branch_name]
```

- 刪除branch

```
git branch -d [branch_name]
```

Git指令教學(branch)

列出所有branch : `git branch -a`

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git branch -a
* master
  new_feature1
```

Git指令教學(branch)

產生新branch：`git branch [branch_name]`

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git branch new_feature2
```

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git branch -a
* master
  new_feature1
  new_feature2
```


Git指令教學(branch)

切換branch：`git checkout [branch_name]`

```
c1oudtu@CLOUDTU-NB-T410 /z/git-demo (master)  
$ git checkout new_feature2  
Switched to branch 'new_feature2'
```

```
c1oudtu@CLOUDTU-NB-T410 /z/git-demo (new_feature2)  
$
```

從「master」branch切到「new_feature2」branch

Git指令教學(branch)

刪除branch：`git branch -d [branch_name]`

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git branch -a
* master
  new_feature1
  new_feature2
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git branch -d new_feature2
Deleted branch new_feature2 (was fd16717).
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git branch -a
* master
  new_feature1
```

Git指令教學(tag)

- 幫特定commit進行”貼標籤”動作，一般用來把將要上線版本資料進行tag動作

- 列出所有tag

`git tag`

- 產生新tag

`git tag [tag_name]`

- 刪除tag

`git tag -d [tag_name]`

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git log --graph --decorate --all
* commit fd167176ca5948a9b589c443c0e6f79ab4e3fbb7 (HEAD, master)
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 15:05:43 2012 +0800

    change function1

* commit a8af617aa3653d1bdf1cbab863f05b99dbff1f1f (new_feature1)
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 15:03:41 2012 +0800

    add function2

* commit dabd7200767e21b10d183a18e5e3c563b546ca7a (tag: prd_release_1)
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 13:43:00 2012 +0800

    add .gitignore

* commit d1757885c68d18156c69cae36894fc81779d2b66
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 11:38:56 2012 +0800

    add function1
```

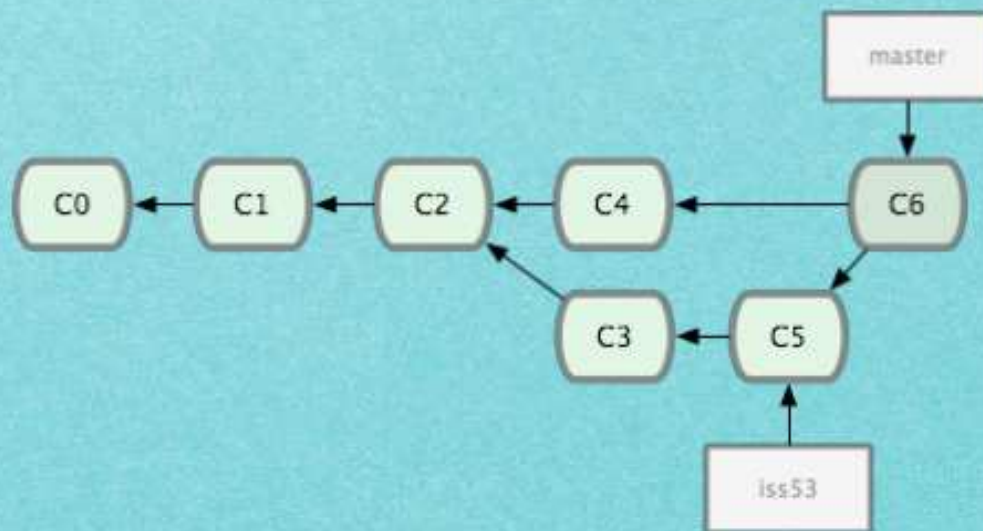
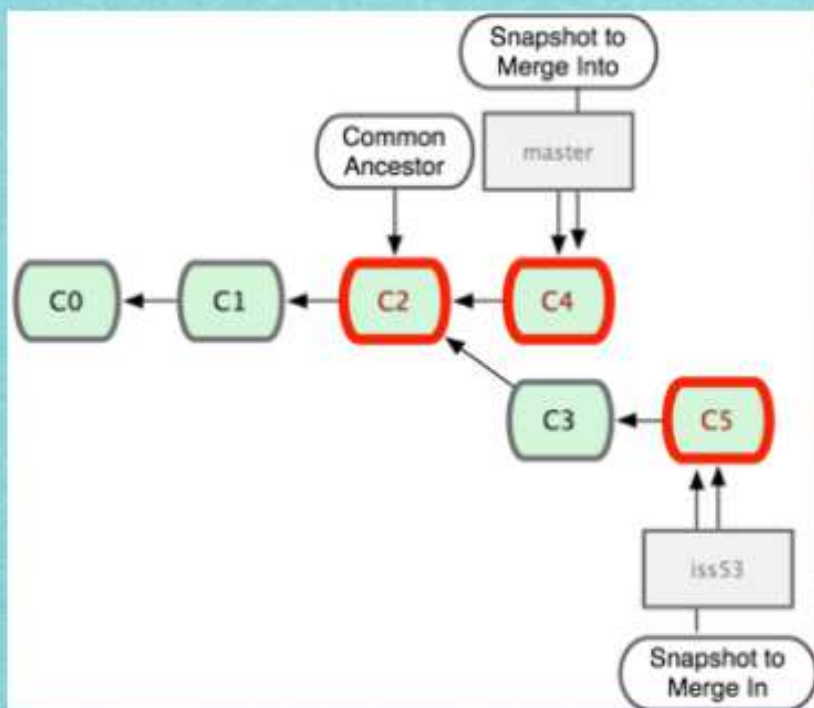
Git指令教學(merge)

- 把其它branch接在current branch的尾巴
- 範例

```
$ git checkout master  
$ git merge iss53  
Merge made by recursive.
```

```
README | 1 +
```

```
1 files changed, 1 insertions(+), 0 deletions(-)
```



Git指令教學(merge)

- merge時衝到了(conflict)，怎麼辦？

▶ 別慌張！

- ▶ git status 一下看是哪個檔案出問題
- ▶ 到出問題的檔案找問題，手動解決
- ▶ 再檢查 git status 後，用 git commit 手動 Merge
(會自動生成 Merge 的 Commit Message)

```
<<<<<<< HEAD:index.html
<div id="footer">contact : email.support@github.com</div>
=====
<div id="footer">
  please contact us at support@github.com
</div>
>>>>>>> iss53:index.html
```

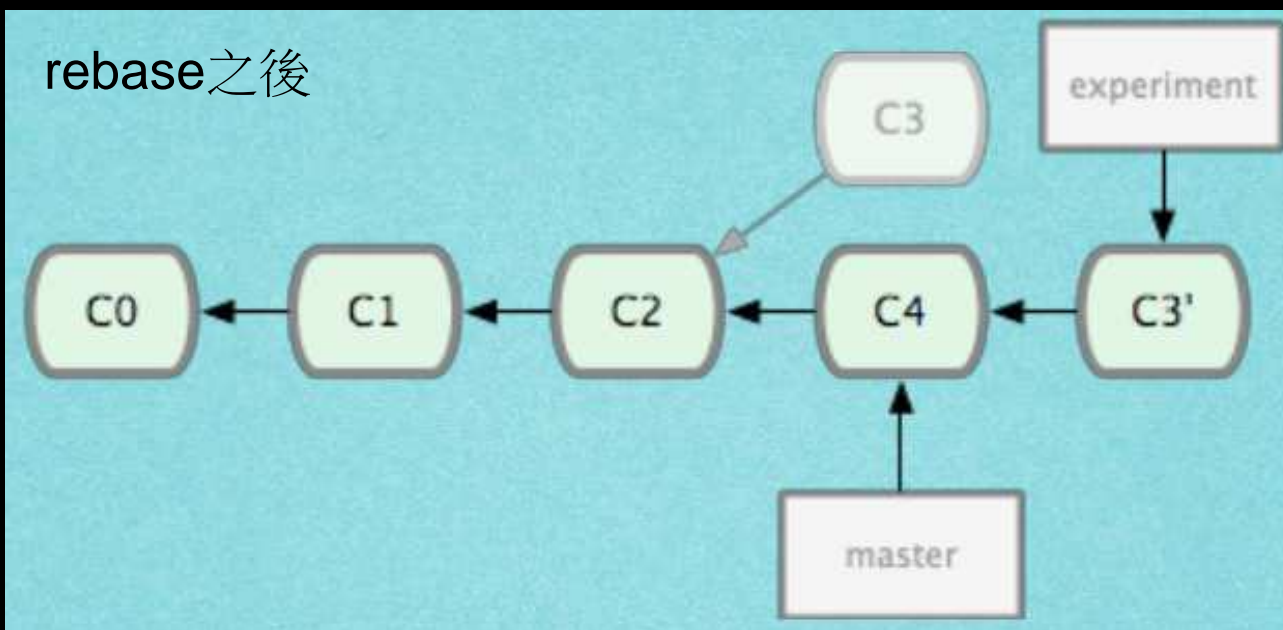
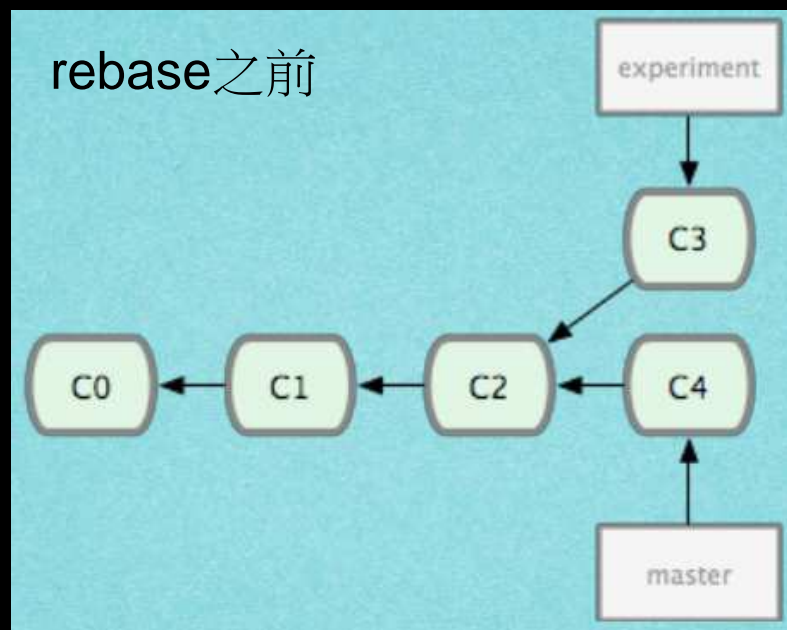
Git指令教學(rebase)

- current branch的頭改接在其它branch之後

- 範例

git checkout experiment

git rebase master



Git指令教學(merge/rebase)

▶ Merge

- ▶ 把岔開來的分支在往後「合起來」
- ▶ 通常的建議方式

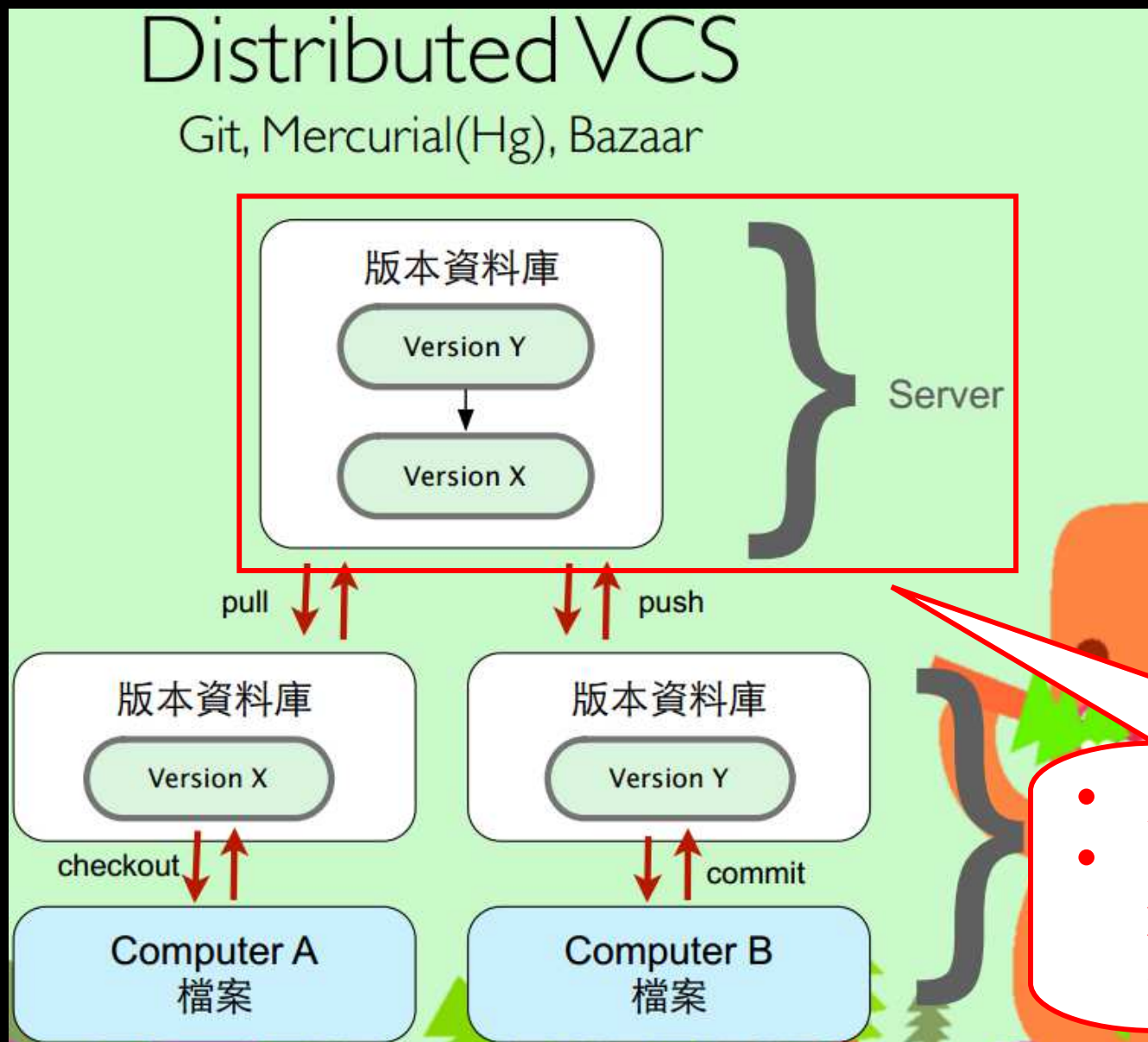
▶ Rebase

- ▶ 把岔開來的分支「裝回去」主線
- ▶ 「還沒Push出去的東西」才可以Rebase！

Git指令教學(merge/rebase)

- 還是搞不清楚merge與rebase差異？
 - 多練習，多玩几次就會懂。
 - 一般狀況下大多用到merge，不會rebase其實影響不大。
 - 除非你(妳)搞的清狀況，Production repository 別亂 rebase。因此搞爛Production repository 的話，把搞爛的人拖出來毒打一頓。

Git指令教學(remote)

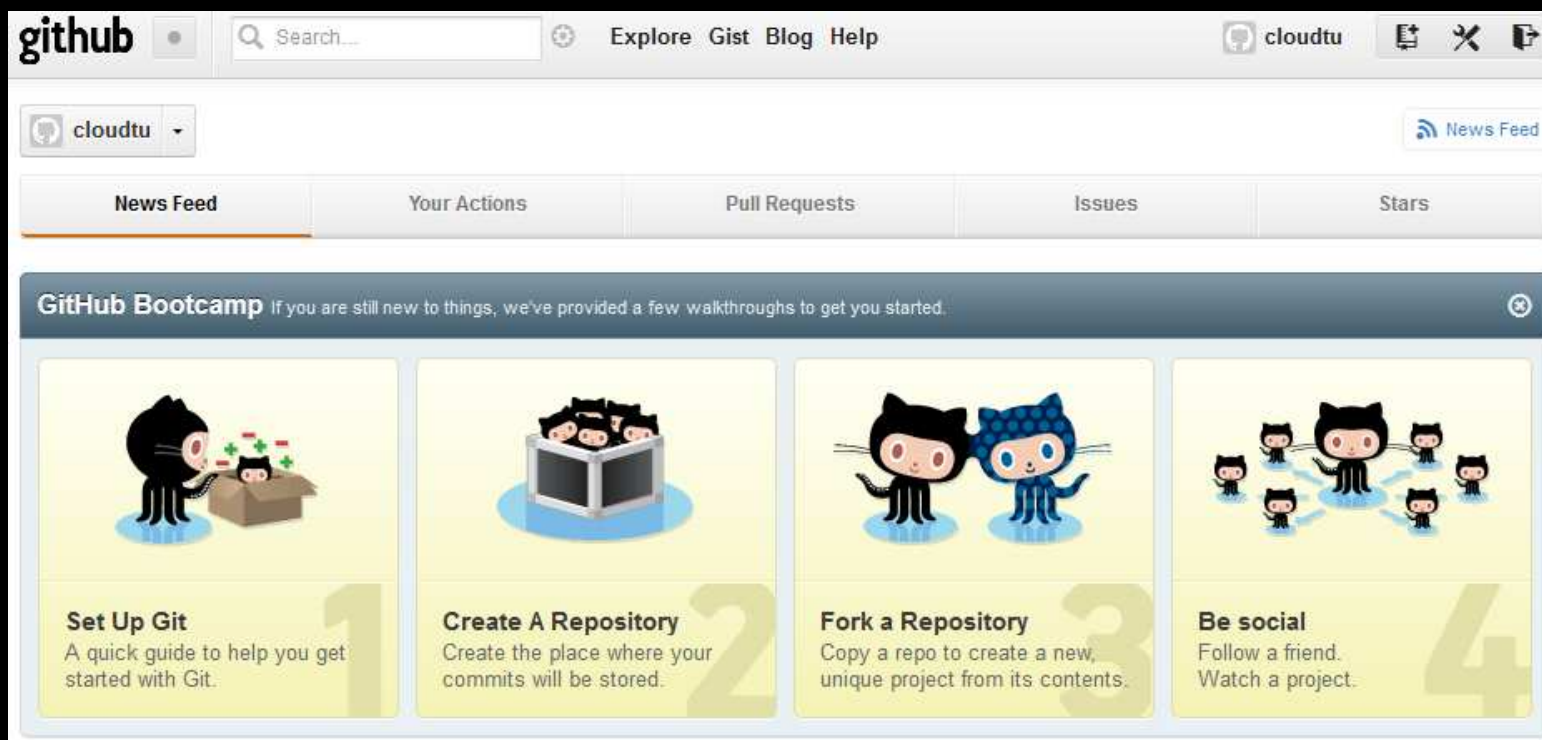


- 又回到了這張圖... :)
- 前述指令都圍繞在**local**端，現在要講**server**端指令

Git指令教學(remote)

離題一下，看看github這玩意兒

- <https://github.com>
- 簡單來說就是商業公司提供的git host服務；大廟一間，超多專案放上面
- 談錢傷感情，不過github上只有public project host免費，private project host要錢，好在價格便宜



Git指令教學(remote)

離題一下，看看github這玩意兒

- 建立repository(step1)

The screenshot shows the GitHub homepage. At the top, there's a navigation bar with the GitHub logo, a search bar, and links for Explore, Gist, Blog, and Help. Below this is a secondary navigation bar with tabs for News Feed, Your Actions, Pull Requests, Issues, and Stars. The main content area features a 'GitHub Bootcamp' section with four cards: 'Set Up Git', 'Create A Repository', 'Fork a Repository', and 'Be social'. Below the bootcamp section, there's a 'Welcome to GitHub! What's next?' section with links to 'Create a Repository', 'Tell us about yourself', 'Browse Interesting Repos', and 'Follow @github on Twitter'. To the right of the welcome section is a 'Notifications & Stars' section. At the bottom right, there's a 'Your Repositories (0)' section with a 'New repository' button highlighted by a red box. Below this button, it says 'You don't have any repositories yet! Create your first repository or learn more about Git and GitHub'.

github Search... Explore Gist Blog Help cloudtu

cloudtu News Feed

News Feed Your Actions Pull Requests Issues Stars

GitHub Bootcamp If you are still new to things, we've provided a few walkthroughs to get you started.

Set Up Git
A quick guide to help you get started with Git.

Create A Repository
Create the place where your commits will be stored.

Fork a Repository
Copy a repo to create a new, unique project from its contents.

Be social
Follow a friend. Watch a project.

Welcome to GitHub! What's next? (2 months ago)
Create a Repository
Tell us about yourself
Browse Interesting Repos
Follow @github on Twitter

Notifications & Stars
We're changing what it means to watch repositories. Come learn about our new Notifications & Stars.

Your Repositories (0) **New repository**

You don't have any repositories yet!
Create your first repository or learn more about Git and GitHub

Git指令教學(remote)

離題一下，看看github這玩意兒

- 建立repository(step2)

github [Explore](#) [Gist](#) [Blog](#) [Help](#) [cloudtu](#)

Owner **Repository name**

Great repository names are short and memorable. Need inspiration? How about [furry-dangerzone](#).

Description (optional)

☒ **Public** Anyone can see this repository. You choose who can commit.

☐ **Private** You choose who can see and commit to this repository.

☒ **Initialize this repository with a README**
This will allow you to `git clone` the repository immediately.

Add .gitignore: **None**

Create repository

Git指令教學(remote)

離題一下，看看github這玩意兒

- 建立repository(step3)

The screenshot shows the GitHub interface for a repository named 'git-demo' by user 'cloudtu'. The top navigation bar includes the GitHub logo, a search bar, and links for 'Explore', 'Gist', 'Blog', and 'Help'. The repository name 'cloudtu / git-demo' is displayed, along with 'Unwatch' and 'Star' buttons. Below the repository name, there are tabs for 'Code', 'Network', 'Pull Requests', 'Issues', 'Wiki', 'Graphs', and 'Admin'. A 'Quick setup' section provides instructions for cloning the repository, with a green button for 'Setup in Windows' and a text input field for the SSH URL: 'https://github.com/cloudtu/git-demo.git'. A recommendation states that every repository should have a README, LICENSE, and .gitignore. Two sections follow: 'Create a new repository on the command line' and 'Push an existing repository from the command line', each with a code block containing the necessary Git commands.

github Search... Explore Gist Blog Help cloudtu

cloudtu / git-demo Unwatch Star 0

Code Network Pull Requests 0 Issues 0 Wiki Graphs Admin

Quick setup — if you've done this kind of thing before

Setup in Windows or HTTP SSH

We recommend that every repository has a **README**, **LICENSE**, and **.gitignore**

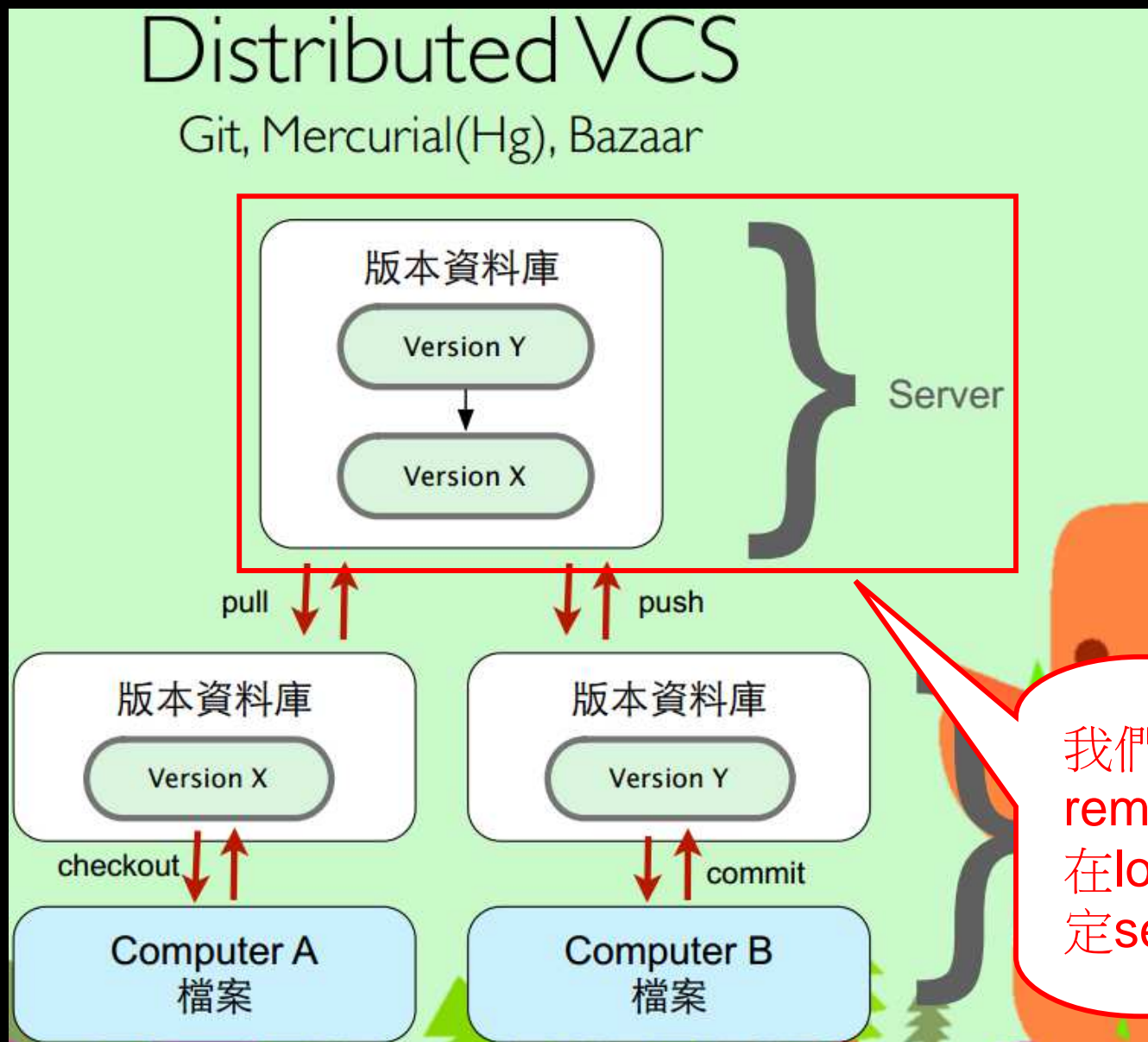
Create a new repository on the command line

```
touch README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/cloudtu/git-demo.git
git push -u origin master
```

Push an existing repository from the command line

```
git remote add origin https://github.com/cloudtu/git-demo.git
git push -u origin master
```


Git指令教學(remote)



我們在server端(github)建立了remote repository，接著可以在local端使用remote指令設定server端名稱與位址

Git指令教學(remote)

- remote指令用來設定local端要連線至server端時的連線位址
- 列出server端位址
`git remote -v`
- 設定server端位址
`git remote add origin https://github.com/cloudtu/git-demo.git`
- 刪除server端位址
`git remote rm origin`

Git指令教學(remote)

列出server端位址：`git remote -v`

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git remote -v
origin  https://github.com/cloudtu/git-demo.git (fetch)
origin  https://github.com/cloudtu/git-demo.git (push)
```


Git指令教學(remote)

設定server端位址

```
git remote add origin https://github.com/cloudtu/git-demo.git
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)  
$ git remote -v
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)  
$ git remote add origin https://github.com/cloudtu/git-demo.git
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)  
$ git remote -v  
origin https://github.com/cloudtu/git-demo.git (fetch)  
origin https://github.com/cloudtu/git-demo.git (push)
```

Git指令教學(remote)

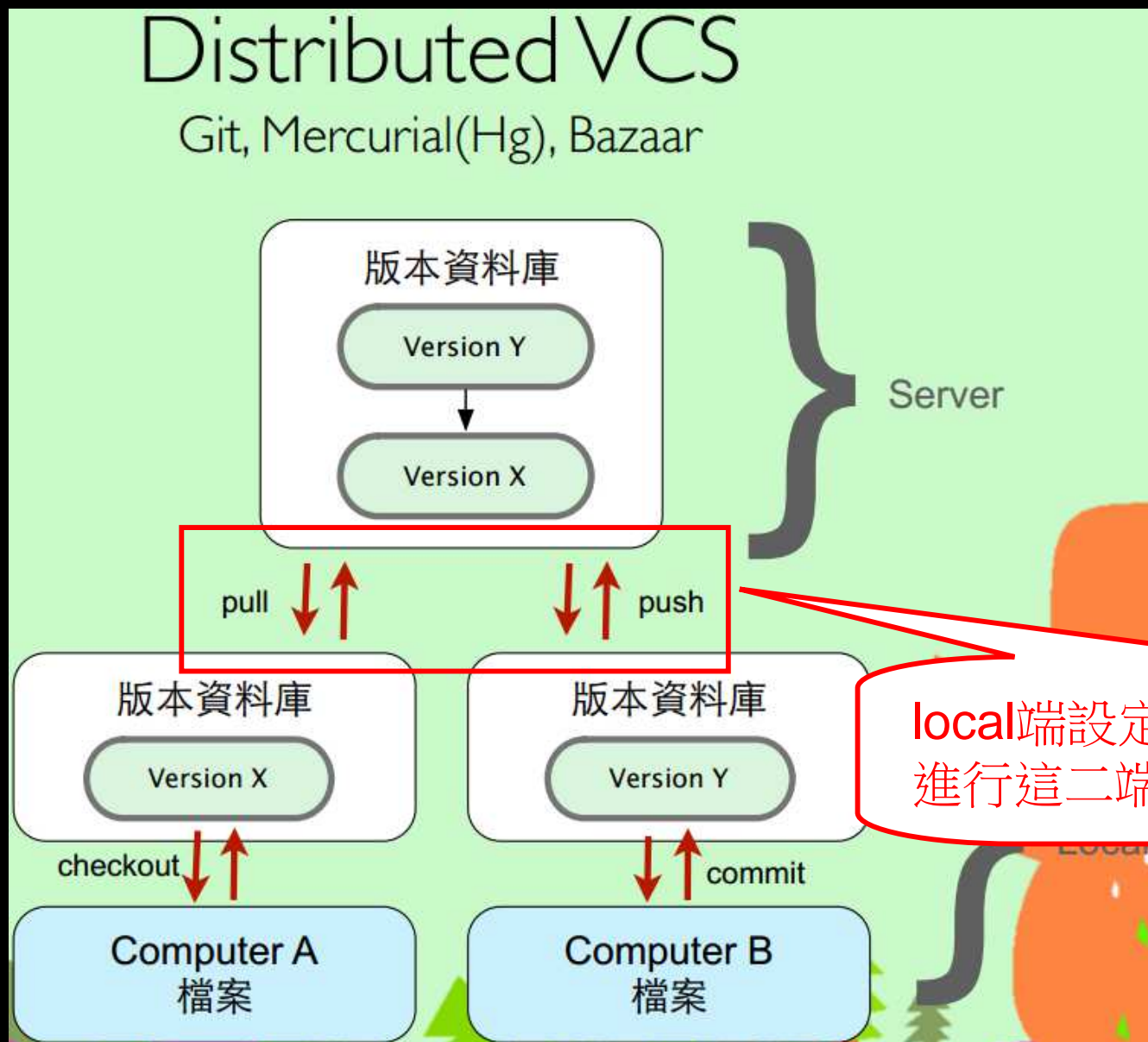
刪除server端位址：`git remote rm origin`

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git remote -v
origin  https://github.com/cloudtu/git-demo.git (fetch)
origin  https://github.com/cloudtu/git-demo.git (push)
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git remote rm origin
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git remote -v
```

Git指令教學(push/pull)



Git指令教學(push/pull)

- 把local端資料推送到server端
 - 用push指令
 - 推送所有branch到server : `git push --all origin`
 - 推送所有tag到server : `git push --tags origin`
 - 推送特定branch到server : `git push origin master`
- 把server端資料拉回並合併到local端
 - 用pull指令
 - 拉回所有branch並合併到local : `git pull origin`
 - 拉回特定branch並合併到local : `git pull origin master`
 - 拉回並合併後把local repository搞的亂七八糟，想要後悔!
 - 那就退回這次的動作吧 : `git reset --hard HEAD^`

Git指令教學(push/pull)

- push完成後的log

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git push --all origin
Username for 'https://github.com': cloudtu
Password for 'https://cloudtu@github.com':
Counting objects: 11, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (11/11), 880 bytes, done.
Total 11 (delta 2), reused 0 (delta 0)
To https://github.com/cloudtu/git-demo.git
 * [new branch]      master -> master
 * [new branch]      new_feature1 -> new_feature1

cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git push --tags origin
Username for 'https://github.com': cloudtu
Password for 'https://cloudtu@github.com':
Total 0 (delta 0), reused 0 (delta 0)
To https://github.com/cloudtu/git-demo.git
 * [new tag]         prd_release_1 -> prd_release_1

cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git log --graph --decorate --all
* commit fd167176ca5948a9b589c443c0e6f79ab4e3fbb7 (HEAD, origin/master, master)
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Tue Aug 21 15:05:43 2012 +0800

       change function1

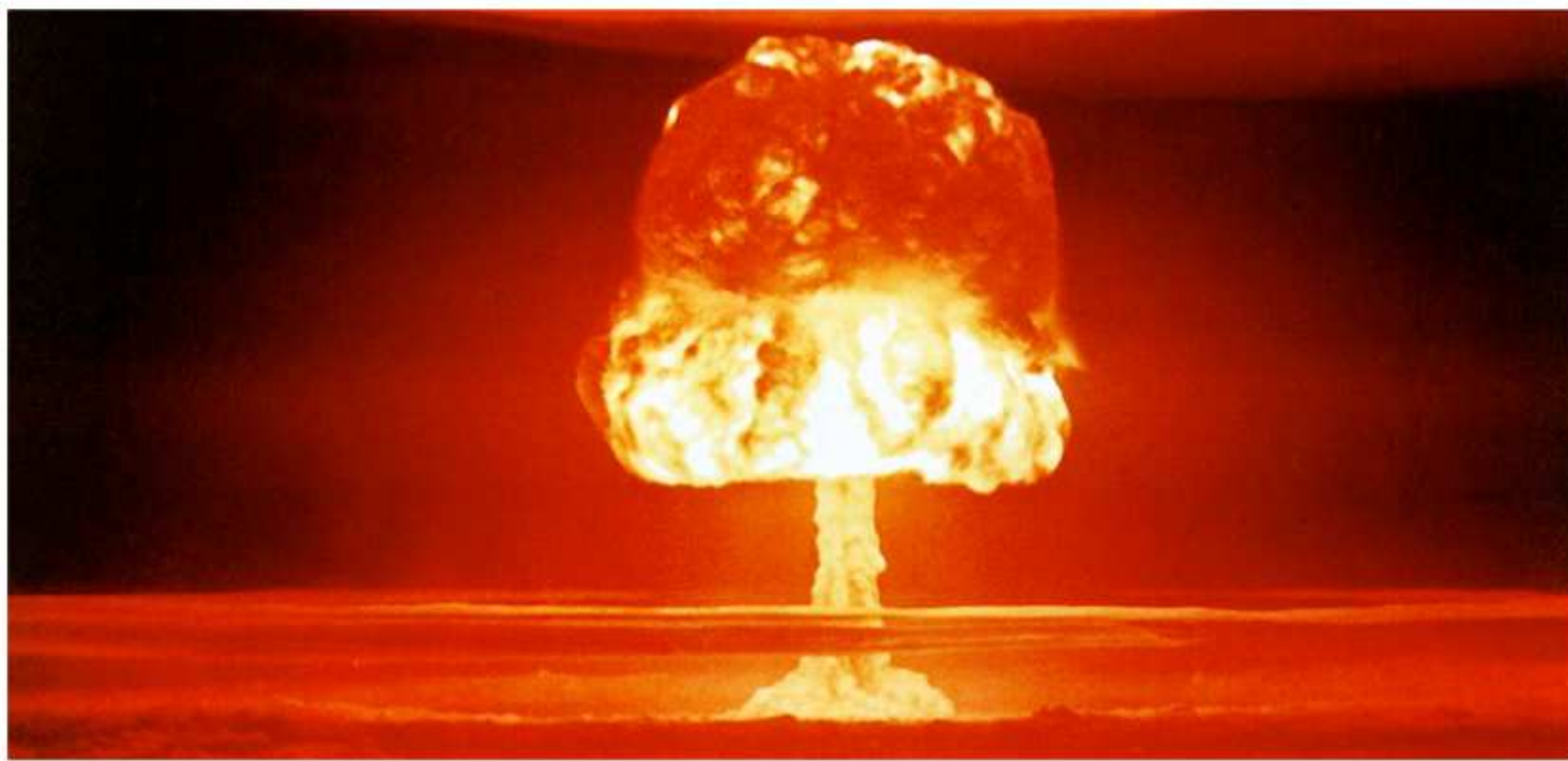
 * commit a8af617aa3653d1bdf1cbab863f05b99dbff1f1f (origin/new_feature1, new_fe
   / Author: cloudtu <cloud.tu@gmail.com>
   Date:   Tue Aug 21 15:03:41 2012 +0800

       add function2

 * commit dabd7200767e21b10d183a18e5e3c563b546ca7a (tag: prd_release_1)
   | Author: cloudtu <cloud.tu@gmail.com>
   | Date:   Tue Aug 21 13:43:00 2012 +0800
   |
   | add .gitignore
   |
 * commit d1757885c68d18156c69cae36894fc81779d2b66
   | Author: cloudtu <cloud.tu@gmail.com>
   | Date:   Tue Aug 21 11:38:56 2012 +0800
   |
   | add function1
```

server端remote branch也會記錄在log裡面

Git指令教學(push/pull)



「千萬不要」對已經*Push*的東西
作*Rebase*！

除非你想成為害群之馬。

Pic: http://commons.wikimedia.org/wiki/File:Castle_Romeo.jpg, Public Domain, United States Department of Energy

* 取自「寫給大家的Git教學」

Git指令教學(push/pull)

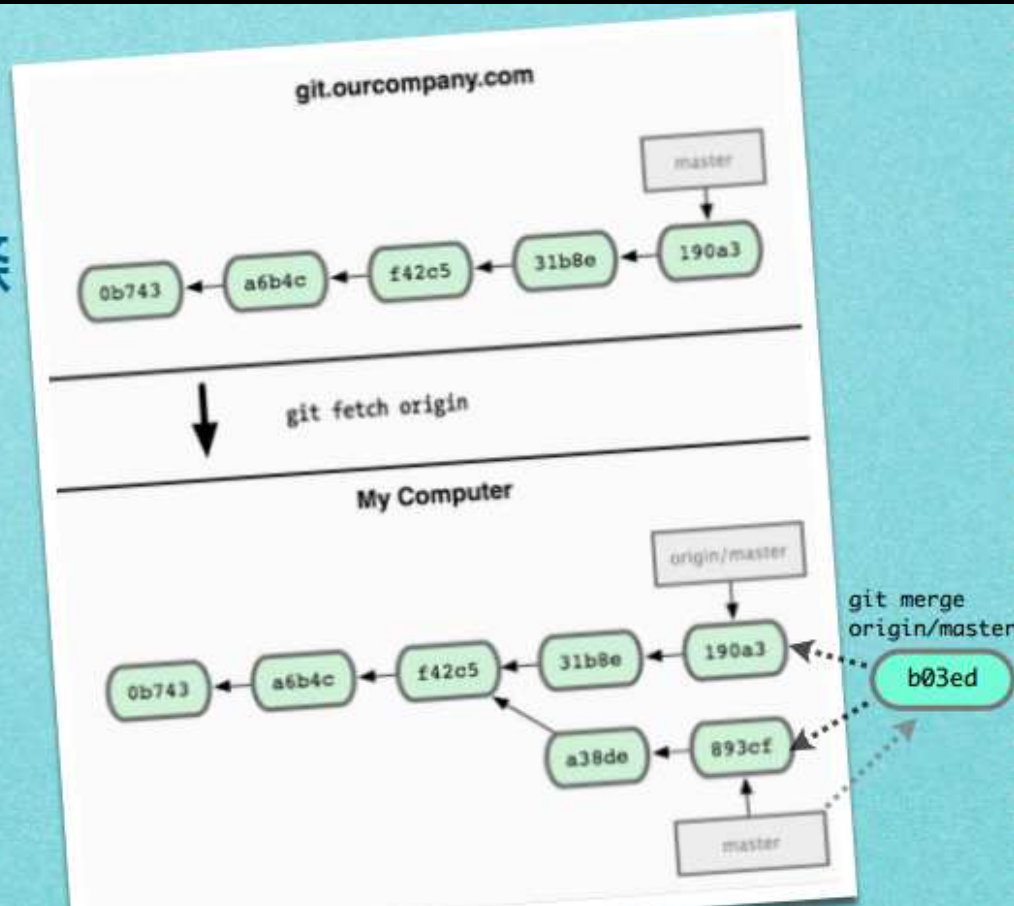
- pull = fetch + merge

▶ 把Origin作Fetch之後，兩邊同時有更動，因此會變成兩條分支

▶ 通常的作法是，用Merge合併回一條

▶ 所以會有git pull：

```
git pull origin =  
git fetch origin 之後  
git merge origin/master
```



* 取自「寫給大家的Git教學」

Git指令教學(push/pull)

- pull - special case demo(step1)

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git log --graph --decorate --all
* commit fd167176ca5948a9b589c443c0e6f79ab4e3fbb7 (HEAD, origin/master, master)
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 15:05:43 2012 +0800
    change function1

* commit a8af617aa3653d1bdf1cbab863f05b99dbff1f1f (origin/new_feature1, new_fe
  / Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 15:03:41 2012 +0800
    add function2

* commit dabd7200767e21b10d183a18e5e3c563b546ca7a (tag: prd_release_1)
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 13:43:00 2012 +0800
    add .gitignore

* commit d1757885c68d18156c69cae36894fc81779d2b66
  Author: cloudtu <cloud.tu@gmail.com>
  Date: Tue Aug 21 11:38:56 2012 +0800
    add function1

cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git pull origin
remote: Counting objects: 6, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 1), reused 4 (delta 1)
Unpacking objects: 100% (4/4), done.
From https://github.com/cloudtu/git-demo
  fd16717..47d85e7 master -> origin/master
  a8af617..83bbd83 new_feature1 -> origin/new_feature1
You asked to pull from the remote 'origin', but did not specify
a branch. Because this is not the default configured remote
for your current branch, you must specify a branch on the command line.
```


Git指令教學(push/pull)

- pull - special case demo(step2)

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git log --graph --decorate --all
* commit 83bbd83addc435d722ba05a2390bf2c1d62a6973 (origin/new_feature1)
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Fri Aug 24 14:40:48 2012 +0800

       change function2

* commit a8af617aa3653d1bdf1cbab863f05b99dbff1f1f (new_feature1)
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Tue Aug 21 15:03:41 2012 +0800

       add function2

* commit 47d85e75883c904d29dfe699a319d7c11a61428a (origin/master)
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Fri Aug 24 13:50:55 2012 +0800

       add function2

* commit fd167176ca5948a9b589c443c0e6f79ab4e3fbb7 (HEAD, master)
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Tue Aug 21 15:05:43 2012 +0800

       change function1

* commit dabd7200767e21b10d183a18e5e3c563b546ca7a
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Tue Aug 21 13:43:00 2012 +0800

       add .gitignore

* commit d1757885c68d18156c69cae36894fc81779d2b66
   Author: cloudtu <cloud.tu@gmail.com>
   Date:   Tue Aug 21 11:38:56 2012 +0800

       add function1
```

每個branch只有進行fetch，沒有merge，要自己手動處理

Git指令教學(push/pull)

- pull - special case demo(step3)

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
```

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

```
Updating fd16717..47d85e7
```

```
Fast-forward
```

```
function2.txt | 1 +  
1 file changed, 1 insertion(+)  
create mode 100644 function2.txt
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
```

```
$ git checkout new_feature1
```

```
Switched to branch 'new_feature1'
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (new_feature1)
```

```
$ git merge origin/new_feature1
```

```
Updating a8af617..83bbd83
```

```
Fast-forward
```

```
function2.txt | 2 +-  
1 file changed, 1 insertion(+), 1 deletion(-)
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (new_feature1)
```

```
$ git log --graph --decorate --all
```

```
* commit 83bbd83addc435d722ba05a2390bf2c1d62a6973 (HEAD, origin/new_feature1, new_feature1)  
Author: cloudtu <cloud.tu@gmail.com>  
Date: Fri Aug 24 14:40:48 2012 +0800
```

```
change function2
```

```
* commit a8af617aa3653d1bdf1cbab863f05b99dbff1f1f
```

```
Author: cloudtu <cloud.tu@gmail.com>  
Date: Tue Aug 21 15:03:41 2012 +0800
```

```
add function2
```

```
* commit 47d85e75883c904d29dfe699a319d7c11a61428a (origin/master, master)
```

```
Author: cloudtu <cloud.tu@gmail.com>  
Date: Fri Aug 24 13:50:55 2012 +0800
```

```
add function2
```

```
* commit fd167176ca5948a9b589c443c0e6f79ab4e3fbb7
```

```
Author: cloudtu <cloud.tu@gmail.com>  
Date: Tue Aug 21 15:05:43 2012 +0800
```

```
change function1
```

```
* commit dabd7200767e21b10d183a18e5e3c563b546ca7a (tag: prd_release_1)
```

```
Author: cloudtu <cloud.tu@gmail.com>  
Date: Tue Aug 21 13:43:00 2012 +0800
```

```
add .gitignore
```

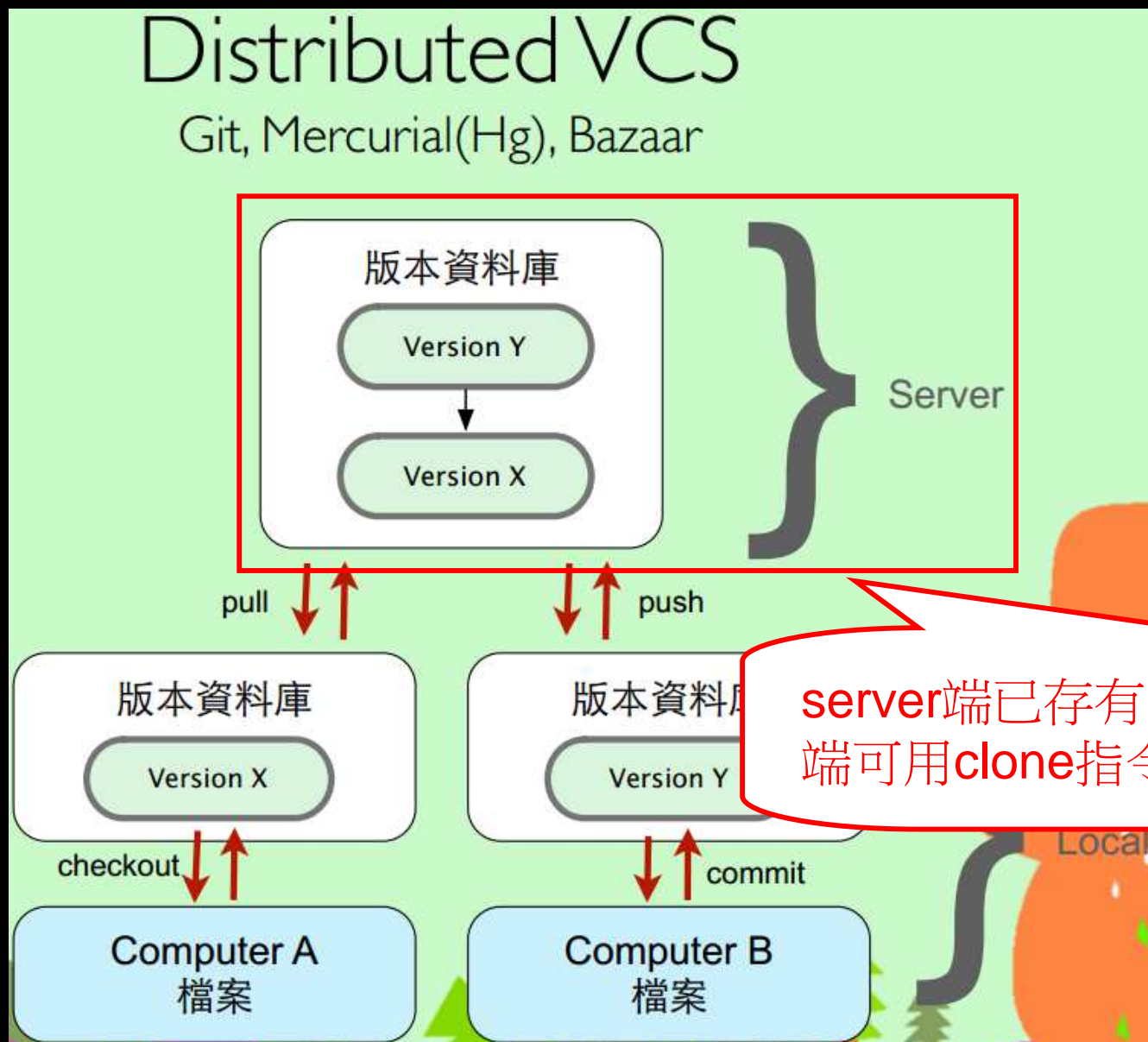
```
* commit d1757885c68d18156c69cae36894fc81779d2b66
```

```
Author: cloudtu <cloud.tu@gmail.com>  
Date: Tue Aug 21 11:38:56 2012 +0800
```

```
add function1
```

手動merge

Git指令教學(clone)



Git指令教學(clone)

- clone一份server端repository至local端

`git clone https://github.com/cloudtu/git-demo.git git-demo`

```
cloudtu@CLOUDTU-NB-T410 /z
$ git clone https://github.com/cloudtu/git-demo.git git-demo
Cloning into 'git-demo'...
remote: Counting objects: 11, done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 11 (delta 2), reused 11 (delta 2)
Unpacking objects: 100% (11/11), done.
```

```
cloudtu@CLOUDTU-NB-T410 /z
$ cd git-demo/
```

```
cloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$
```

Git指令教學(clone)

- clone出來的repository裡，local branch只會有master branch

Graph	Actions	Message
		Working dir changes
		master origin/HEAD origin/master change function1
		origin/new_feature1 add function2
		prd_release_1 add .gitignore
		add function1

- 需要其它的local branch要自己建

```
ccloudtu@CLOUDTU-NB-T410 /z/git-demo (master)
$ git checkout origin/new_feature1 -b new_feature1
Branch new_feature1 set up to track remote branch new_feature1 from origin.
Switched to a new branch 'new_feature1'
```

Graph	Actions	Message
		Working dir changes
		master origin/HEAD origin/master change function1
		new_feature1 origin/new_feature1 add function2
		prd_release_1 add .gitignore
		add function1

Agenda

- 版控系統發展史
- Git設計目標
- 安裝Git
- Git指令教學
- 常用的Git指令
- Git GUI工具教學(TortoiseGit)
- Git基本守則
- 常用協同開發流程
- 參考資料

常用的Git指令

- **Local**

- `git config`
- `git init`
- `git add`
- `git commit`
- `git status`
- `git log`
- `git tag`

- **Branch**

- `git checkout`
- `git branch`
- `git merge`
- `git rebase`

- **Remote**

- `git remove`
- `git fetch`
- `git pull`
- `git clone`
- `git push`

- **Patch**

- `git diff`
- `git apply`
- `git format-patch`
- `git am`

常用的Git指令

想要快速複習一下?



都放在那裡了，自己去拿吧!

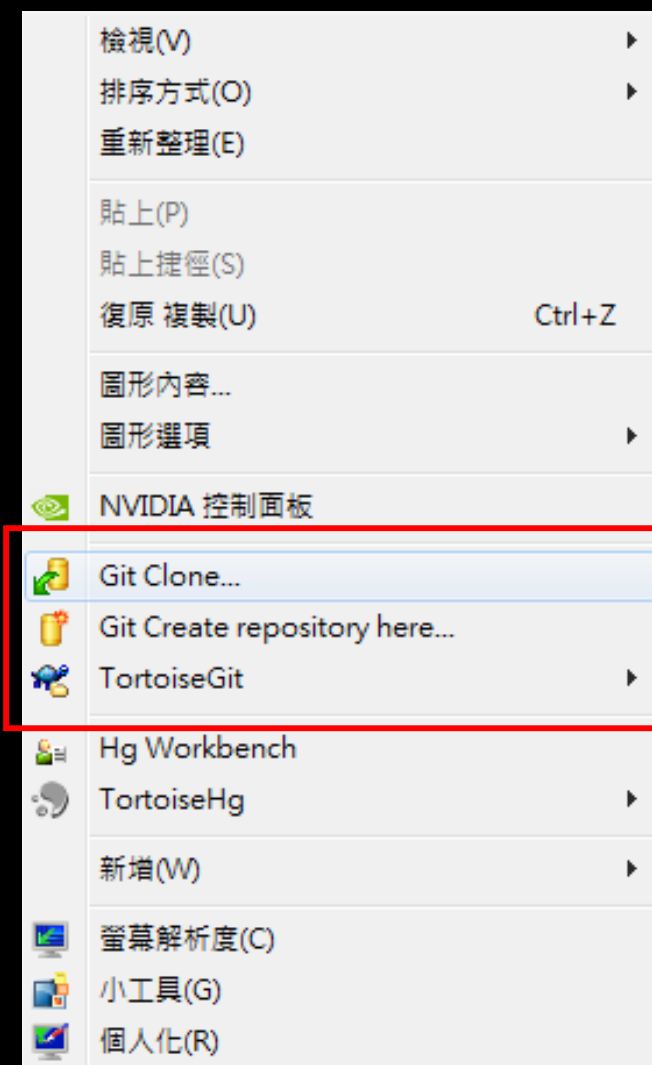
<http://cloudtu.blogspot.tw/2012/08/git-command-fast-memo.html>

Agenda

- 版控系統發展史
- Git設計目標
- 安裝Git
- Git指令教學
- 常用的Git指令
- **Git GUI工具教學(TortoiseGit)**
- Git基本守則
- 常用協同開發流程
- 參考資料

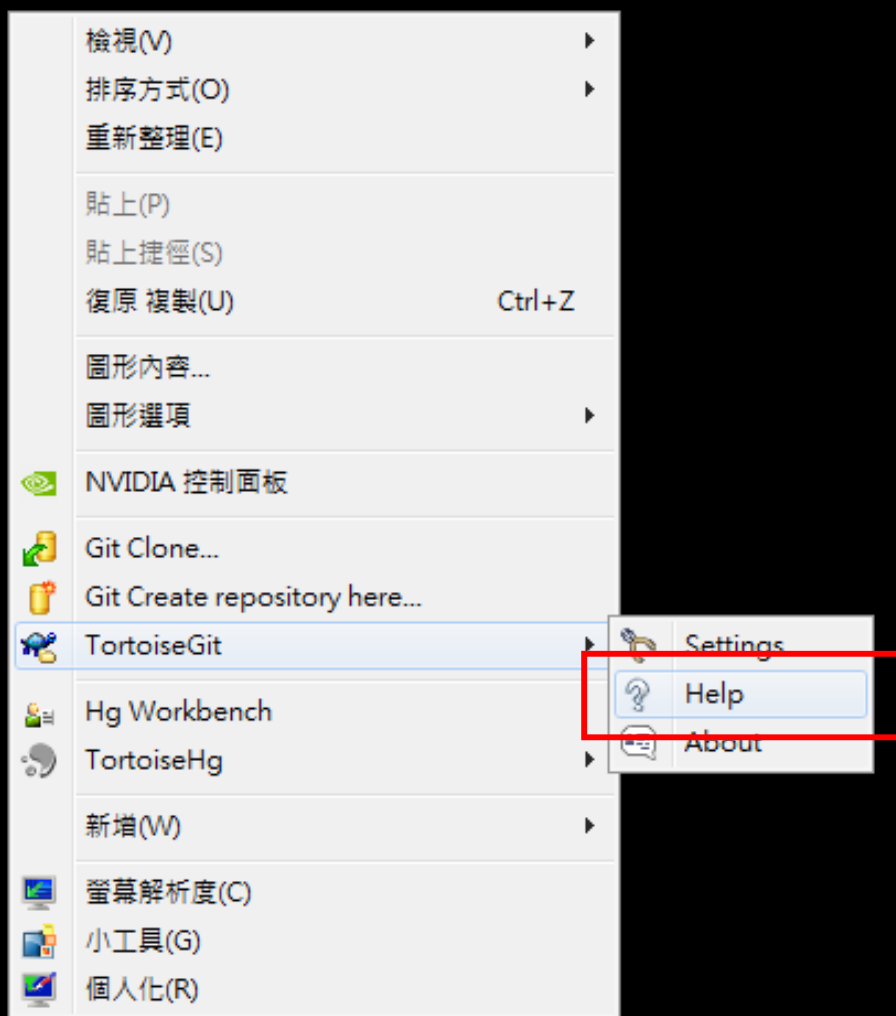
Git GUI工具教學(TortoiseGit)

TortoiseGit跟檔案總管直接整合，按下滑鼠右鈕就可以直接使用...：)



Git GUI工具教學(TortoiseGit)

查TortoiseGit怎麼用



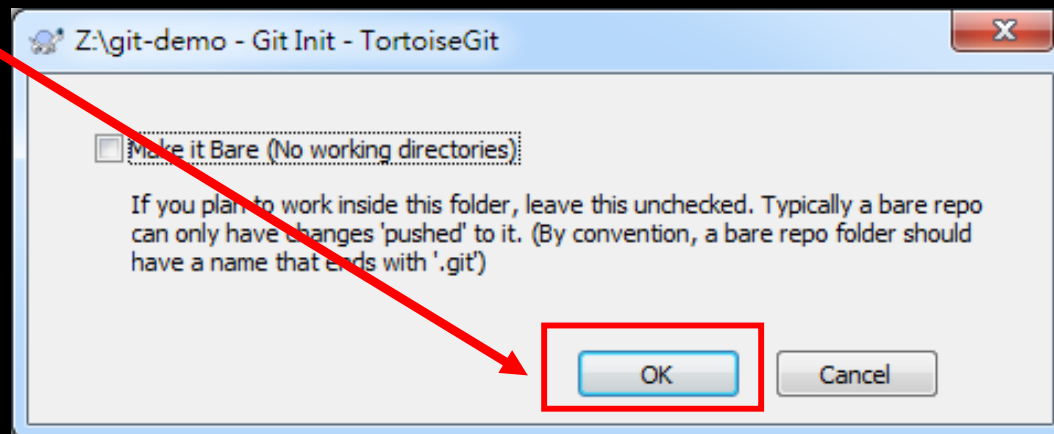
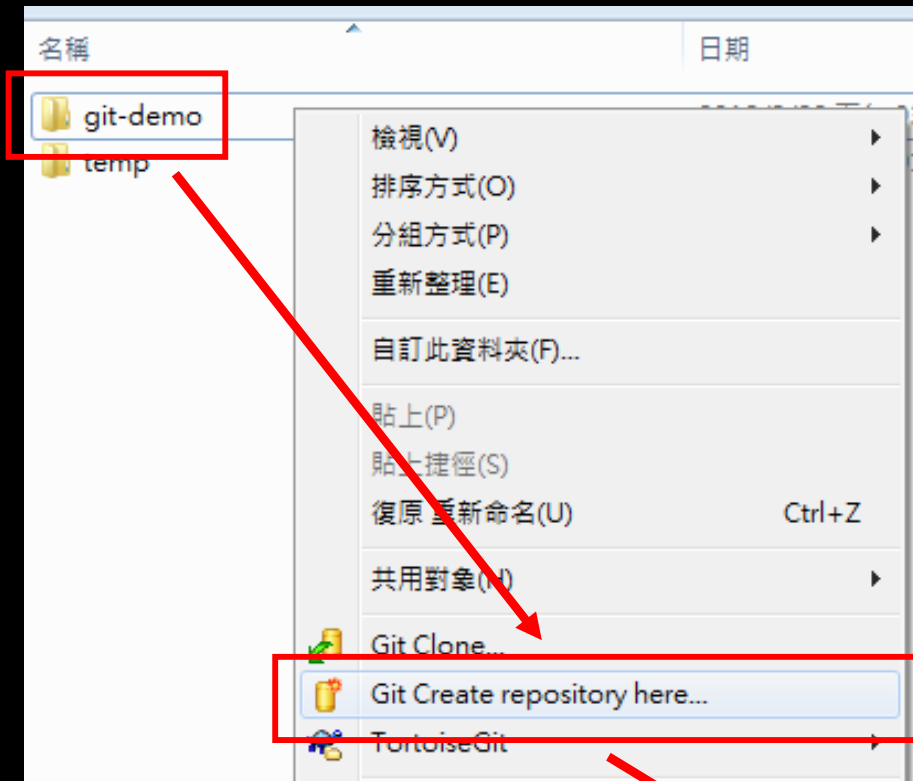
Git GUI工具教學(TortoiseGit)

- 設定Git系統參數
 - TortoiseGit → Settings



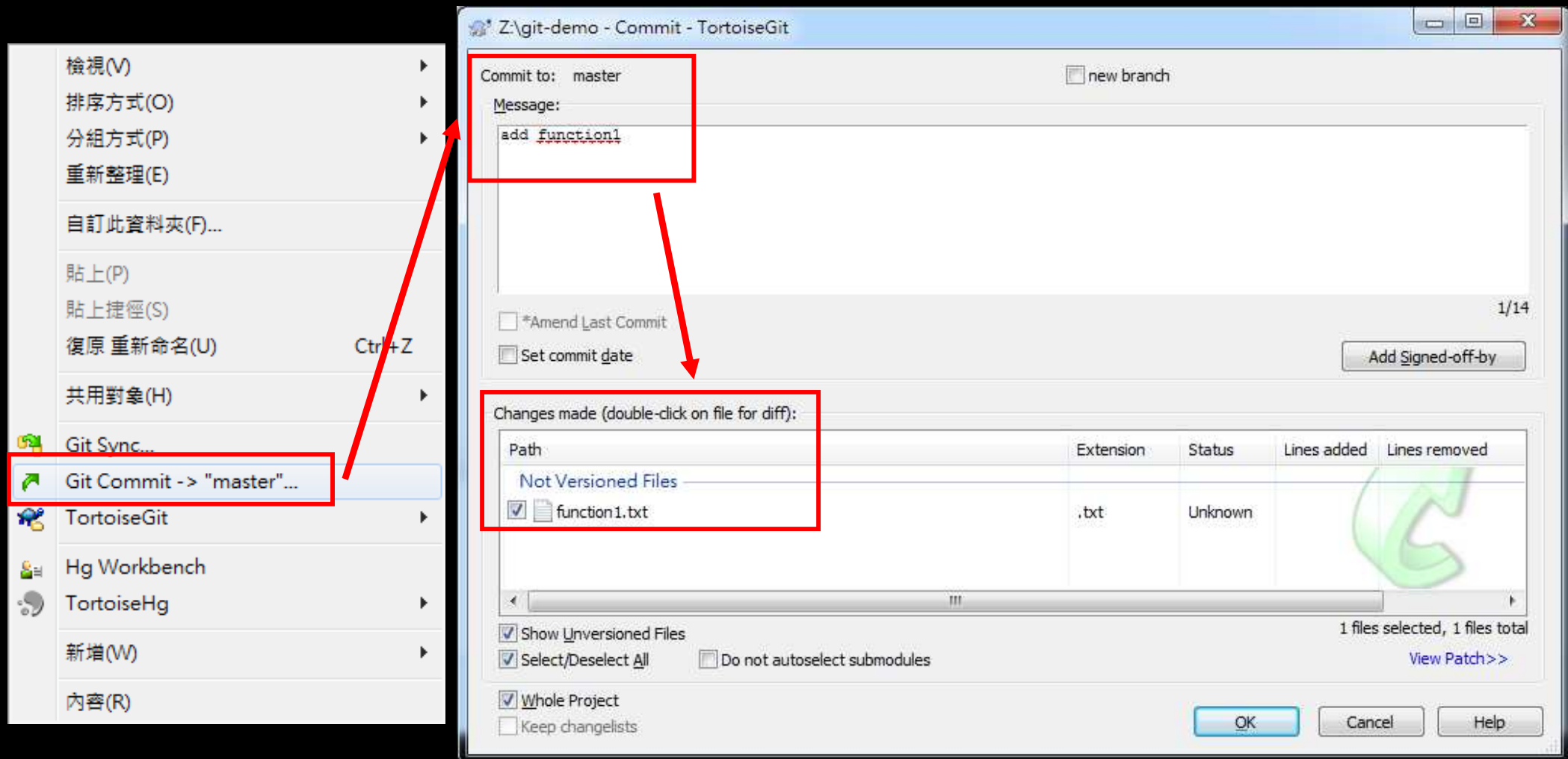
Git GUI工具教學(TortoiseGit)

建立repository



Git GUI工具教學(TortoiseGit)

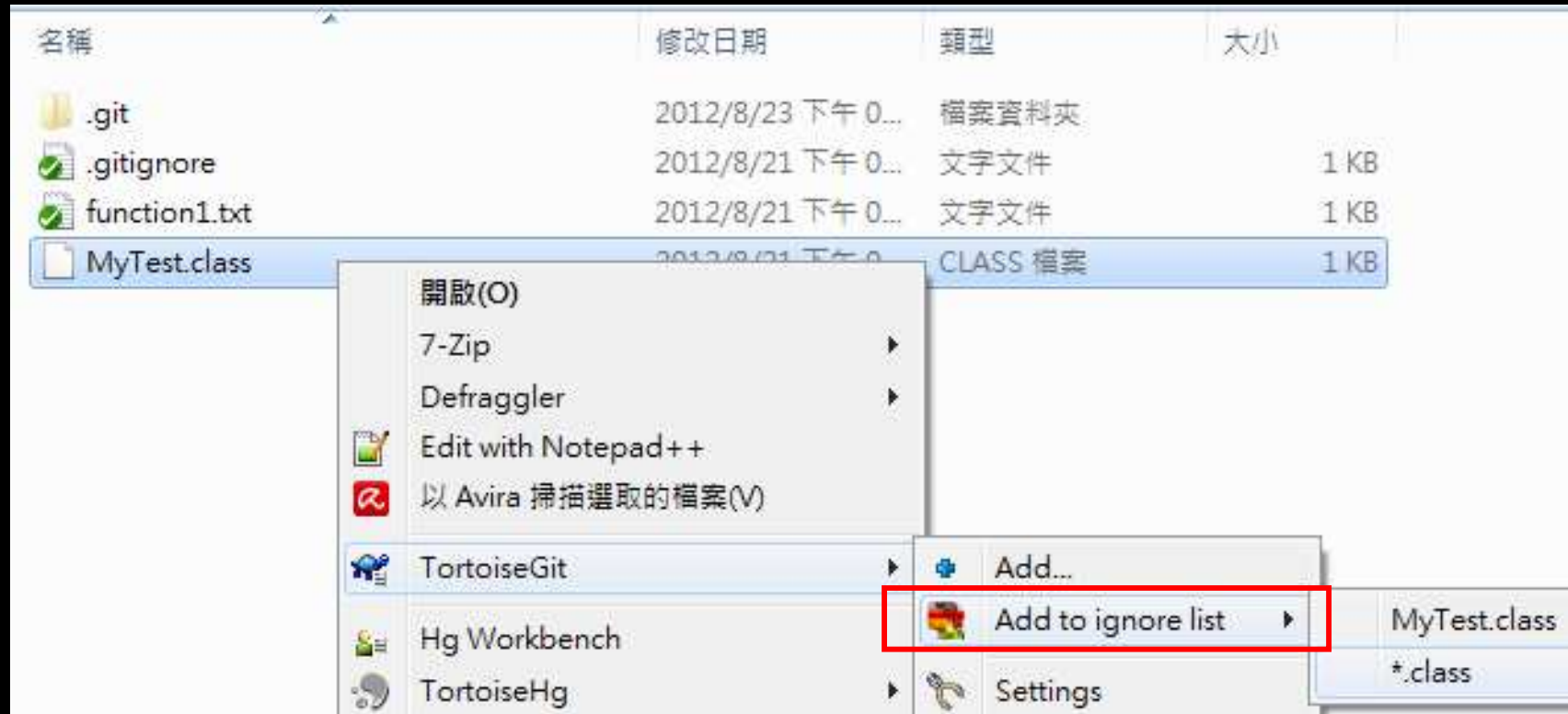
將working directory資料送進repository



Git GUI工具教學(TortoiseGit)

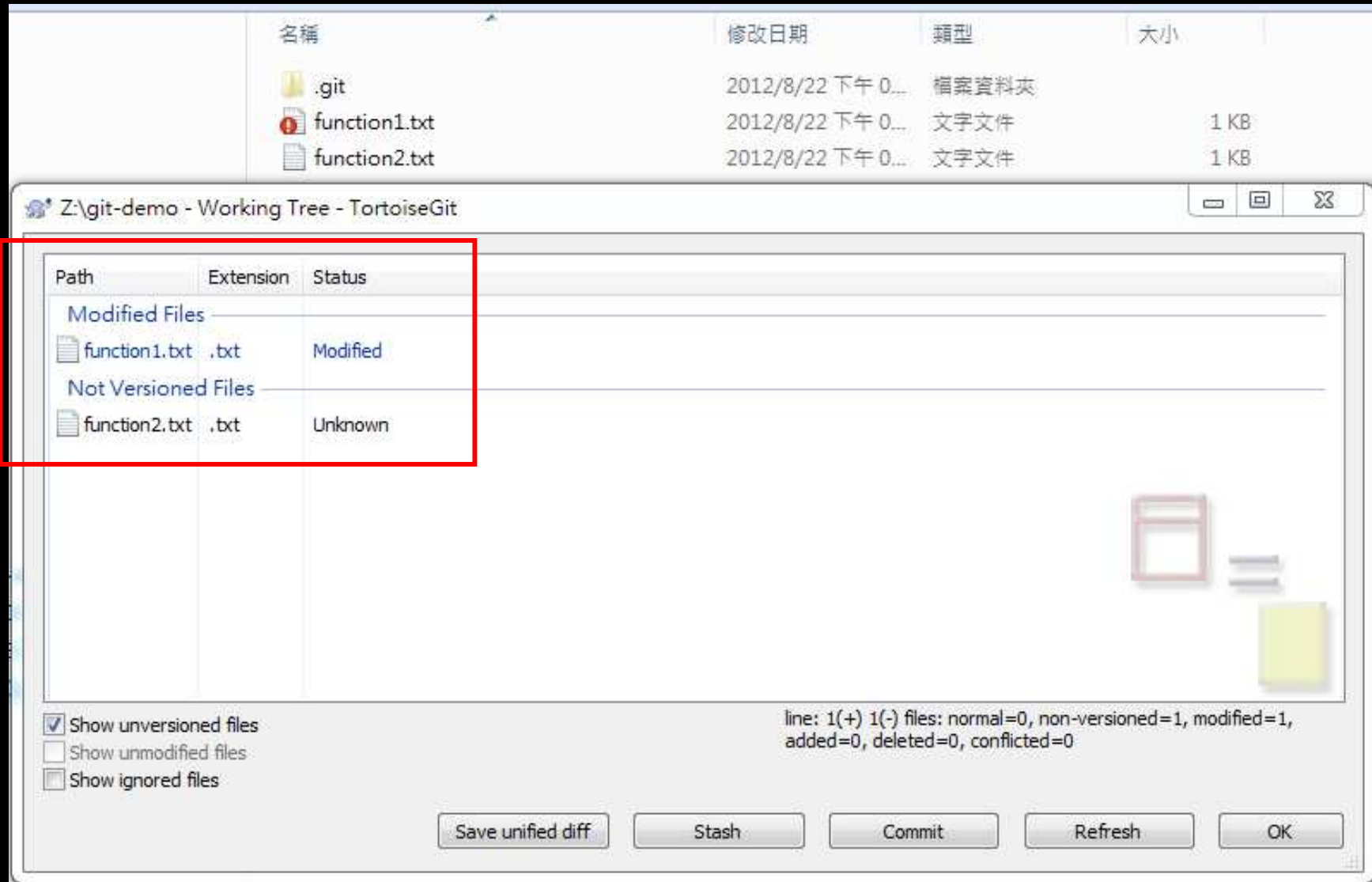
- .gitignore

– 告訴Git哪些檔案類型不用進版控



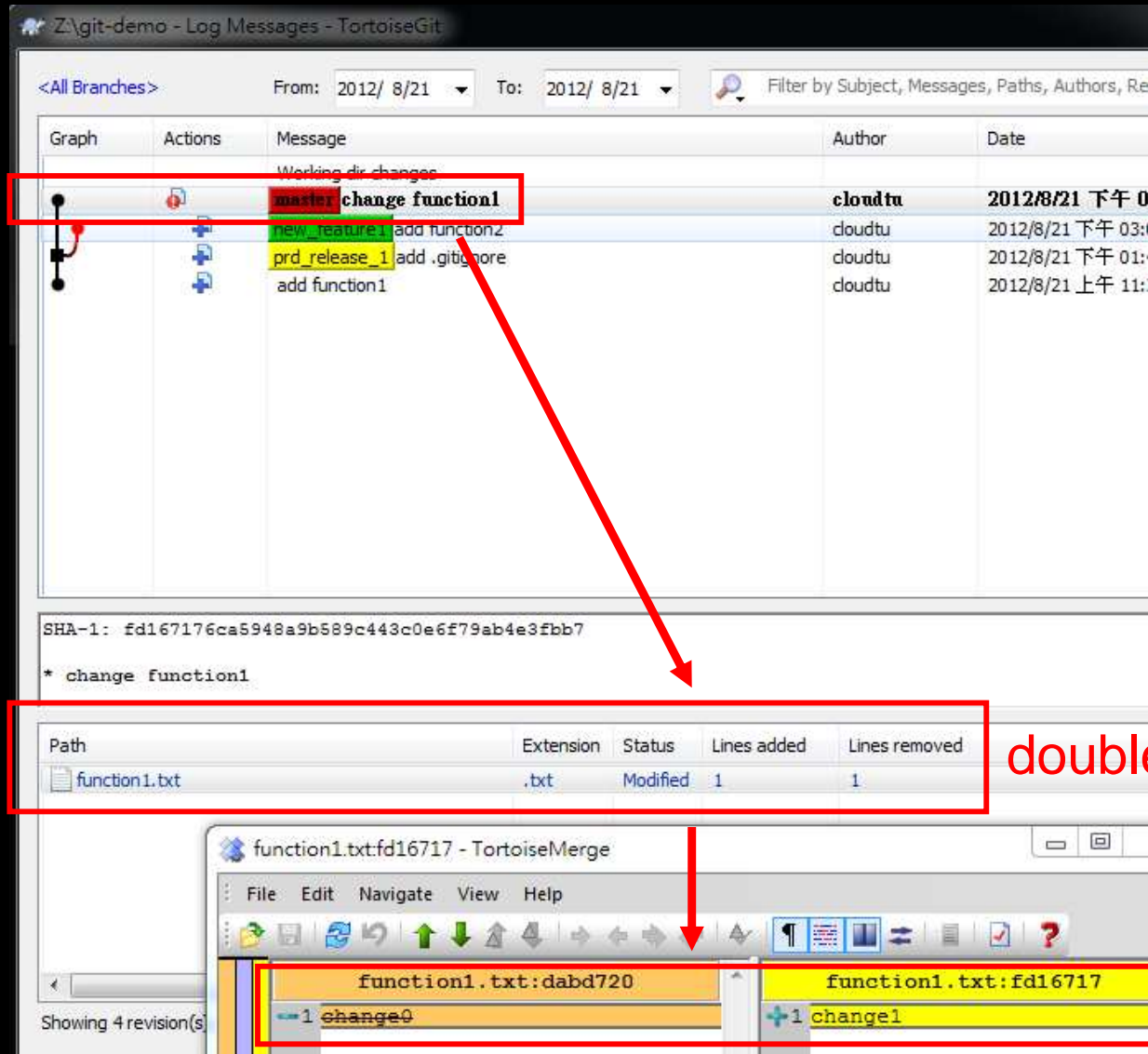
Git GUI工具教學(TortoiseGit)

- commit之前看檔案內容差異
 - TortoiseGit → Diff



Git GUI工具教學(TortoiseGit)

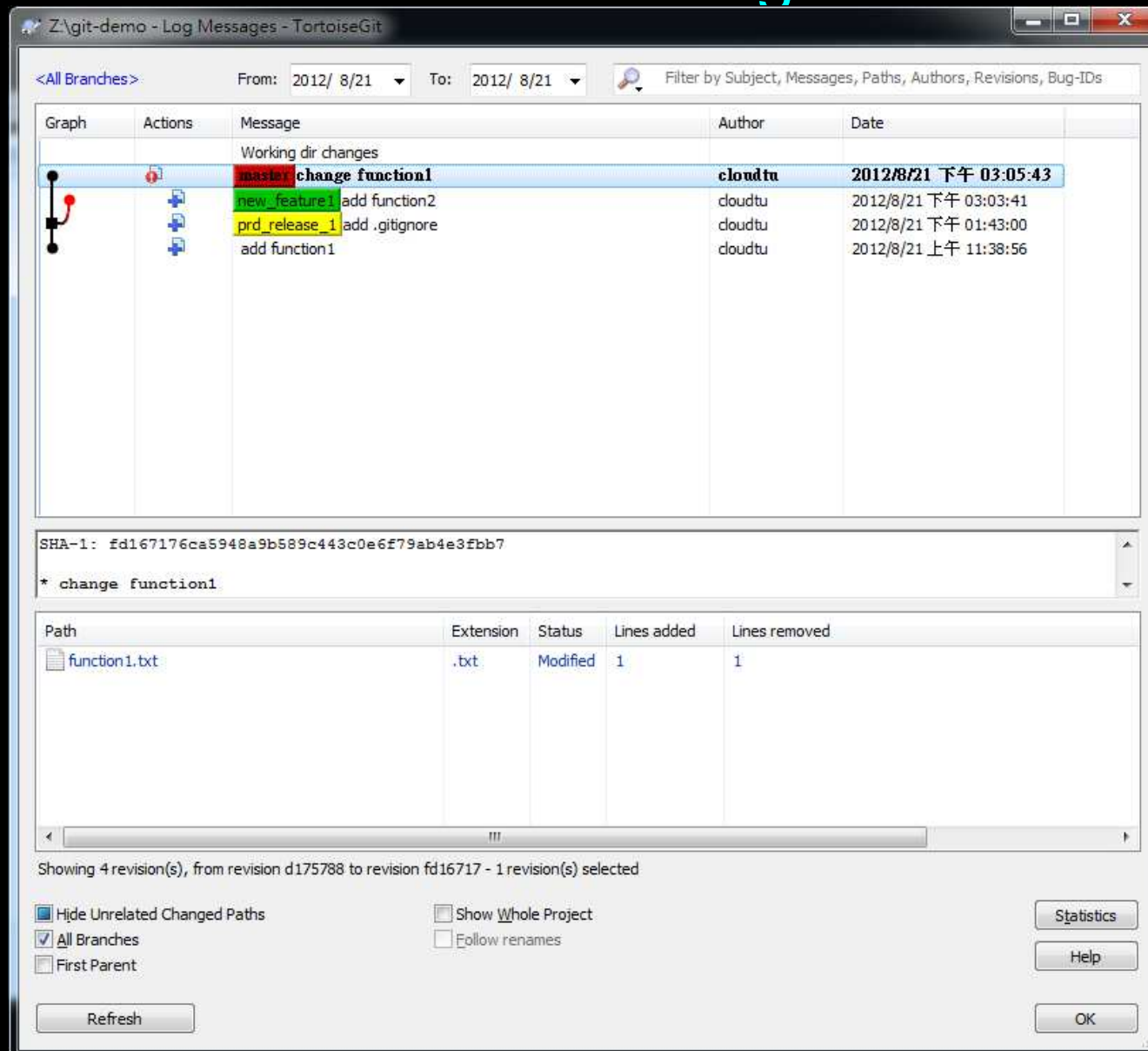
- commit之後看檔案內容差異
 - TortoiseGit → Show Log



double click it

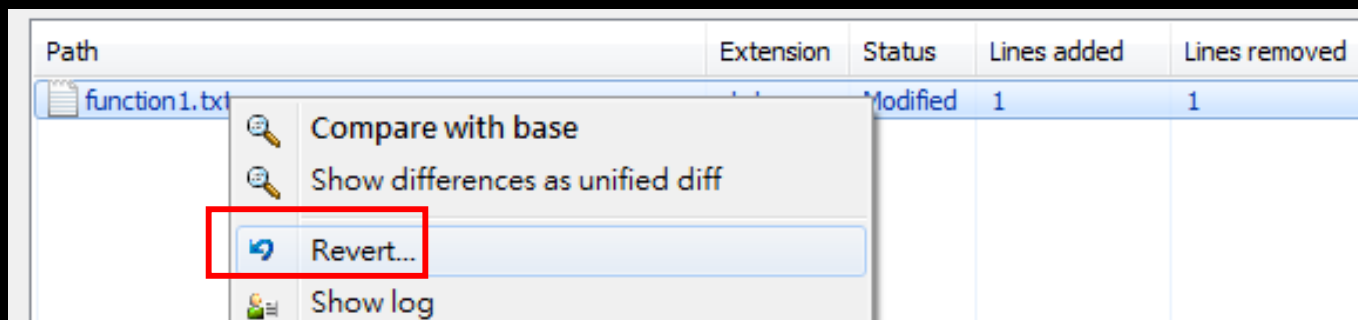
Git GUI工具教學(TortoiseGit)

- 查看commit的歷史記錄
 - TortoiseGit → Show Log

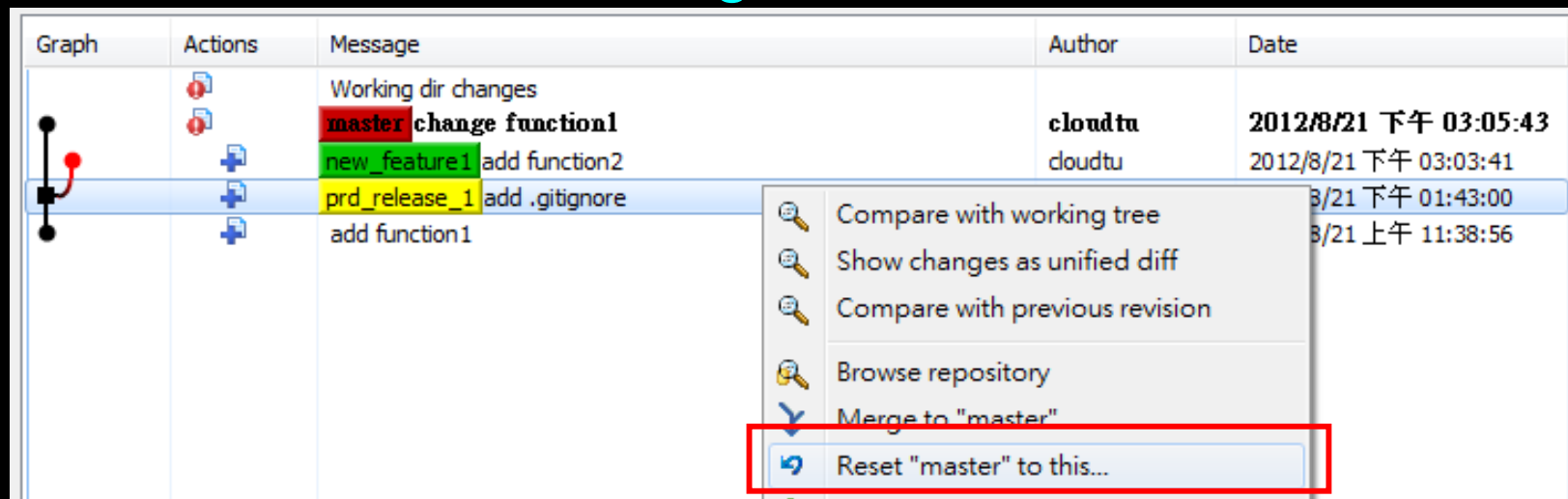


Git GUI工具教學(TortoiseGit)

- 檔案內容改爛了!怎麼辦?
 - commit之前改爛了，恢復到修改前的內容
 - TortoiseGit → Show Log → Revert



- commit之後後悔了，恢復到前一版的commit
 - TortoiseGit → Show Log → Reset



Git GUI工具教學(TortoiseGit)

使用branch(分支)

The screenshot shows the 'Log Messages' window in TortoiseGit for a repository named 'Z:\git-demo'. The window displays a list of commits with columns for 'Graph', 'Actions', and 'Message'. The 'Message' column shows the following commits:

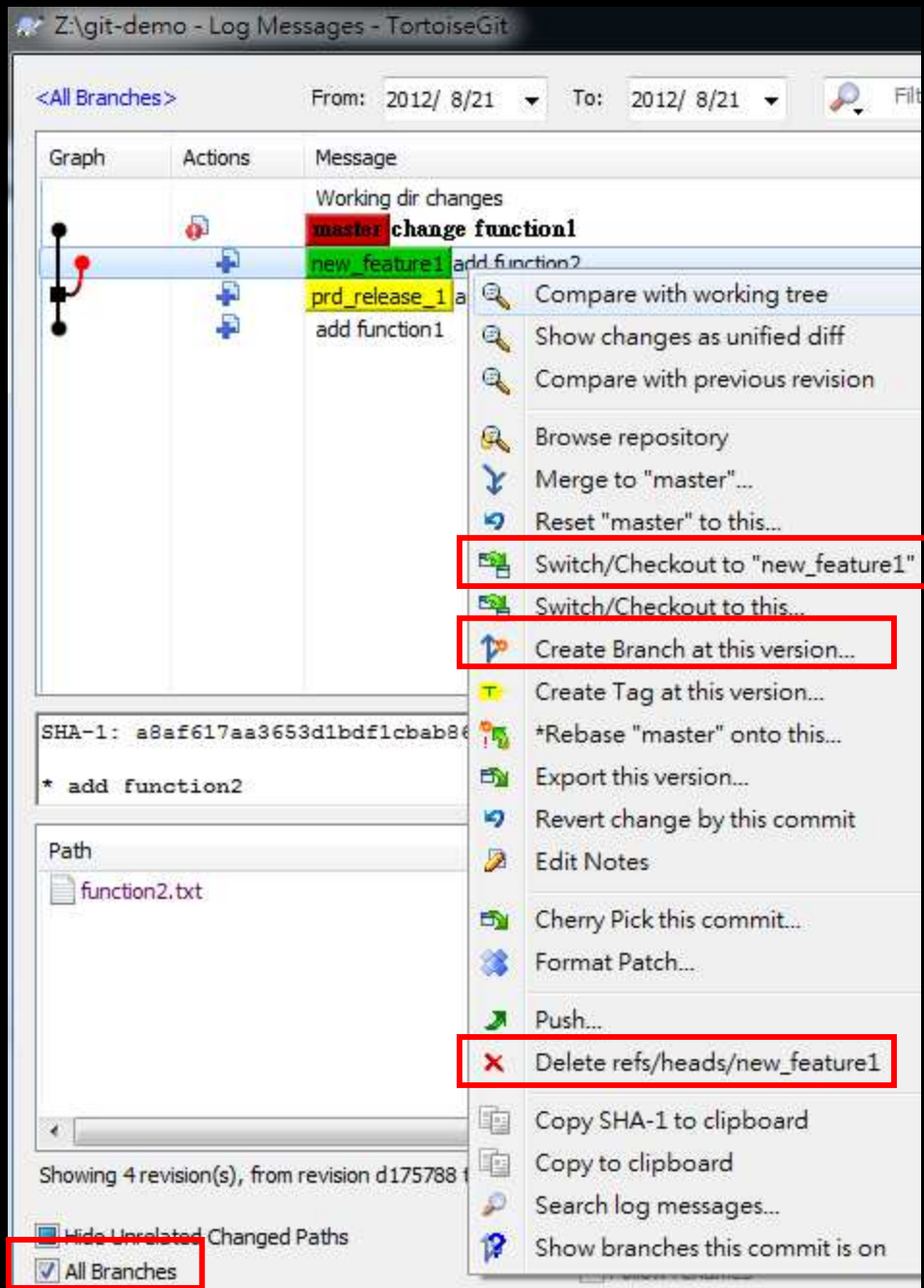
- Working dir changes
- master** change function1
- new_feature1 add function2
- prd_release_1 add .gitignore
- add function1

Three red callout boxes provide explanations for the branch colors:

- 紅底方塊表示目前正在使用的branch (Red block indicates the currently used branch)
- 綠底方塊表示目前未使用的branch (Green block indicates the currently unused branch)
- 黃底方塊表示tag (Yellow block indicates tag)

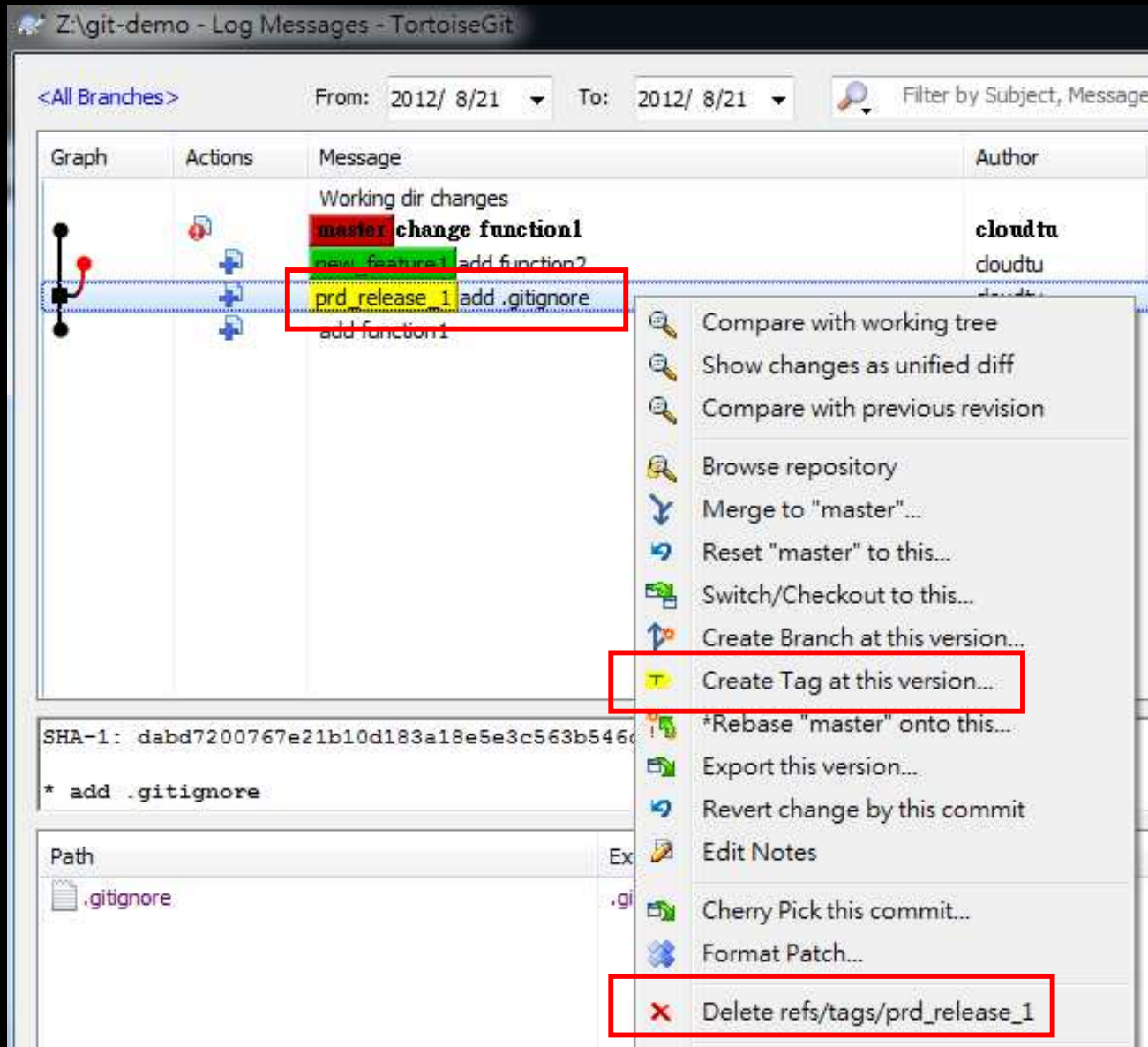
Git GUI工具教學(TortoiseGit)

新增、刪除、查詢、切換branch



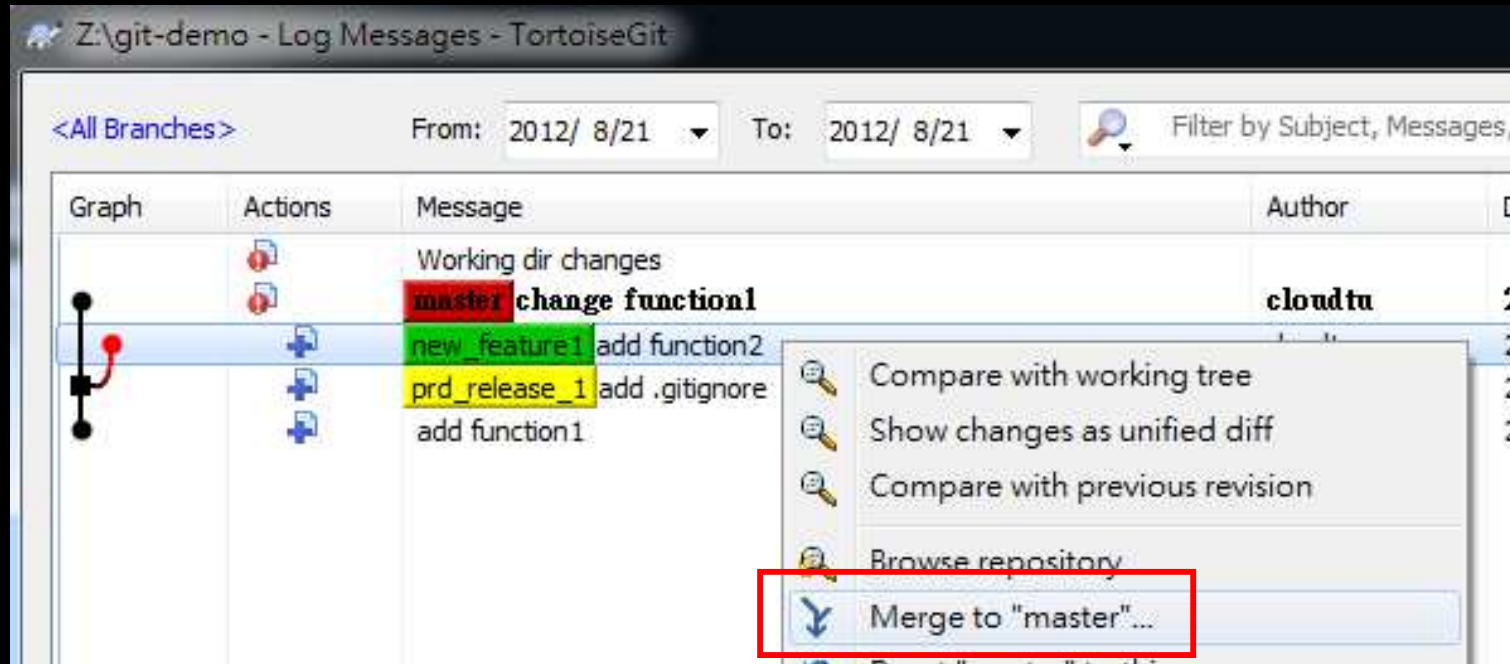
Git GUI工具教學(TortoiseGit)

新增、刪除、查詢tag



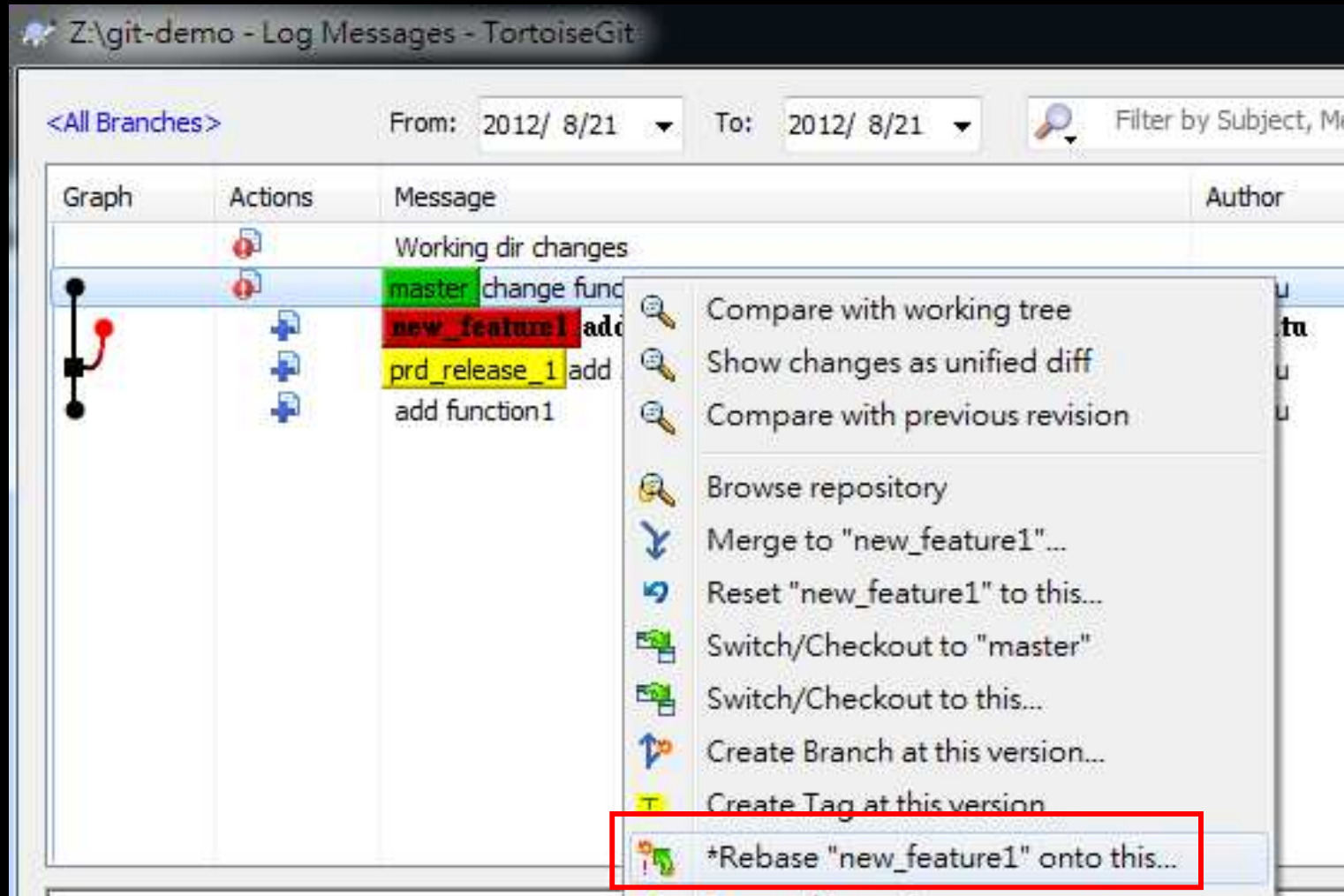
Git GUI工具教學(TortoiseGit)

- merge：把其它branch接在current branch的尾巴
- 範例



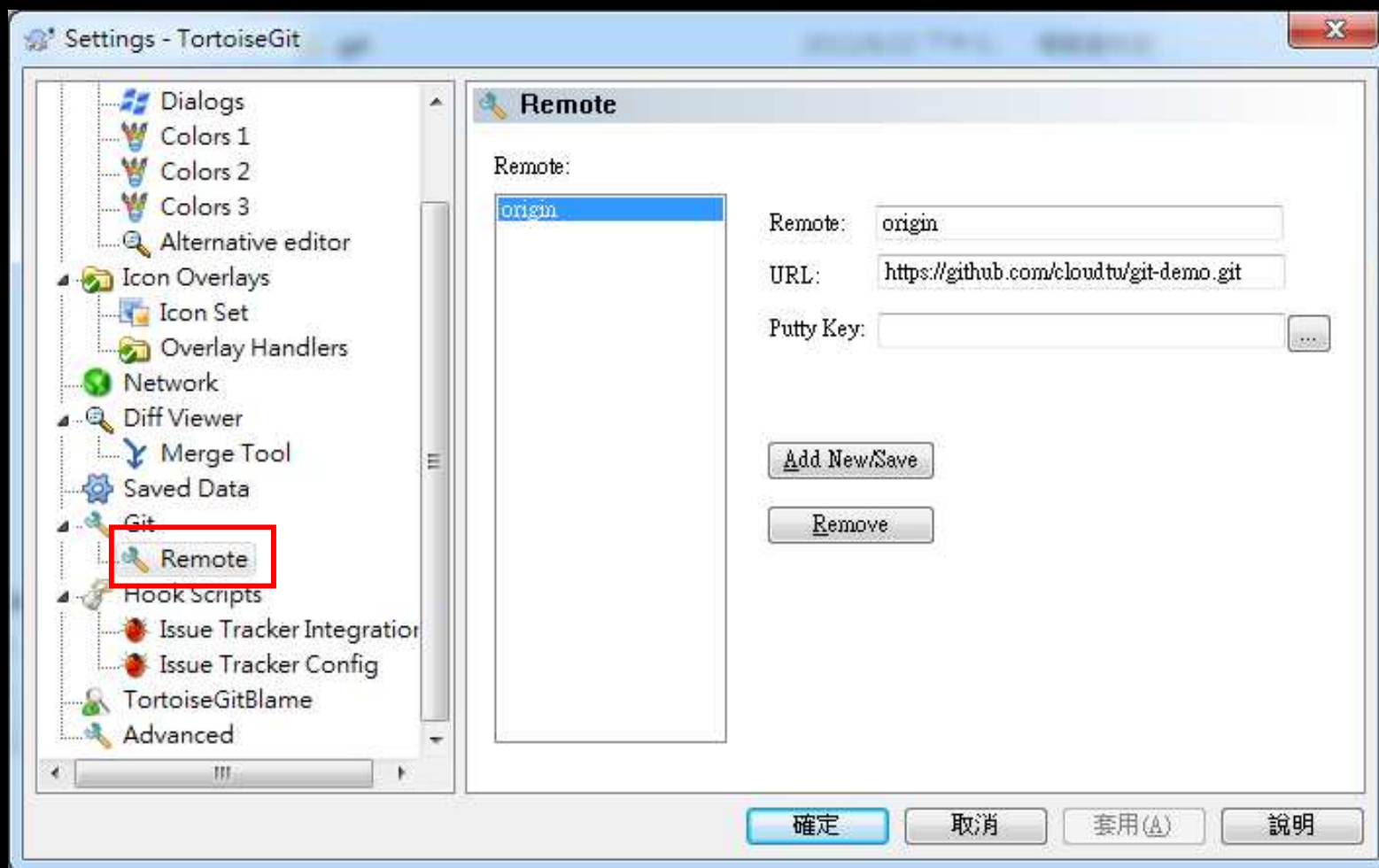
Git GUI工具教學(TortoiseGit)

- rebase : current branch的頭改接在其它branch之後
- 範例



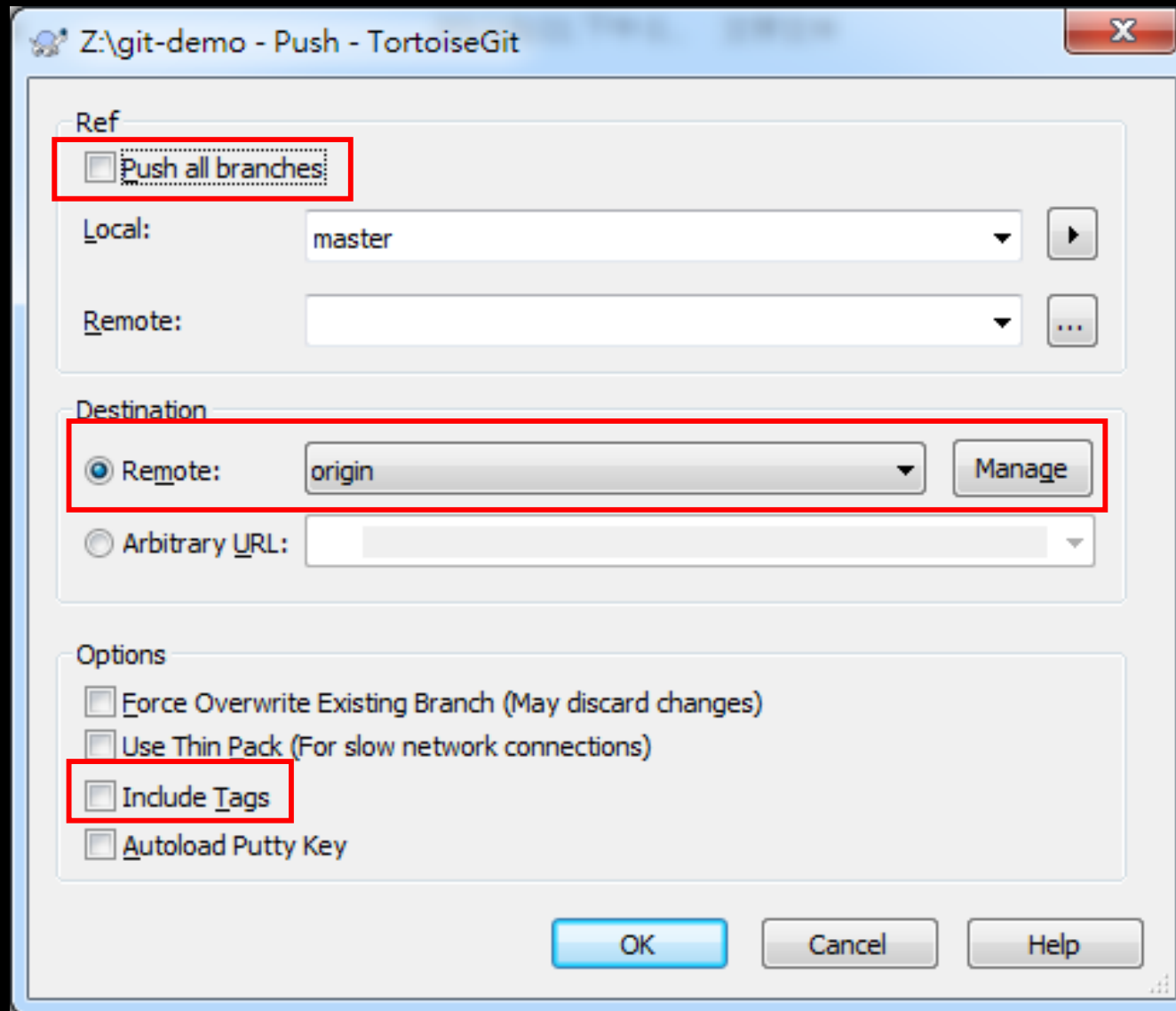
Git GUI工具教學(TortoiseGit)

- remote指令用來設定local端要連線至server端時的連線位址
 - TortoiseGit → Settings



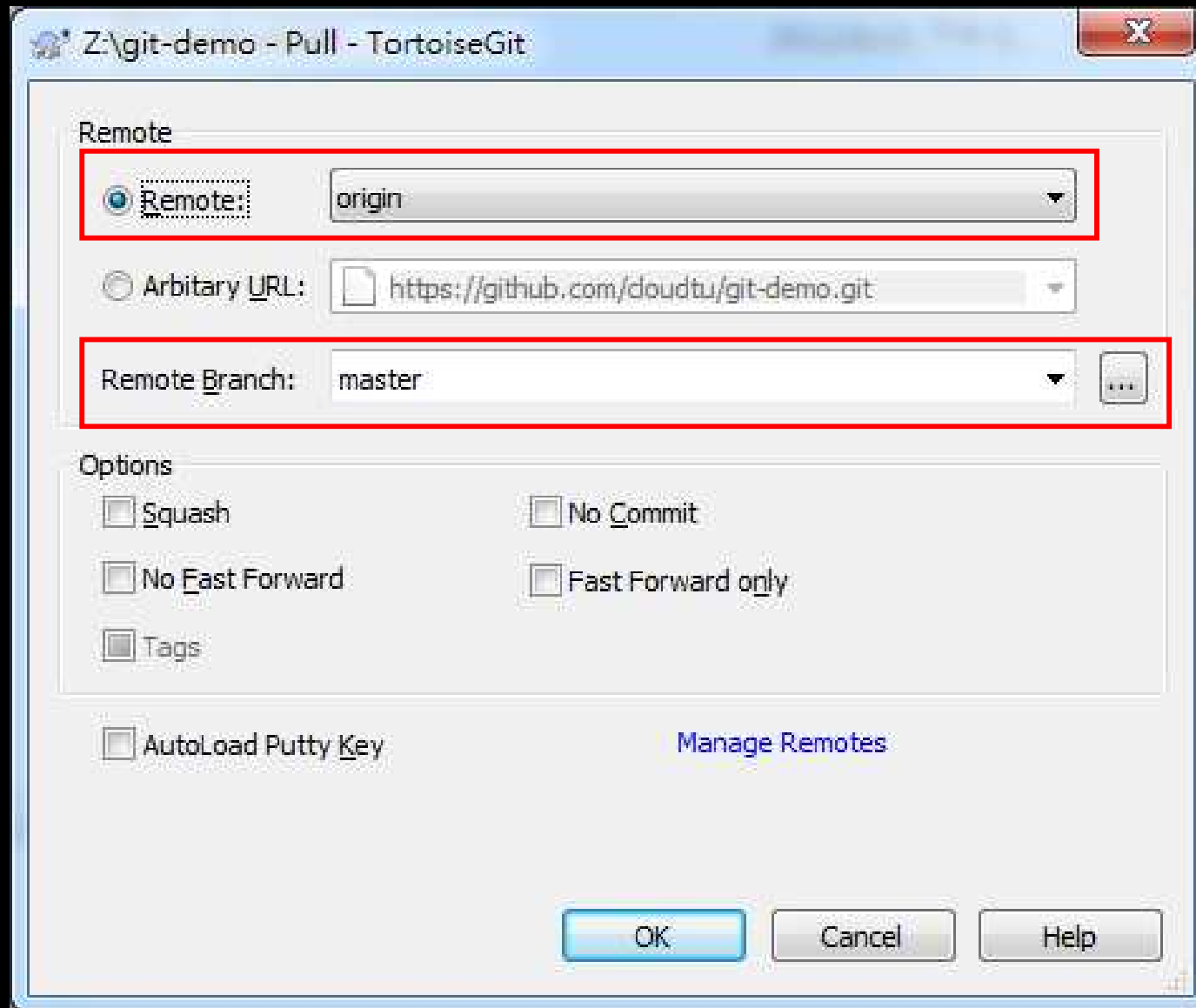
Git GUI工具教學(TortoiseGit)

- 把local端資料推送到server端
 - TortoiseGit → push



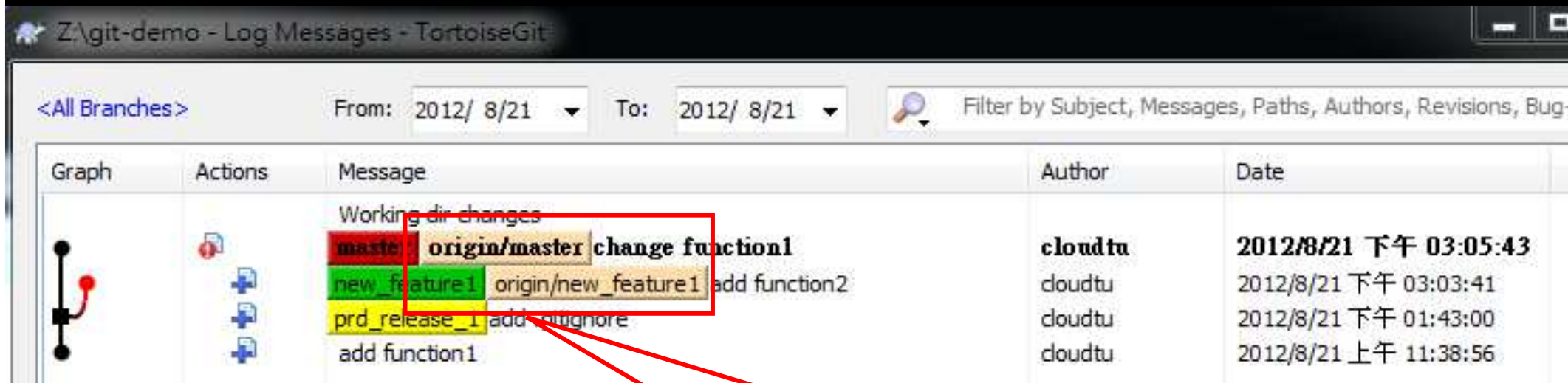
Git GUI工具教學(TortoiseGit)

- 把server端資料拉回並合併到local端
 - TortoiseGit → pull





Git GUI工具教學(TortoiseGit)

- push完成後的log



Z:\git-demo - Log Messages - TortoiseGit

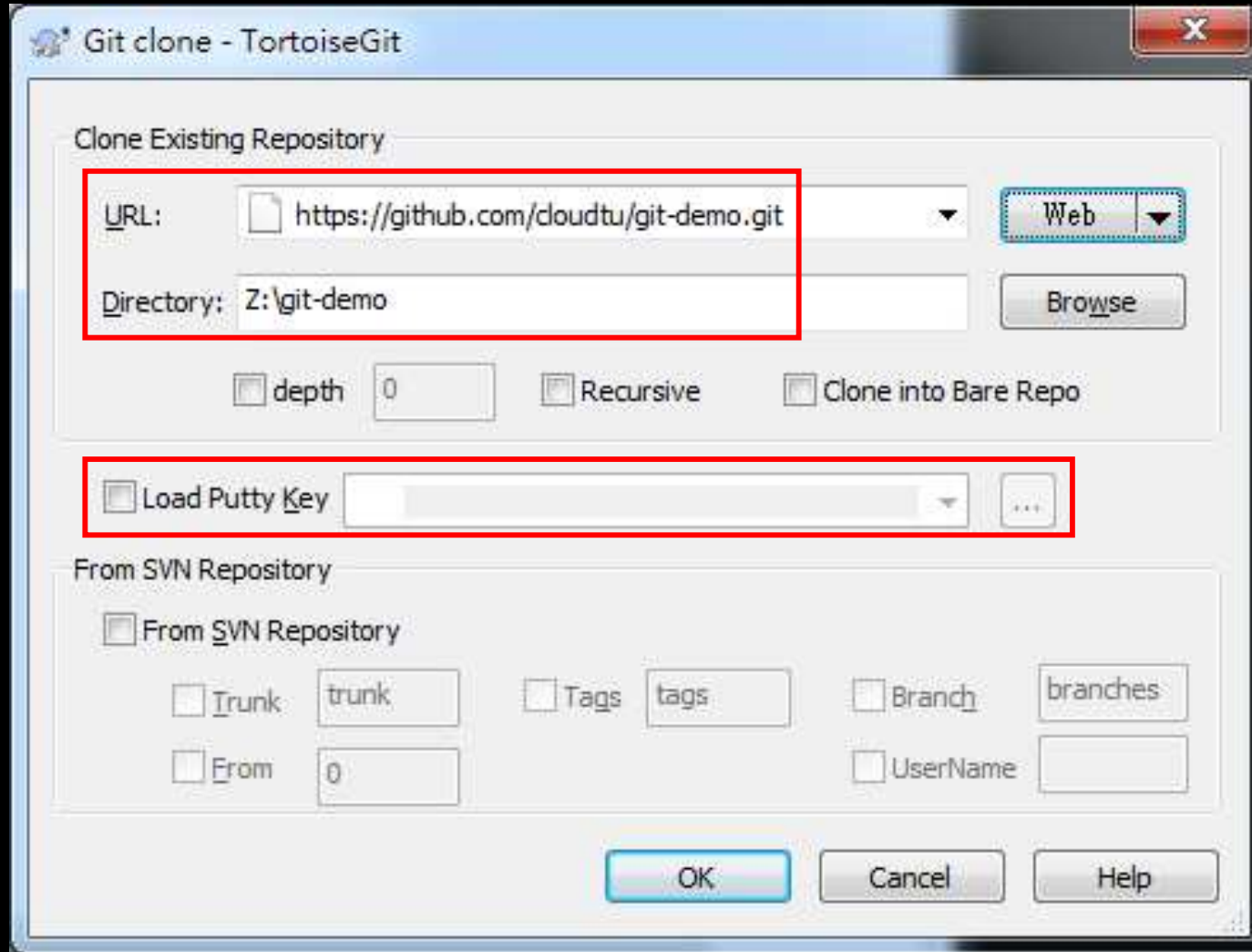
<All Branches> From: 2012/ 8/21 To: 2012/ 8/21 Filter by Subject, Messages, Paths, Authors, Revisions, Bug

Graph	Actions	Message	Author	Date
		Working dir changes		
		master: origin/master change function1	cloudtu	2012/8/21 下午 03:05:43
		new_feature1: origin/new_feature1 add function2	cloudtu	2012/8/21 下午 03:03:41
		prd_release_1: add ignore	cloudtu	2012/8/21 下午 01:43:00
		add function1	cloudtu	2012/8/21 上午 11:38:56

server端remote branch也會記錄在log裡面

Git GUI工具教學(TortoiseGit)

- clone一份server端repository至local端



Agenda

- 版控系統發展史
- Git設計目標
- 安裝Git
- Git指令教學
- 常用的Git指令
- Git GUI工具教學(TortoiseGit)
- **Git基本守則**
- 常用協同開發流程
- 參考資料

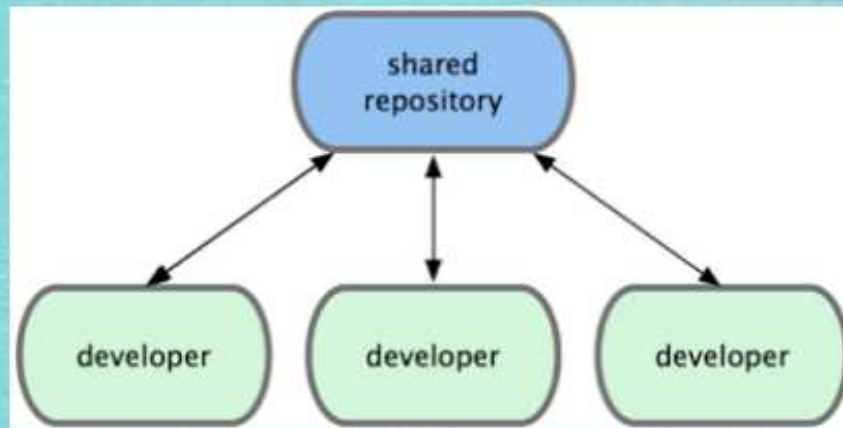
Git基本守則

- 編輯文字檔案請用UTF8編碼
- 設定.gitignore，只commit必要檔案
 - compiled binary、log、temp file不要放到repository
- commit守則
 - 每次commit只改一件事情
 - 寫清楚commit message
- 別對production repository下你不熟的git指令
- 「千萬不要」對已經push的東西作rebase

Agenda

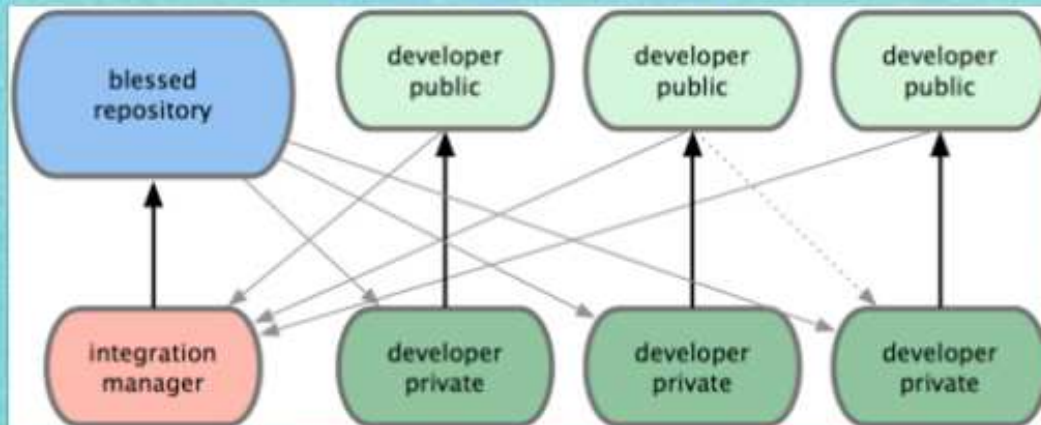
- 版控系統發展史
- Git設計目標
- 安裝Git
- Git指令教學
- 常用的Git指令
- Git GUI工具教學(TortoiseGit)
- Git基本守則
- 常用協同開發流程
- 參考資料

常用協同開發流程

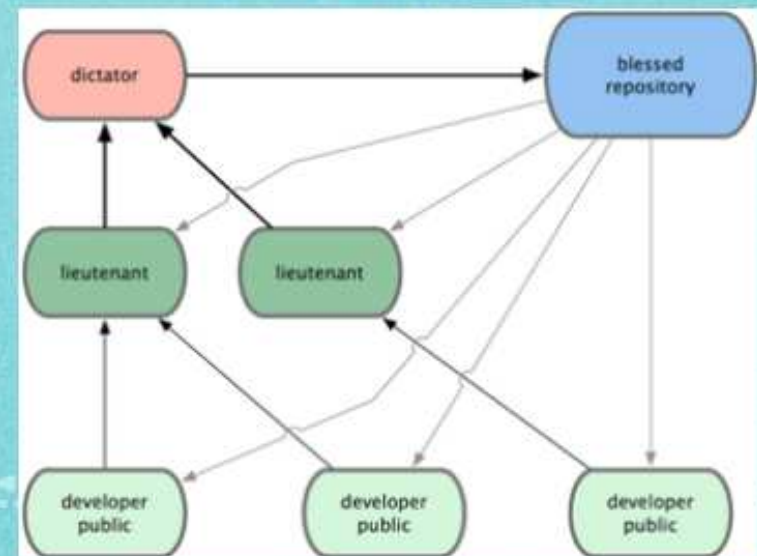


中央式的開發

整合管理員



司令官和副手



Agenda

- 版控系統發展史
- Git設計目標
- 安裝Git
- Git指令教學
- 常用的Git指令
- Git GUI工具教學(TortoiseGit)
- Git基本守則
- 常用協同開發流程
- 參考資料

參考資料

- 寫給大家的Git教學
 - <http://www.slideshare.net/littlebtc/git-5528339>
- Git in a nutshell(投影片)
 - <http://www.slideshare.net/Dannvix/git-in-a-nutshell>
- Git in a nutshell(影片)
 - <http://www.youtube.com/watch?v=1vkbR9itr0I>
- Git Tutorial
 - <http://www.slideshare.net/ihower/git-tutorial-13695342>
- TortoiseGit Intro in Traditional Chinese
 - <http://www.slideshare.net/jason8301/tortoisegit-intro-in-traditional-chinese>
- Stackoverflow
 - <http://stackoverflow.com/questions/tagged/git>