

# Git Learning Lesson One

---

Git Learning Lesson One v0.1

© 2015 MStar Semiconductor, Inc. All rights reserved.

MStar Semiconductor makes no representations or warranties including, for example but not limited to, warranties of merchantability, fitness for a particular purpose, infringement of any intellectual property right or the accuracy or completeness of this document, and reserves the right to make changes without further notice to any products herein to improve reliability, function or design. No responsibility is assumed by MStar Semiconductor arising out of the application or user of any product or circuit described herein; neither does it convey any license under its patent rights, nor the rights of others.

MStar is a trademark of MStar Semiconductor, Inc. Other trademarks or names herein are only for identification purposes only and owned by their respective owners.

## REVISION HISTORY

Revision No.	Description	Date
V0.1	<ul style="list-style-type: none"><li>{Initial release}</li><li></li></ul>	{06/24/2015}

## TABLE OF CONTENTS

REVISION HISTORY .....	i
TABLE OF CONTENTS.....	ii
LIST OF TABLES.....	iv
LIST OF FIGURES .....	iv
1. 前言.....	1
2. Git setup .....	2
2.1. login gerrit server with windows id/passwd .....	2
2.1.1 以 <b>browser</b> 登入 http://hcggit.mstarsemi.com.tw:8080 .....	2
2.2. create public key # 每換一個 build server 就要多添加一組 key.....	2
2.2.1 先確認登入 build server 帳號名稱與 windows 帳號名稱一樣 .....	2
2.2.2 在 <b>build server</b> 使用 ssh-keygen -t rsa 指令取得 ssh key. (passphrase 目前建議為空).....	2
2.2.3 copy content of .ssh/id_rsa.pub to http://hcggit.mstarsemi.com.tw:8080.....	2
2.2.4 ssh -p 29418 hcggit #test connection on buildserver .....	3
2.3. Setup user config on buildserver.....	4
2.3.1 git config --global user.email <b>you@example.com</b> .....	4
2.3.2 git config --global user.name " <b>Your Name</b> " .....	4
3. git 常用指令介紹.....	5
3.1. Download code.....	5
3.1.1 Git clone.....	5
3.1.2 Git pull .....	5
3.1.3 Git cherry-pick.....	5
3.2. Submit code.....	5
3.2.1 Git add .....	5
3.2.2 Git push.....	5
3.2.3 Git commit.....	6
3.2.4 Git status.....	6
3.2.5 Git diff.....	6
3.2.6 Git revert.....	6
3.3. Others.....	6
3.3.1 Git branch.....	6
3.3.2 Git log .....	6
3.3.3 Git checkout.....	7
3.3.4 Git reset .....	7
4. git 抓 code & 上 code 流程.....	8
4.1. Git 上 code 流程.....	8
4.1.1 Search project and get clone command.....	8
4.1.2 Clone project from git server.....	9
4.1.3 Switch branch .....	10
4.1.4 Modify code .....	10
4.1.5 Commit code.....	11
4.1.6 Push code.....	12
4.1.7 Detail information on git web .....	12
4.1.8 Find Reviewer to help code review.....	13

<b>5. Q&amp;A.....</b>	<b>14</b>
5.1. Q:Git 這麼多指令，有沒有跟 <b>performe</b> 操作的對應表阿？ .....	14
5.2. Q:為什麼在做 <b>git</b> 操作的時候，一直發生下圖的 <b>error</b> .....	14
5.3. Q:為什麼在 <b>git push</b> 的時候跳出找不到 <b>Change-Id</b> 的錯誤.....	15
5.4. Q:為什麼 <b>module owner merge code</b> 時，會遇到下圖的問題？ .....	15
5.5. Q:如果我覺得我的 <b>code base</b> 壞掉了，想要重抓一包 <b>code</b> ，只能整個目錄砍掉重抓嗎？ .....	16
5.6. Q:如果想要很多個不同版本的 <b>code base</b> ，只能 <b>clone</b> 很多份 <b>code</b> 嗎？ .....	17

## LIST OF TABLES

找不到圖表目錄。

## LIST OF FIGURES

Figure 3 search project.....	8
Figure 4 clone project.....	9
Figure 5 clone command.....	9
Figure 6 branches .....	10
Figure 7 switch branch .....	10
Figure 8 modify file .....	10
Figure 9 commit description window .....	11
Figure 10 example of commit message.....	11
Figure 11 commit done.....	11
Figure 12 push code to master .....	12
Figure 13 commit list on git web.....	12
Figure 14 commit view on git web .....	13

## 1. 前言

---

因應 `git` 全面化推行，是故撰寫文件來簡介 `git` 的基本使用方式，希望能讓大家更輕鬆的從 `perforce` 環境切換到 `git` 上。本文件將簡單的介紹從設定到上 `code` 的流程，以及一些基本的操作與觀念。

## 2. GIT SETUP

## 2.1. login gerrit server with windows id/passwd

在 git web server 上建立 buildserver 的 ssh key 資訊，設定連線的權限。

### 2.1.1 以 **browser** 登入 <http://hcgjt.mstarsemi.com.tw:8080>

[Sign In](#)

Updated

Press '?' to view keyboard shortcuts

Powered by [Gerrit Code Review \(2.6-rc0\)](#) | [Report Bug](#)

## 2.2. create ssh public key on build server

### 2.2.1 登入任意一台 build server，並確認登入 build server 帳號名稱與 windows 帳號名稱一樣

### 2.2.2 在 **build server** 使用 `ssh-keygen -t rsa` 指令取得 ssh key. (passphrase 目前建議為空)

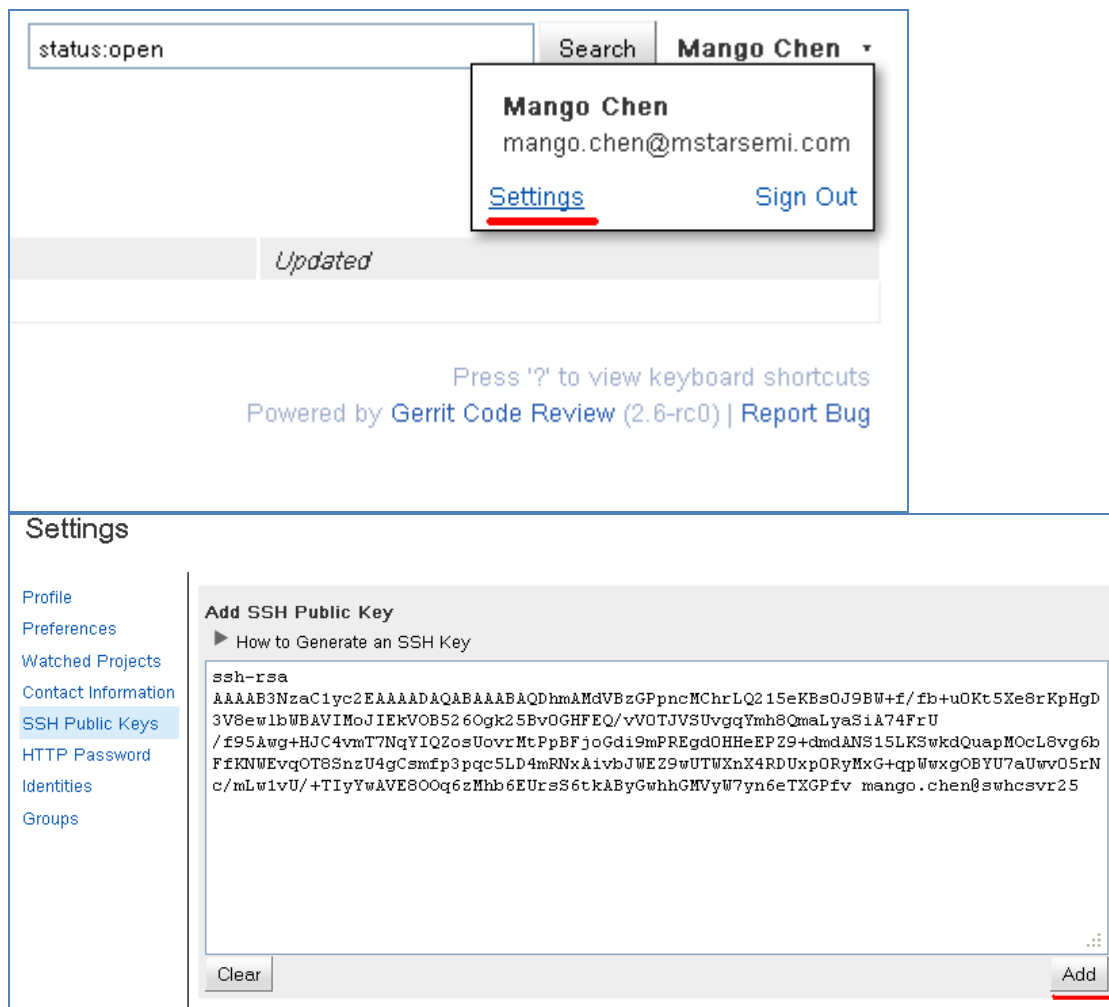
```
mango.chen@mango.chen$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/users/mango.chen/.ssh/id_rsa):
Created directory "/users/mango.chen/.ssh".
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /users/mango.chen/.ssh/id_rsa.
Your public key has been saved in /users/mango.chen/.ssh/id_rsa.pub.
The key fingerprint is:
09:d7:07:52:e9:41:22:c8:fe:14:8a:7e:88:bc:e5:ec mango.chen@swlcsvr25
The key's randomart image is:
+--[ RSA 2048 ]-----+
|      .  .  oo+.      |
|      o  .  +o.      |
|      o  .  .  .  .  .  |
|      .  o  .o  .  .  .  |
|  .o  .  o  S        |
|  .o.  .  .          |
|      =.             |
|      .  o           |
|      .E            |
+-----+

```

### 2.2.3 copy content of .ssh/id\_rsa.pub to <http://hcgit.mstarsemi.com.tw:8080>

```
mango.chen@mango.chen$ cat .ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDAImAdVbZGPPncMChrLQ215eKBS0tJ9BW+f/fb+u0Kt5Xe8rKpHgD3V8ew1bWBAVIMoJTEkV0
osUovrMtPbPfJgGdI9mPREgd0HHePZ9r+chdAdAS15LKSwdkQuapM0cL8vg6bFvKKNwEvq0T8Suz4gCsmfp3pqc5LD4mRNXaivbJWEZ9T1UWXhX
B7GubhH6vXGPRFgEzGmChen.chen@subhcs.cn$
```





The screenshot shows the Gerrit Code Review interface. At the top, there's a search bar with "status:open" and a user profile for "Mango Chen" with email "mango.chen@mstarsemi.com". Below the search bar, there's a "Updated" status bar. The main content area shows a message "Press '?' to view keyboard shortcuts" and "Powered by Gerrit Code Review (2.6-rc0) | Report Bug". The "Settings" page is open, showing a sidebar with links like Profile, Preferences, Watched Projects, Contact Information, SSH Public Keys (selected), HTTP Password, Identities, and Groups. The main content area is titled "Add SSH Public Key" and contains a text area with a long SSH public key string. There are "Clear" and "Add" buttons at the bottom.

注意：每換一個 build server 就要多添加一組 key

## 2.2.4 test connection on buildserver

command: `ssh -p 29418 hcgit`

```
mango.chen:~$ ssh -p 29418 hcgit
The authenticity of host '[hcgit]:29418 ([172.16.20.7]:29418)' can't be established.
RSA key fingerprint is 7f:9f:7e:87:b6:75:99:f5:65:6c:af:16:9e:b1:9a:61.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[hcgit]:29418,[172.16.20.7]:29418' (RSA) to the list of known hosts.

**** Welcome to Gerrit Code Review ****

Hi Mango Chen, you have successfully connected over SSH.

Unfortunately, interactive shells are disabled.
To clone a hosted Git repository, use:

git clone ssh://mango.chen@hcgit:29418/REPOSITORY_NAME.git

Connection to hcgit closed.
```

## 2.3. Setup user config on buildserver

2.3.1 git config --global user.email **you@example.com**

2.3.2 git config --global user.name **"Your Name"**

### 3. GIT 常用指令介紹

---

以下簡單介紹後續會用到的基本 git 指令的功能

#### 3.1. Download code

- git clone : clone 一份 server 上的 data 到 local 端.
- git pull : 與 server sync , update local 端的 history data
- git cherry-pick : 從網頁上 sync 尚未 submit 的 commit 到 local 端
- 或者是從其他的 branch sync commit 到想要的 branch 上

##### 3.1.1 Git clone

Git clone 使用於第一次從 server 抓 code 下來的時候，可以利用 clone 指令從 server 端 copy 一份 code tree data 到 local server 。

##### 3.1.2 Git pull

Git pull 使用於已經抓好 code 之後，想要將 local 端的 code tree data，更新至與 server 端一致時。可以利用 pull 指令從 server 端 sync data 回 local server 做 update 。

##### 3.1.3 Git cherry-pick

Git cherry-pick 使用於抓好 code 以後，想要將某個 solution 的 commit，合到自己的 local 端 code tree 時。可以利用 cherry-pick 指令將來源端的改動 merge 進 local 的 code tree 中。

#### 3.2. Submit code

- git add : 將想要上 code 的檔案加入 submit file list 中
- git push : 將準備上 code 的 commit 推上 review system 等待 review
- git commit : 產生一個新的 commit 或是 修改目前 local 端已新增的最新的 commit
- git status : 檢查目前已改動的檔案有哪些
- git diff : 顯示出目前改動的 code 的詳細情況 或是比對 兩個 commit/branch 間的差異
- git revert : 將已 merge 的 commit revert 並生成一個 commit

##### 3.2.1 Git add

Git add 用於修改完 local code base 以後，使用 add 指令將準備要 push 上 review system 的檔案，加進 commit file list 中。可利用 `git add [filename]` 加入單一檔案，或是利用 `git add .` 來加入目前 project 中所有修改到的檔案。

##### 3.2.2 Git push

Git push 用於建立好準備上 code 的 commit 以後，使用 push 指令將 commit 推上對應的 branch 並上至 review system，以等待 code review 。

Branch list 可以透過 `git branch -av` 來查詢。

### 3.2.3 Git commit

Git commit 用於 git add 完成以後，使用 git commit 新增新的 commit，或是 git commit --amend 來修改目前 local 端以新增的最新的 commit。

### 3.2.4 Git status

Git status 用於需要檢視目前有哪些檔案被改動到時。

**注意：**若修改的檔案，已透過 git add, git commit 新增成一個 commit。git status 將看不到這些檔案。

這個指令是與目前 local commit history 中最新的狀況來做比對，僅列出尚未新增成 commit 的檔案。

### 3.2.5 Git diff

Git diff 用於檢視 git status 可以查看到的檔案的修改。或者是利用 git diff commit1...commit2 來檢視兩個 commit 間的差異。甚至 tag, branch 亦可利用此方法比對修改差異。

### 3.2.6 Git revert

Git rever 用於需要將已 merge 的 commit 退掉的時候。執行後會自動產生一個新的 commit，一樣需要進行 git push 動作，透過 code review 來完成上 code。

## 3.3. Others

- git branch：查看 branch 資訊
- git log：查看 local 端 commit history
- git checkout：切換/新增 local branch
- git reset：重設 local branch code base 狀況

### 3.3.1 Git branch

Git branch 用於對 branch 的操作，包含顯示目前 local branch，顯示遠端 server branch，刪除/新增 local branch 等。

- |                                   |                             |
|-----------------------------------|-----------------------------|
| 3.3.1.1. git branch -av           | # show all branches         |
| 3.3.1.2. git branch -r            | # show remote branches      |
| 3.3.1.3. git branch               | # show local branches       |
| 3.3.1.4. git branch BranchName    | # create a new local branch |
| 3.3.1.5. git branch -d BranchName | # delete a local branch     |

### 3.3.2 Git log

Git log 用於檢視目前 code tree 的 history 情況。

### 3.3.3 Git checkout

Git checkout 用於切換目前 local branch 的操作，包含退 code，切換 branch，等功能。

3.3.3.1. git checkout BranchName # switch to exist branch

3.3.3.2. git checkout -b BranchName # create and switch to a new local branch

3.3.3.3. git checkout --track aosp/ BranchName # create and switch to a new branch to track remote branch

3.3.3.4. git checkout \$commit-id # checkout to null branch with commit-id

### 3.3.4 Git reset

Git reset 用於重新設定 local code tree，移除新增的 commit，或是退 local commit。

3.3.4.1. git reset --hard HEAD~2 # 退到前兩版並刪除對檔案的變更

3.3.4.2. git reset --soft HEAD~1 # 退到前一版並保留對檔案的變更

## 4. GIT 抓 CODE & 上 CODE 流程

### 4.1. Git 上 code 流程

Git 上完成整個上 code 的流程，一共有幾個步驟。

抓 code -> 修改 code -> 新增 commit -> push commit to review system -> review pass -> merge to remote server

以上步驟，除了最後兩步以外，都是 **developer**（也就是要改 code 的人）必須執行的步驟。

Review 步驟，是由該 module reviewer 負責。

Merge 步驟，是由該 module owner 完成。

**注意：**『必須執行到 **merge** 步驟完成，才算真的完成整個上 **code flow**。』

沒有完成 **merge** 步驟以前，其他人僅能透過 **cherry-pick** 取得修改，無法透過 **git clone** 來抓到 **code**。

以下章節，舉 utopia 修改為例，來說明 git 上 code 到 code review system 的流程。

#### 4.1.1 Search project and get clone command

Go to website : <http://hcgkit:8080> and login your account. Then switch to project page and search pitaya project.



Figure 1 search project

Select pitaya project and we will get the detail description of this project as Figure 4. And follow the sequence in Figure 4. to generate a clone command.

All My **Projects** People Documentation

List General Branches Access Dashboards

### Project device/mstar/pitaya

clone **clone with commit-msg hook** <sup>1</sup> Anonymous HTTP **SSH** <sup>2</sup> HTTP <sup>3</sup>

git clone ssh://feng.lin@hcggit:29418/device/mstar/pitaya && scp -p -P 29418 feng.lin@hcggit:hooks/commit-msg pitaya/.git/hooks

**Description**

**Project Options**

State:

Submit Type:

Automatically resolve conflicts:

Require Change-Id in commit message:

Maximum Git object size limit:

**Contributor Agreements**

Require Signed-off-by in commit message:

Figure 2 clone project

#### 4.1.2 Clone project from git server

Go back to console and paste the clone command copied from step 4.1.1.  
We will get a clone project pitaya.

```
[feng.lin@hcbcsvr6407 l-pitaya]$ git clone ssh://feng.lin@hcggit:29418/device/mstar/pitaya && scp -p -P 29418 feng.lin@hcggit:hooks/commit-msg pitaya/.git/hooks/
Cloning into 'pitaya'...
remote: Counting objects: 3379, done
remote: Finding sources: 100% (3379/3379)
remote: Total 3379 (delta 1889), reused 3090 (delta 1889)
Receiving objects: 100% (3379/3379), 643.22 MiB | 2.44 MiB/s, done.
Resolving deltas: 100% (1889/1889), done.
Checking connectivity... done
commit-msg 100% 4360 4.3KB/s 00:00
[feng.lin@hcbcsvr6407 l-pitaya]$
```

Figure 3 clone command

Now you can see detail branch information by "git branch -avv"

```
[feng.lin@hcbcsvr6407 l-pitaya]$ ls
pitaya
[feng.lin@hcbcsvr6407 l-pitaya]$ cd pitaya/
[feng.lin@hcbcsvr6407 pitaya (master)]$ git branch -avv
* master          7a901e9 [origin/master] Initial empty repository
remotes/origin/HEAD -> origin/master
remotes/origin/kitkat-clippers-rel 935fd43 Service mdnsd already exist in init
remotes/origin/kitkat-kaizer-rel   94a9b63 Add AC3 Plus codec type for media_codecs.xml
remotes/origin/kitkat-madison-rel  7a901e9 Initial empty repository
remotes/origin/kitkat-madison-rel2 94a9b63 Add AC3 Plus codec type for media_codecs.xml
remotes/origin/kitkat-monaco-rel   94a9b63 Add AC3 Plus codec type for media_codecs.xml
remotes/origin/kitkat-mstar-master 94a9b63 Add AC3 Plus codec type for media_codecs.xml
remotes/origin/kitkat-muji-rel     bc395c0 Update utopia(CL@1016359)
remotes/origin/kitkat-napoli-rel   94a9b63 Add AC3 Plus codec type for media_codecs.xml
remotes/origin/lollipop-monaco-rel b149ff4 Set Mali CMA memory
remotes/origin/lollipop-mstar-master df884fc Recover a2dp output/input audio_policy.conf
remotes/origin/lollipop-muji-rel   8cc0ea7 Update build number to 1.0.0
remotes/origin/master              7a901e9 Initial empty repository
[feng.lin@hcbcsvr6407 pitaya (master)]$
```

Figure 4 branches

### 4.1.3 Switch branch

Switch to master line to get tip code base.

```
[feng.lin@hcbcsvr6407 pitaya (master)]$ git checkout -t origin/lollipop-mstar-master
Branch lollipop-mstar-master set up to track remote branch lollipop-mstar-master from origin.
Switched to a new branch 'lollipop-mstar-master'
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$
```

Figure 5 switch branch

注意：範例中為修改 **utopia**，故切換至 **lollipop-mstar-master**，請視情況切換至需要的 **branch**。

### 4.1.4 Modify code

Modify code and add it into stage.

```
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$ vi Android.mk
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$ git status
# On branch lollipop-mstar-master
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
#       modified:   Android.mk
#
no changes added to commit (use "git add" and/or "git commit -a")
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$ git add Android.mk
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$ git status
# On branch lollipop-mstar-master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   Android.mk
#
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$
```

Figure 6 modify file



#### 4.1.5 Commit code

After adding all modified files into stage, invoke "git commit" to show commit description window.



```
hcbcsvr6407 (hcbcsvr6407.mstarsemi.com.tw) [119x38]
連線(C) 編輯(E) 檢視(V) 視窗(W) 選項(O) 說明(H)

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch lollipop-mstar-master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   Android.mk
#
```

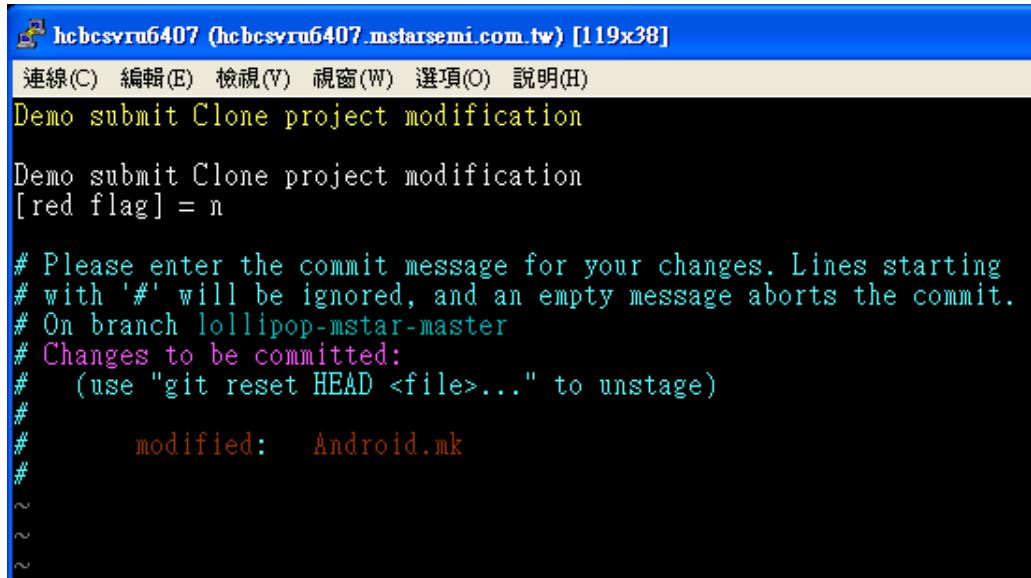
Figure 7 commit description window

And fill needed information in commit description.

Following is commit message rule documentation.

<http://hcgithtdocs/codestyle/Commit%20message.pdf>

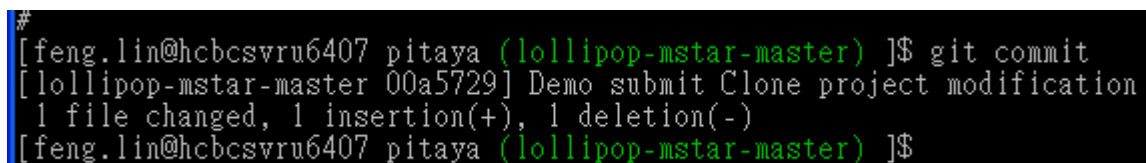
Ex:



```
hcbcsvr6407 (hcbcsvr6407.mstarsemi.com.tw) [119x38]
連線(C) 編輯(E) 檢視(V) 視窗(W) 選項(O) 說明(H)
Demo submit Clone project modification
Demo submit Clone project modification
[red flag] = n

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch lollipop-mstar-master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       modified:   Android.mk
#
~
~
~
```

Figure 8 example of commit message



```
#
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$ git commit
[lollipop-mstar-master 00a5729] Demo submit Clone project modification
1 file changed, 1 insertion(+), 1 deletion(-)
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$
```

Figure 9 commit done

#### 4.1.6 Push code

Push code to remote master line, and waiting review.

```
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$ git push origin HEAD:refs/for/lollipop-mstar-master
Counting objects: 5, done.
Delta compression using up to 48 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 346 bytes | 0 bytes/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2)
remote: Processing changes: new: 1, refs: 1, done
remote: RefName:refs/heads/lollipop-mstar-master
remote: NameEmail:Feng Lin <feng.lin@mstarsemi.com>
remote:
remote: New Changes:
remote:   http://hcggit:8080/20477
remote:
To ssh://feng.lin@hcggit:29418/device/mstar/pitaya
 * [new branch]      HEAD -> refs/for/lollipop-mstar-master
[feng.lin@hcbcsvr6407 pitaya (lollipop-mstar-master)]$
```

Figure 10 push code to master

注意：範例中為修改 **utopia**，故 **push** 至 **lollipop-mstar-master**，請視情況切換至需要的 **branch**。

#### 4.1.7 Detail information on git web

You can see detail commit information on git web review system.

All
My
Projects
People
Documentation

Changes
Drafts
Draft Comments
Watched Changes
Starred Changes

### My Reviews

Subject
<b>Outgoing reviews</b>
★ Skip disk check prevent sdcard/usb crash
★ Test clone project
★ Demo submit Clone project modification

Figure 11 commit list on git web

[All](#)
[My](#)
[Projects](#)
[People](#)
[Documentation](#)

[Changes](#)
[Drafts](#)
[Draft Comments](#)
[Watched Changes](#)
[Starred Changes](#)

Change **20477** - Needs Code-Review

Edit Message
Reply...
Code-Review+2

Demo submit Clone project modification  
Demo submit Clone project modification  
[red flag] = n

Owner [Feng Lin](#)  
Reviewers [Add...](#)  
Project [device/mstar/pitaya](#)  
Branch [lollipop-mstar-master](#)  
Topic [Add...](#)  
Strategy Fast Forward Only  
Uploaded 2 minutes ago  
Cherry Pick Abandon

Code-Review  
Verified

Author [Feng Lin <feng.lin@mstarsemi.com>](#)  
Committer [Feng Lin <feng.lin@mstarsemi.com>](#)  
Commit [00a5729a2fea7ca9e37c074716d521156e2991f7](#)  
Parent(s) [df884fc714d8d9f78926f21604bbdc79c68c2ed4](#)  
Change-Id [I00a5729a2fea7ca9e37c074716d521156e2991f7](#)

Dec 19, 2014 5:38 PM  
Dec 19, 2014 5:38 PM  
[\(gitweb\)](#)  
[\(gitweb\)](#)  
[\(gitweb\)](#)

Files
Open All
Diff against: Base

File Path	Comments	Size
Commit Message		
Android.mk	2	
	+1, -1	

History
Expand All

Feng Lin
Uploaded patch set 1.

Figure 12 commit view on git web

#### 4.1.8 Find Reviewer to help code review

## 5. Q&A

---

本節整理常常會遇到的問題，跟一些基本觀念

### 5.1. Q:Git 這麼多指令，有沒有跟 perforce 操作的對應表阿？

A: 目前沒有完整的對應表格，在這裡先簡單的對應兩邊的基本操作給大家熟悉一下。

Git	Perforce
Commit	Change List
Git push + merge to remote server	Submit
Git cherry-pick/ git merge + merge to remote server	Integrate
Git 開 branch 權限僅限管理員可以做，透過 gitweb 網頁來執行	Branch
Git log	看 history
Git diff	diff
Git 上沒有 lock code 的操作，但是可以調整權限來限制可以 push code 的人員名單。	Lock/unlock code

### 5.2. Q:為什麼在做 git 操作的時候，一直發生下圖的 error

```
[feng.lin@swhcsvr28 Git_learning]$ git clone ssh://feng.lin@hcggit02.mstarsemi.com.tw:28418/GitTesting/SW
Cloning into 'SW'...
Permission denied (publickey).
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
```

A: public key 設定錯誤，請查照本文件第二章設定 public key

### 5.3. Q:為什麼在 git push 的時候跳出找不到 Change-Id 的錯誤

```
[feng.lin@swhcsvr28 SW (test-dev)]$ git push origin HEAD:refs/for/test-dev
Counting objects: 7, done.
Delta compression using up to 32 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 367 bytes | 0 bytes/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2)
remote: Processing changes: refs: 1, done
remote: ERROR: missing Change-Id in commit message footer
remote: Suggestion for commit message:
remote: Test conflict
remote:
remote: Test conflict
remote: [red flag] = n
remote:
remote: Change-Id: I15f687ef5c45a017082f914533c8b71c1b79cd
remote:
remote: Hint: To automatically insert Change-Id, install the hook:
remote:  gitdir=$(git rev-parse --git-dir); scp -p -P 28418 feng.lin@hcg02.mstarsemi.com.tw:hooks/commit-msg ${gitdir}/hooks/
remote:
remote:
To ssh://feng.lin@hcg02.mstarsemi.com.tw:28418/GitTesting/SW
! [remote rejected] HEAD -> refs/for/test-dev (missing Change-Id in commit message footer)
error: failed to push some refs to 'ssh://feng.lin@hcg02.mstarsemi.com.tw:28418/GitTesting/SW'
[feng.lin@swhcsvr28 SW (test-dev)]$
```

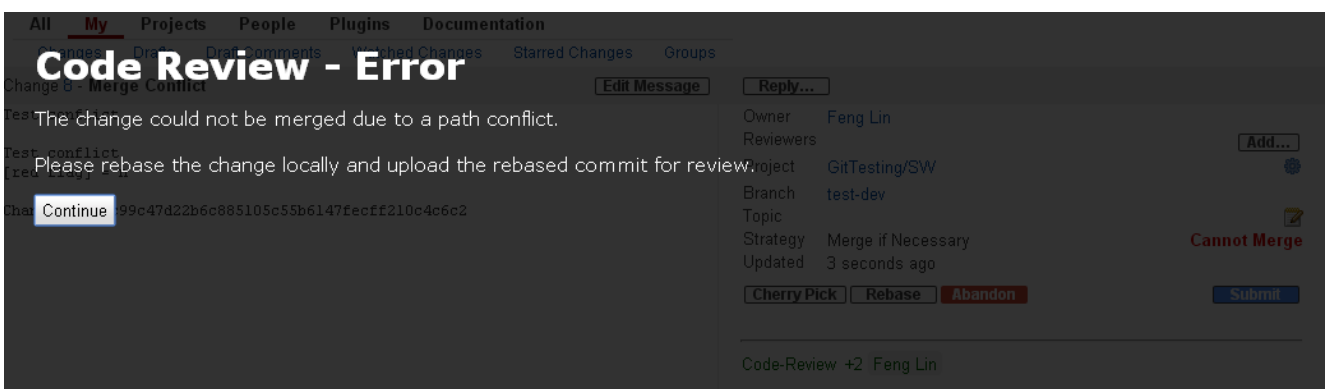
A: 兩種情況。

情況一：該 git server 目前沒有 repo 功能，所以沒有自動產生 commit-msg 的 soft link 到我們預設好的設定檔。

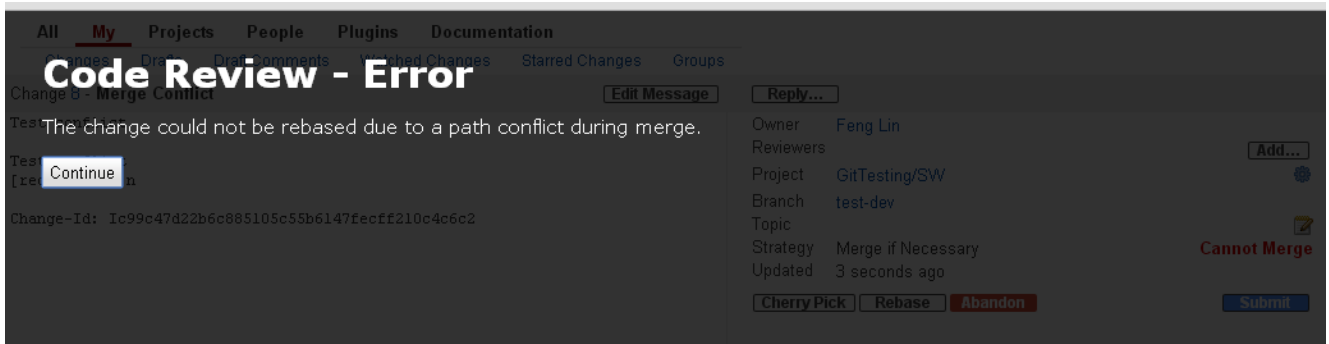
解決方式：按照圖中 hint 指示執行指令，可以完成產生的動作，再重新執行一次 git commit --amend 即可更新 commit 內容為包含 Change-Id 的樣板。

情況二：該 git server 的 repo 設定出問題了，請聯絡管理員幫忙處理。☺

### 5.4. Q:為什麼 module owner merge code 時，會遇到下圖的問題？



The screenshot shows a code review interface with a dark theme. The main area displays a message: "The change could not be merged due to a path conflict. Please rebase the change locally and upload the rebased commit for review." Below this message is a "Continue" button. To the right, there is a sidebar with details about the review: Owner (Feng Lin), Reviewers (empty), Project (GitTesting/SW), Branch (test-dev), Topic (empty), Strategy (Merge if Necessary), and Updated (3 seconds ago). At the bottom of the sidebar are buttons for "Cherry Pick", "Rebase", "Abandon", and "Submit". The top of the interface has tabs for "All", "My", "Projects", "People", "Plugins", and "Documentation".



A: 發生這種情況，表示你的 commit 與 remote server 的 code base 發生衝突了。  
如果有 rebase button，可以先嘗試按下 rebase 讓 git 自動 merge code。  
如果沒有 rebase button 或是 rebase 失敗，請參照以下連結做 resolve conflict 的動作。  
<http://hcggit/htdocs/git/How%20to%20resolve%20conflict.pdf>

## 5.5. Q: 什麼樣的情況下，我上的 code 會產生 conflict 呢？

A: 以下舉例說明

Case 1. 若大毛的 commit A 已 merge 成功，那麼在二毛 code 要被 merge 時，git 會先自動進行 merge，不幸無法完成 merge 時會產生 conflict，那麼二毛要進行 resolve conflict 的動作。

Case 2. 若大毛的 commit A 尚未 merge，那麼二毛 code 要被 merge 時，將會 merge 成功。  
然後角色對調，回到 Case 1. 的處理流程，大毛需要視情況決定是否需要進行 resolve conflict。

## 5.6. Q: 如果我覺得我的 code base 壞掉了，想要重抓一包 code，只能整個目錄砍掉重抓嗎？

A: 是不用砍掉目錄重抓的。

如果 server 上沒有 repo 指令，可以用以下步驟完成 reset code base。

```
git reset HEAD --hard; git clean -df; git rebase --abort;
git pull; git checkout origin/master;
git branch -D `git branch | grep -v \* | xargs`;
```

如果 server 上有 repo 指令，可以用以下步驟完成 reset code base。

```
repo forall -c 'git reset HEAD --hard; git clean -df; git rebase --abort;
repo sync -d -j32
repo forall -c 'git branch -D `git branch | grep -v \* | xargs`'
```

### 5.7. Q:如果想要很多個不同版本的 code base，只能 clone 很多份 code 嗎？

A: 不用的。

Git 的機制下，每一個 local branch 的 code 都是獨立的。

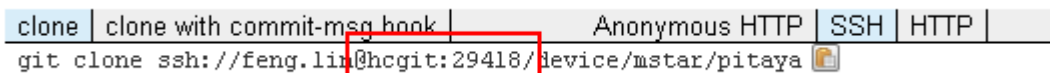
如果需要很多份 code base，只要新增一個 local branch，再把新的 branch 切到想要的版本即可。

### 5.8. Q:為什麼在執行 ssh 測試連線的時候，會有權限的問題？

A: 1. 確認 ssh key 是否正確填至 git web server 上。

2. 確認輸入的測試 port、server name 是否正確。

Ex: 如下圖，針對此 server，測試指令為：ssh -p 29418 hcgit



### 5.9. Q: 為什麼在 setting 頁面點選 ssh public key 或其他選項時，會出現 500 server error？

A: 目前還不知道 root cause，以下為建議方案。

1. 改用 chrome or firefox 瀏覽器

2. 如果 Chrome 無法連線的話先用 IE 登入

(F12 切到開發者工具 到工具列的文件模式改成 IE8 標準，即可登入)