

一、下載虛擬機

https://drive.google.com/file/d/1qIHqxywXU7Ozl_VSsh7ULAFbEu01LEb/view?usp=sharing

二、下載 VMWare Player

https://my.vmware.com/en/web/vmware/free#desktop_end_user_computing/vmware_workstation_player/15_0

三、登入 ZUVIO

1.請登入此網站註冊，使用"學號最後2碼+姓名"當成帳號

<https://irs.zuvio.com.tw/>

=====

剪取工具

https://docs.google.com/document/d/1fn4gyiVvKJJrzAS4EDlzNh2t-wmWvwx7T-aw966_DXU/edit

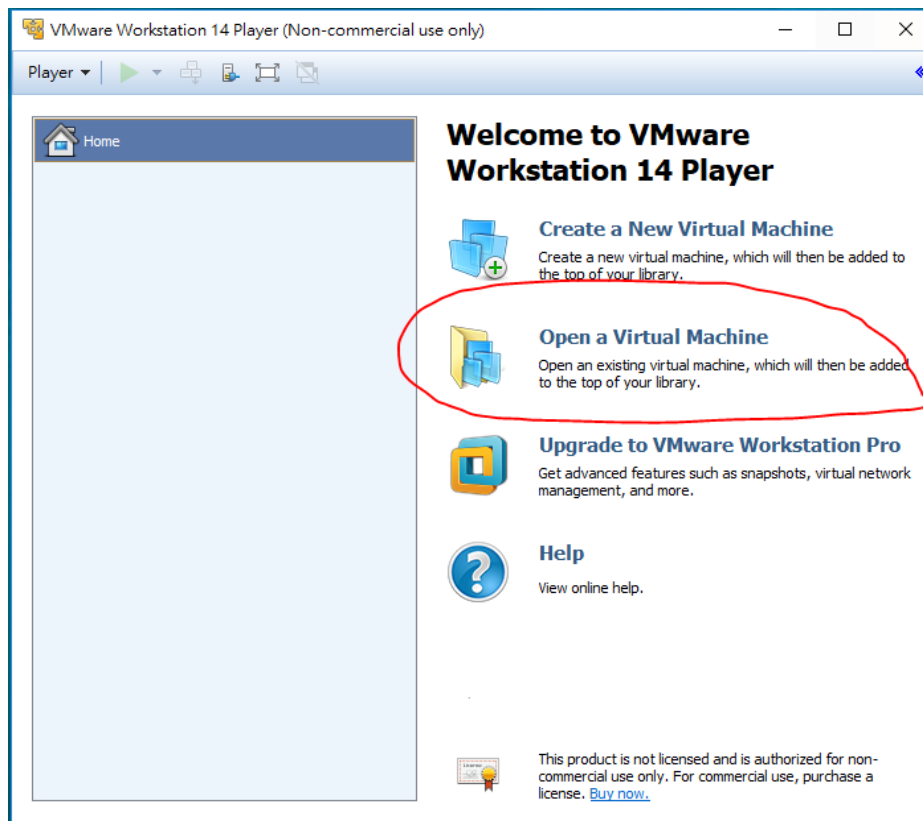
=====

四、準備執行環境

(https://myweb.ntut.edu.tw/~jykuo/train/Ubuntu14_23.pdf)

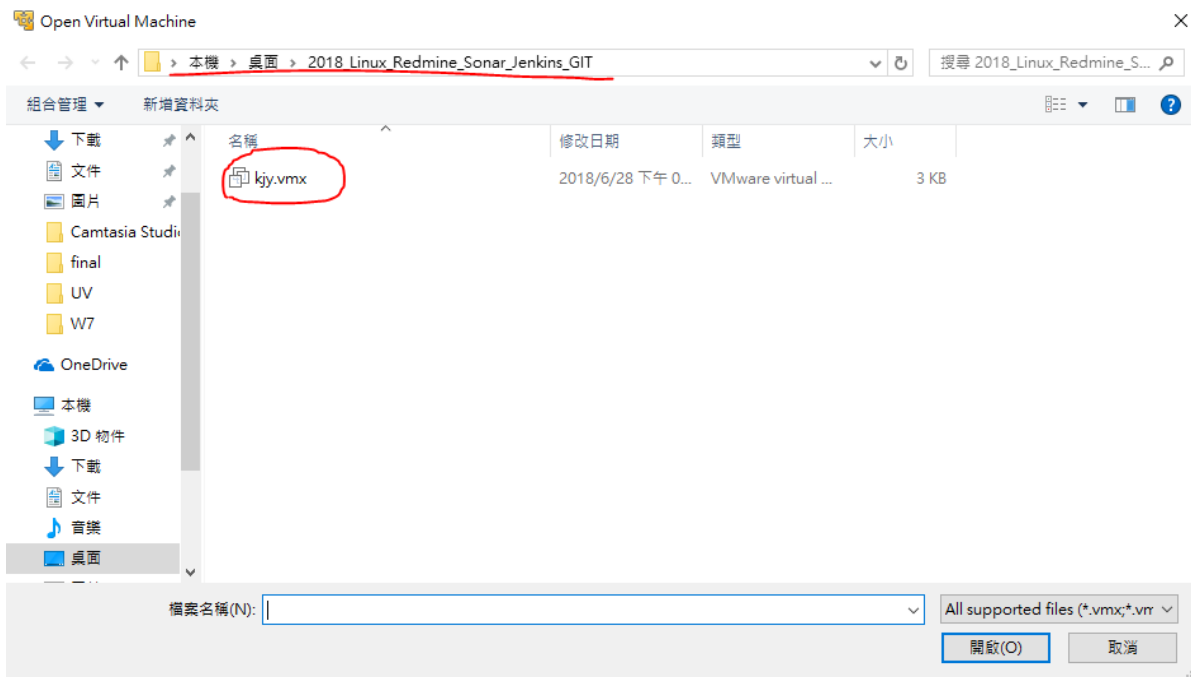
=====

1. 啟動 VM PLAYER

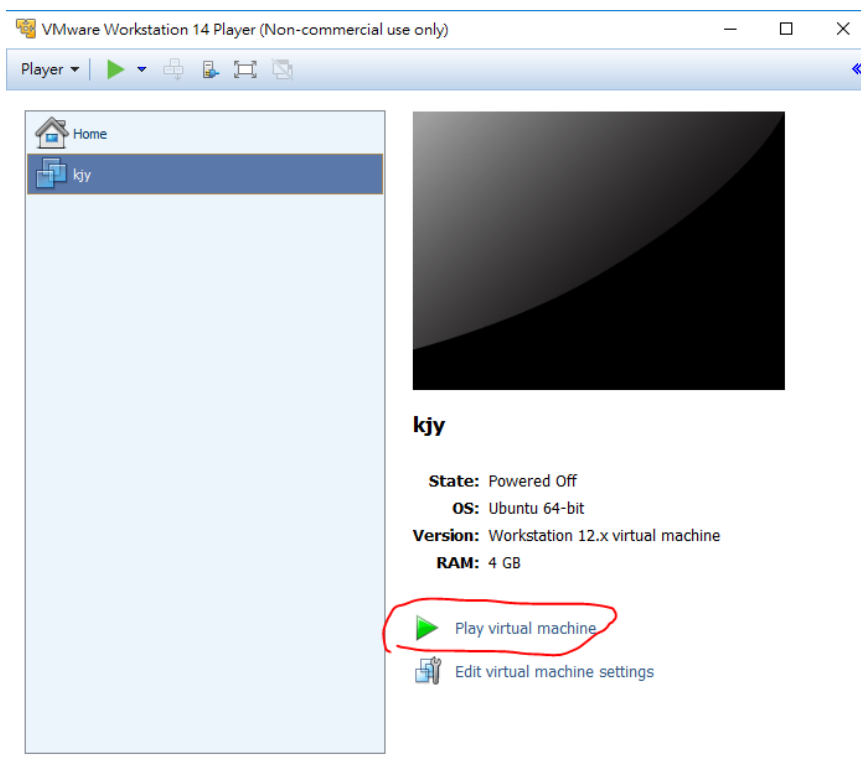


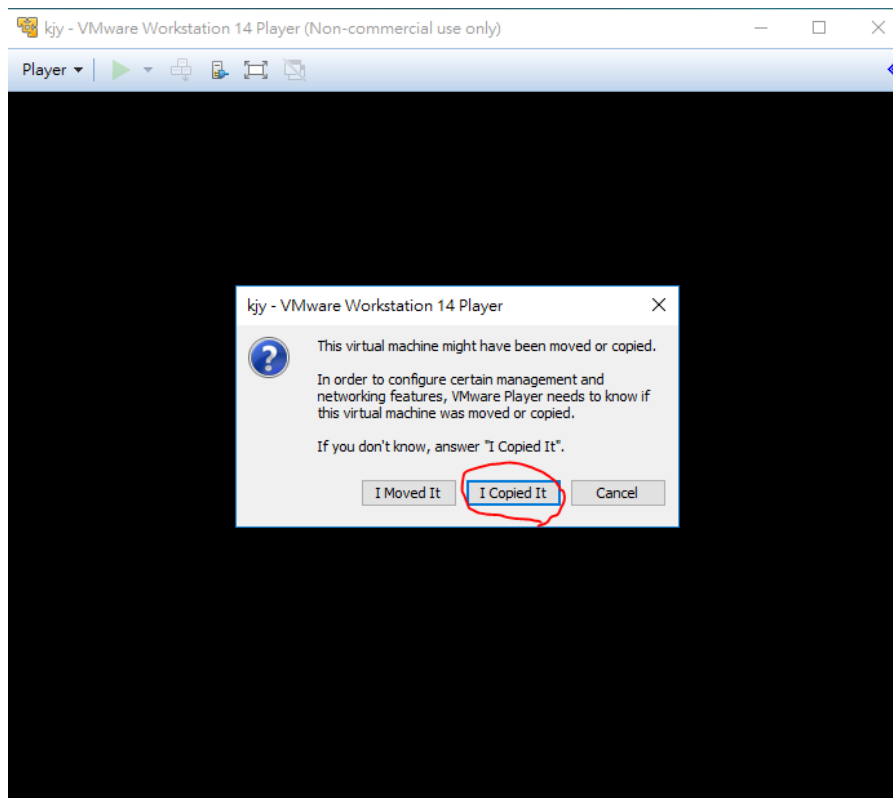
(b) 安裝虛擬機執行環境 VMWare Player 完成後，實施測試與問題排除
<https://myweb.ntut.edu.tw/~jykuo/train/VMNetworkTesting.pdf>

2. 開啟 VM 檔案

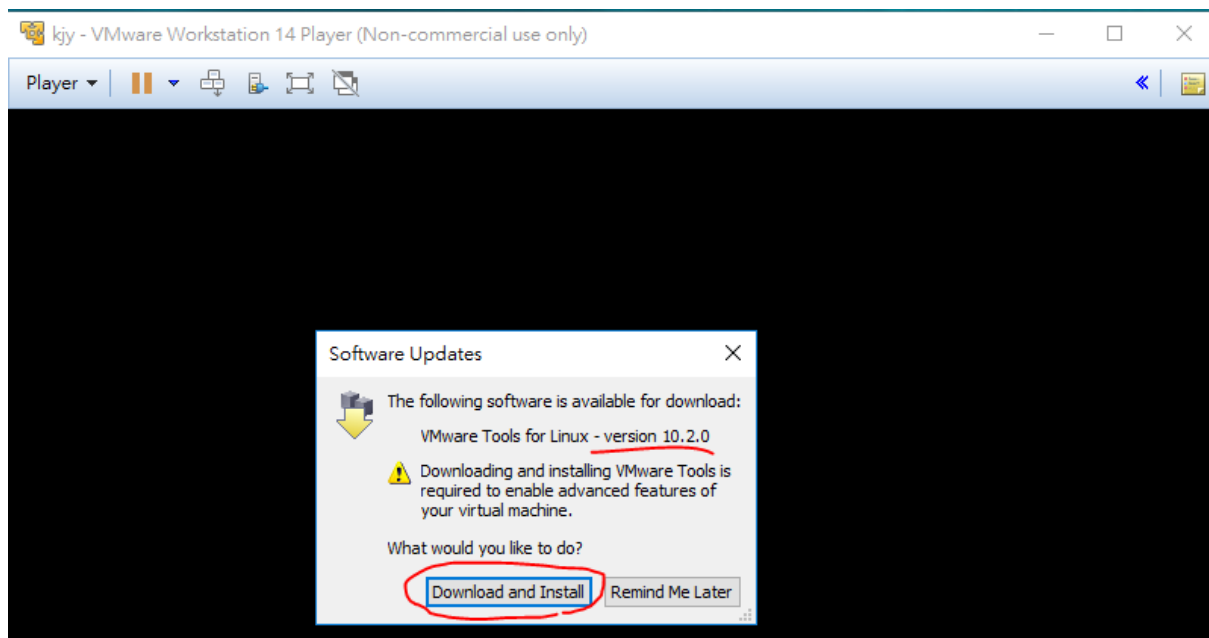


3. 播放VM 啟動 Linux,

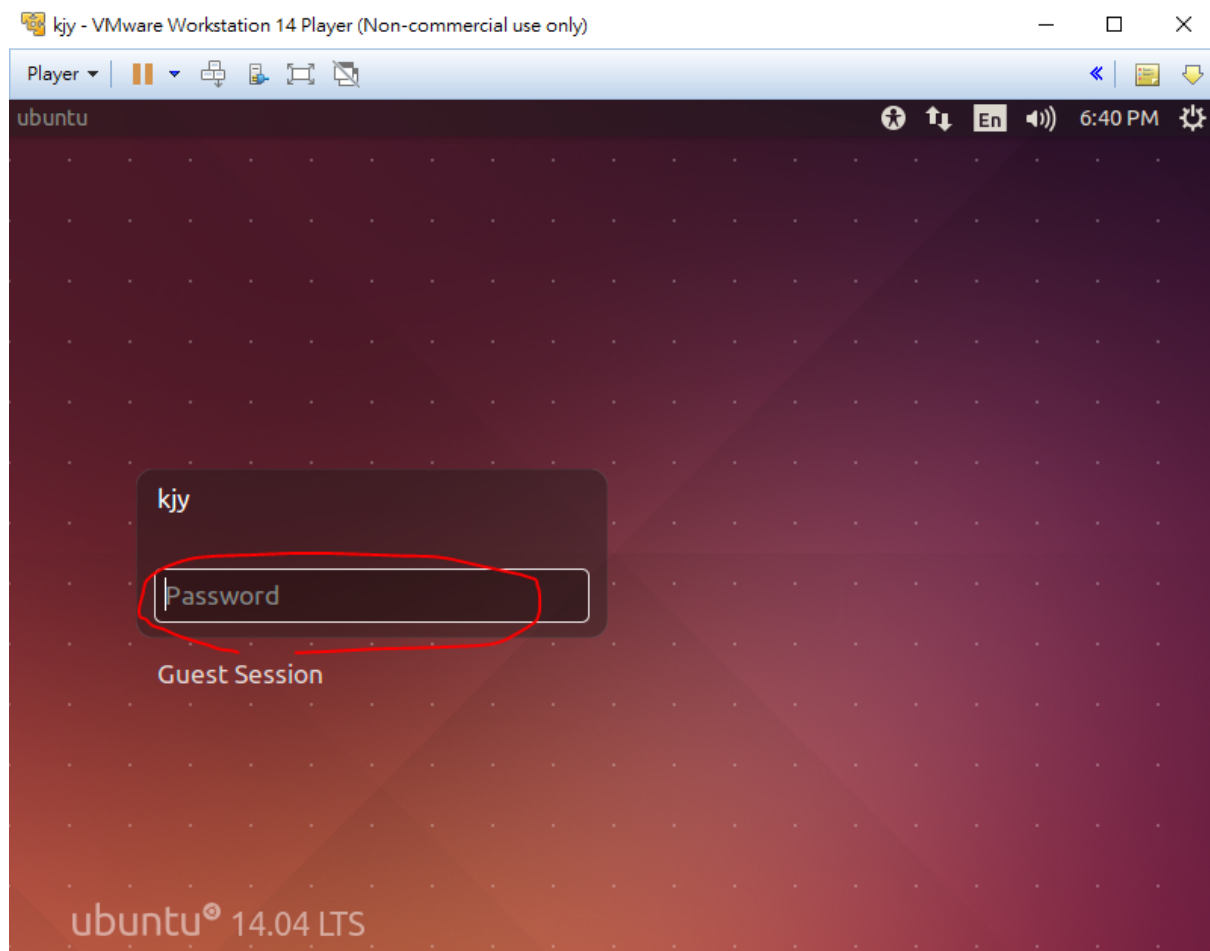




4. 更新VMWare Player 工具

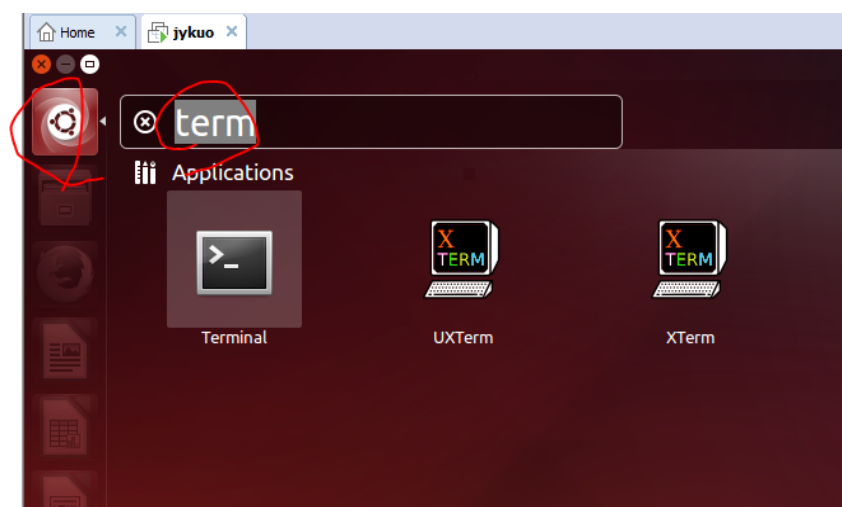


輸入帳號密碼 kjy/kjy



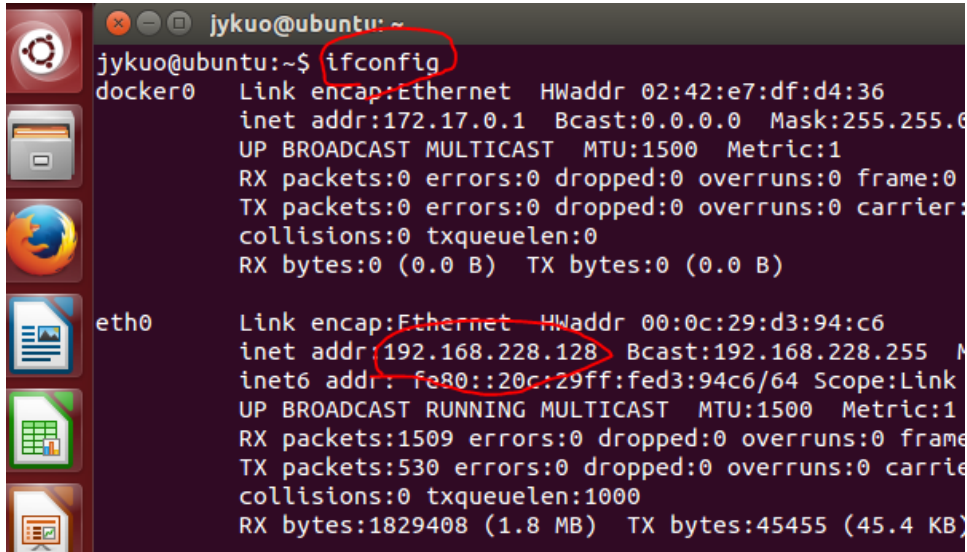
預設啟動 Linux 的 Jenkins Server, Git Server

5. 啟動 Linux Terminal



6. 指令 ifconfig,

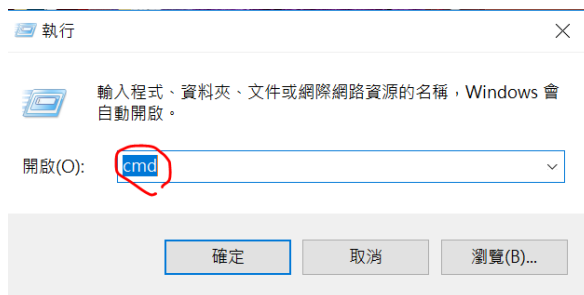
查看虛擬機 VM Linux 的內網 IP



```
jykuo@ubuntu:~$ ifconfig
docker0  Link encap:Ethernet  HWaddr 02:42:e7:df:d4:36
          inet addr:172.17.0.1  Bcast:0.0.0.0  Mask:255.255.0.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

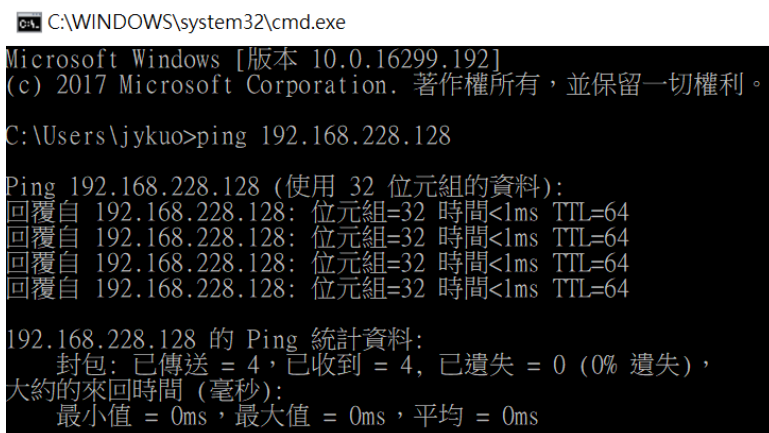
eth0      Link encap:Ethernet  HWaddr 00:0c:29:d3:94:c6
          inet addr:192.168.228.128  Bcast:192.168.228.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fed3:94c6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1509 errors:0 dropped:0 overruns:0 frame:0
          TX packets:530 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1829408 (1.8 MB)  TX bytes:45455 (45.4 KB)
```

本地端執行指令提示視窗



7. 測試虛擬機連線,

ping 192.168.xx.xx



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.16299.192]
(c) 2017 Microsoft Corporation. 著作權所有，並保留一切權利。
C:\Users\jykuo>ping 192.168.228.128

Ping 192.168.228.128 (使用 32 位元組的資料):
回覆自 192.168.228.128: 位元組=32 時間<1ms TTL=64
回覆自 192.168.228.128: 位元組=32 時間<1ms TTL=64
回覆自 192.168.228.128: 位元組=32 時間<1ms TTL=64
回覆自 192.168.228.128: 位元組=32 時間<1ms TTL=64

192.168.228.128 的 Ping 統計資料:
    封包: 已傳送 = 4, 已收到 = 4, 已遺失 = 0 (0% 遺失),
    大約的來回時間 (毫秒):
        最小值 = 0ms, 最大值 = 0ms, 平均 = 0ms
```

8. 使用瀏覽器啟動 jenkins,

192.168.xx.xx:8080, kjy/kjy



講義, https://myweb.ntut.edu.tw/~jykuo/train/001BuildToolLab_2017_33.pdf

9. 啟動 redmine,

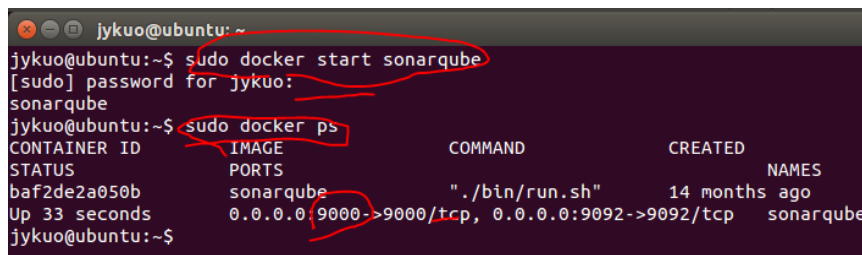
<http://192.168.xx.xx/redmine> , admin/admin



10. 啟動 sonarqube

sudo docker start sonarqube

kjy



在瀏覽器啟動sonarqube, <http://192.168.xx.xx:9000>, admin/admin

參考資料, https://myweb.ntut.edu.tw/~jykuo/train/Ubuntu14_23.pdf

五、Jenkins練習 (project leder或資深工程師, 持續整合建置)

1. 三層選單 Jenkin

-專案 (test01)-第幾次建置



2. Jenkins - 建置歷程

This screenshot shows the Jenkins 'Build History' page. On the left is a sidebar with navigation links: '新增作業' (Add job), '使用者' (Users), '建置歷程' (Build history - circled in red), '專案關連' (Project relationships), '檢查檔案指紋' (Check file fingerprints), '管理 Jenkins' (Manage Jenkins), '我的視景' (My views), 'Credentials', and 'New View'. On the right, there is a table of build history. Above the table are tabs for '全部' (All) and '+'. The table has columns for status (S), weather icon (W), name, and last success. One build is listed with a blue status icon, a sun weather icon, the name 'Test01', and a last success time of '3月2'. Below the table is a link '圖示: S M L'. At the bottom, there are two expandable sections: '建置佇列' (Build queue) which is currently empty, and '建置執行程式狀態' (Build executor status) which shows two executors in a '閒置' (Idle) state.

S	W	名稱 ↓	上次成功
		Test01	3月2

圖示: [S](#) [M](#) [L](#)

建置佇列
佇列中沒有建置作業。

建置執行程式狀態

- 1 閒置
- 2 閒置

3. 第二層選單 Test01

- 馬上建置 - (藍色建置成功, 紅色建置失敗)

Jenkins

Jenkins > Test01 >

回到儀表板

狀態

變更

工作日誌

馬上建置

刪除專案

組態

SonarQube

Rename

Coverage Trend

建置歷程

趨勢

find x

#12

2018/3/7 下午 9:07

專案 Test01

SonarQube

工作區

最近變更

永久連結

- 最新建置 (#12), 3 月 23 天 以前
- 最新穩定建置 (#12), 3 月 23 天 以前
- 最新成功建置 (#12), 3 月 23 天 以前
- [Last completed build \(#12\), 3 月 23 天](#)

4. 點選 Jenkins - Test01- #13

The screenshot shows the Jenkins web interface for a project named 'Test01'. The top navigation bar includes a 'Jenkins' dropdown and a 'Test01' breadcrumb. On the left sidebar, there are links for '回到儀表板', '狀態', '變更', '工作目錄', '馬上建置', '刪除專案', '組態', 'SonarQube', 'Rename', and 'Coverage Trend'. The main content area is titled '專案 Test01' and contains links for 'SonarQube', '工作', and '最近'. Below this is a section titled '永久連結' with a list of links: '最新建置 (#13)', '最新穩定建置', '最新成功建置', and 'Last complete'. The '建置歷程' (Build History) table is visible, with a search bar containing 'find'. The table lists four builds: #13 (2018/6/29 下午 7:53, failed), #12 (2018/3/7 下午 9:07, failed), #11 (2018/3/7 下午 8:55, failed), and #1 (2018/3/7 下午 2:42, failed). Build #13 is circled in red. At the bottom of the table, there are RSS links for 'RSS 摘要: 全部' and 'RSS 摘要: 失敗'.

Build Number	Timestamp	Status
#13	2018/6/29 下午 7:53	Failed
#12	2018/3/7 下午 9:07	Failed
#11	2018/3/7 下午 8:55	Failed
#1	2018/3/7 下午 2:42	Failed

5. 點選第三層選單， 查看建置錯誤訊息

Jenkins

2 搜尋

Jenkins > Test01 > #13

回到專案
狀態
變更
Console Output
View as plain text
編輯建構資訊
刪除這次建構
Git Build Data
No Tags
前一次建構

終端機輸出

```

Started by user kjy
Building in workspace /var/lib/jenkins/workspace/Test01
> git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url /usr/share/gitrepo/proj02 # timeout=10
Fetching upstream changes from /usr/share/gitrepo/proj02
> git --version # timeout=10
Using GIT_ASKPASS to set credentials
> git fetch --tags --progress /usr/share/gitrepo/proj02 +refs/heads/*:refs/re
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision c8eb8a54ccfa8c03c821fb51a889d2e2246f9e5f (refs/remotes/c
> git config core.sparsecheckout # timeout=10
> git checkout -f c8eb8a54ccfa8c03c821fb51a889d2e2246f9e5f
Commit message: "PUSH REMOTE"
> git rev-list --no-walk c8eb8a54ccfa8c03c821fb51a889d2e2246f9e5f # timeout=1
[Test01] $ /bin/sh -xe /tmp/jenkins3021649770148715533.sh
+ gradle build
Starting a Gradle Daemon (subsequent builds will be faster)
:compileJava UP-TO-DATE
:processResources NO-SOURCE
:classes UP-TO-DATE
:jar UP-TO-DATE
:startScripts UP-TO-DATE
:distTar UP-TO-DATE
:distZip UP-TO-DATE
:assemble UP-TO-DATE
:compileTestJava UP-TO-DATE
:processTestResources NO-SOURCE
:testClasses UP-TO-DATE
:test UP-TO-DATE
:check UP-TO-DATE
:build UP-TO-DATE

BUILD SUCCESSFUL in 7s
7 actionable tasks: 7 up-to-date
[Test01] $ /var/lib/jenkins/tools/hudson.plugins.sonar.SonarRunnerInstallation
scanner -e -Dsonar.host.url=http://192.168.10.130:9000/ "-Dsonar.projectName=$
-Dsonar.projectVersion=1.0 -Dsonar.projectKey=moe-st -Dsonar.java.binaries=bui

```

```

INFO: SonarQube Scanner 2.6.1
INFO: Java 1.8.0_171 Oracle Corporation (64-bit)
INFO: Linux 4.2.0-27-generic amd64
INFO: Error stacktraces are turned on.
INFO: User cache: /var/lib/jenkins/.sonar/cache
ERROR: SonarQube server [http://192.168.10.130:9000] can not be reached
INFO: -----
INFO: EXECUTION FAILURE
INFO: -----
INFO: Total time: 5.729s
INFO: Final Memory: 3M/59M
INFO: -----
ERROR: Error during SonarQube Scanner execution
org.sonarsource.scanner.api.internal.ScannerException: Unable to execute SonarQube

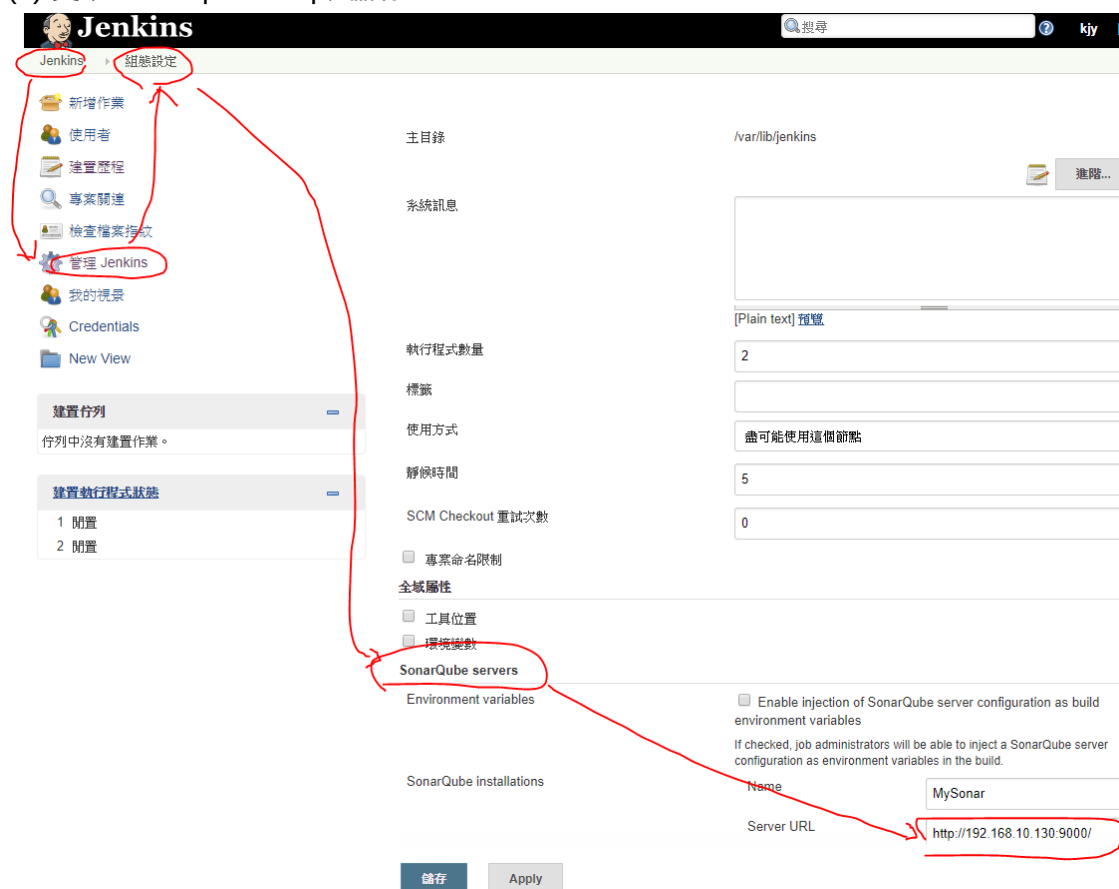
```

6. 修正sonarqube 設定 IP 錯誤

- (1) 到Jenkins 瀏覽器控制畫面, 設定 sonarqube IP, (安裝sonarqube plugin)
選單, Jenkins - 管理 Jenkins - 設定系統 - sonarqube serve URL - ip



- (2) 更改 sonarqube 的 ip, 儲存



設定好, 再到專案 Test01 - 馬上建置 (藍色 - 建置成功)

Jenkins

Jenkins Test01

回到儀表板
狀態
變更
工作目錄
馬上建置
刪除專案
組態
SonarQube
Rename
Coverage Trend

專案 Test01

SonarQube
工作區
最近變更

永久連結

- 最新建置 (#13), 8分6秒以前
- 最新穩定建置 (#12), 3月23天以前
- 最新成功建置 (#12), 3月23天以前
- 最新失敗建置 (#13), 8分6秒以前
- 最新不成功建置 (#13), 8分6秒以前
- Last completed build (#13), 8分6秒以前

建置歷程

Build	Time	Status
#14	2018/6/29 下午 8:01	Success
#13	2018/6/29 下午 7:53	Failure
#12	2018/3/7 下午 9:07	Success
#11	2018/3/7 下午 8:55	Success
#1	2018/3/7 下午 2:42	Success

RSS 摘要: 全部 RSS 摘要: 失敗

7. 觀察建置訊息

Jenkins 第三層選單, 建置號碼, 觀察建置訊息

Jenkins > Test01 > #15

回到專案
狀態
變更
Console Output
View as plain text
編輯建構資訊
刪除這次建構
Git Build Data
No Tags
前一次建構

終端機輸出

```
Started by user jykuo
Building in workspace /var/lib/jenkins/work
> git rev-parse --is-inside-work-tree # ti
Fetching changes from the remote Git reposi
> git config remote.origin.url /usr/share/
Fetching upstream changes from /usr/share/g
> git --version # timeout=10
using GIT_ASKPASS to set credentials
> git fetch --tags --progress /usr/share/g
> git rev-parse refs/remotes/origin/master
> git rev-parse refs/remotes/origin/origir
Checking out Revision ef0ae80c729e10a7bde8t
> git config core.sparsecheckout # timeout
> git checkout -f ef0ae80c729e10a7bde8b2c6
> git rev-list ef0ae80c729e10a7bde8b2c081f
[Test01] $ /bin/sh -xe /tmp/hudson649740934
+ gradle build
```

8. 查看 Coverage

查看 JoCoCo 測試 Coverage 是否足夠

Jenkins 2 搜尋 kji | 登入

Jenkins > Test01 > 開啟自動更新頁

回到儀表板
狀態
變更
工作目錄
馬上建置
刪除專案
組態
SonarQube
Rename
Coverage Trend

專案 Test01

SonarQube
工作區
最近變更

永久連結

- 最新建置 (#14), 4 分 55 秒以前
- 最新穩定建置 (#14), 4 分 55 秒以前
- 最新成功建置 (#14), 4 分 55 秒以前
- 最新失敗建置 (#13), 11 分 以前
- 最新不成功建置 (#13), 11 分 以前
- Last completed build (#14), 4 分 55 秒 以前

Code Coverage Trend

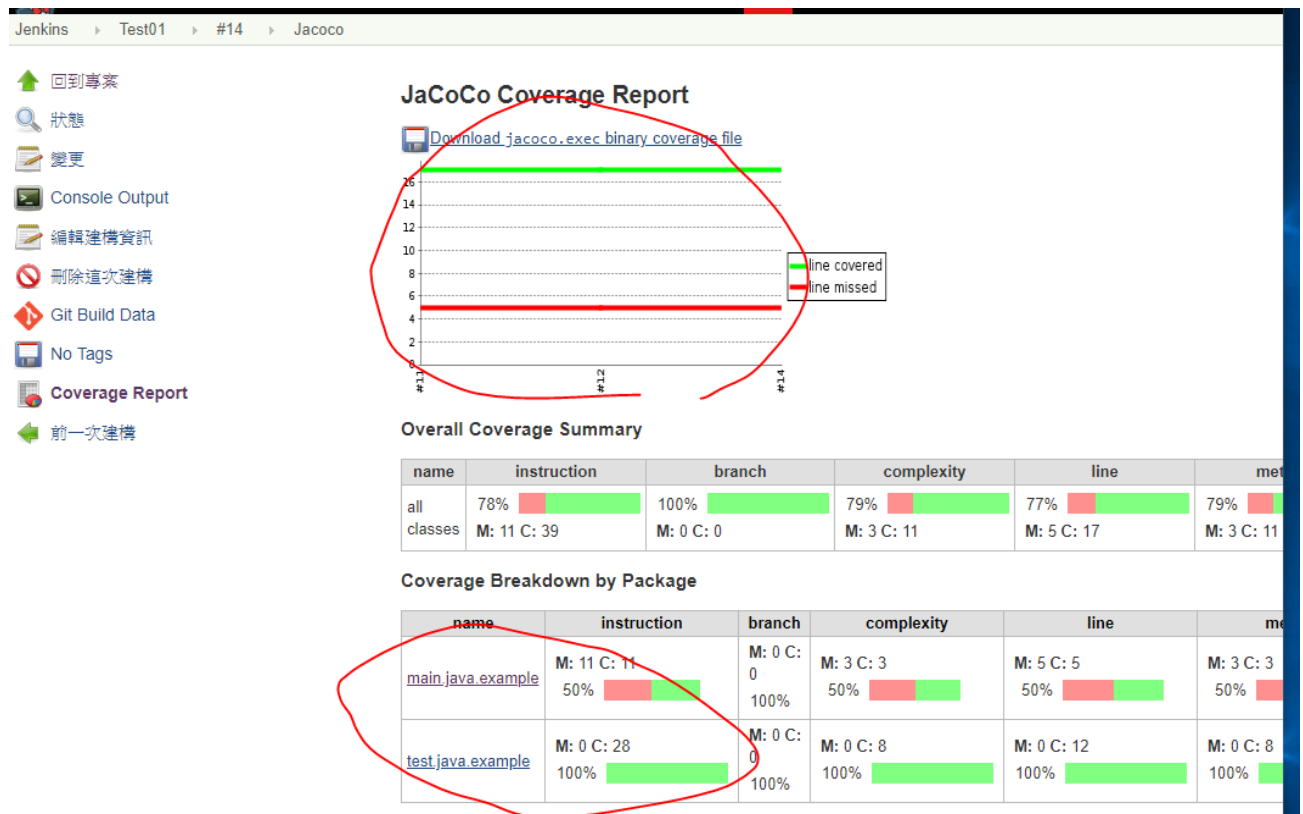
新增
停用專案

enle

建置歷程

趨勢

Build	Time	Status
#14	2018/5/22 下午 6:58	Success
#13	2018/5/22 下午 6:51	Failure
#12	2018/3/7 下午 9:07	Success



9. 改善測試 Coverage

專案 proj02 沒有充分測試，需要改善測試的 Coverage

- (1) 前提是 - 必須建置成功
- (2) JaCoCo - Java Code Coverage
statement code coverage - 每一行程式都有被測試 100%
- (3) Gradle 建置規定
main package - 交付客戶的程式
test package - 測試程式
- (4) Tiger 未被充分測試

10. 定時建置

Daily Build 每日建置 ()

- (1) 第二層選單 Test01- 組態

Jenkins > Test01

回到儀表板
狀態
變更
工作目錄
馬上建置
刪除專案
組態
Coverage Trend
Move
SonarQube

專案 Test01

My Test

SonarQube
工作區
最近變更

永久連結

• 最新建置 (#17) 30 分 以前

Jenkins > Test01

General 原始碼管理 建置觸發程序 建置環境 建置 建置後動作

專案 名稱 Test01

描述 My Test

Software Configuration Management

(2) 設定建置觸發程序 - 定時、固定觸發規則, 持續建置系統

Jenkins > Test01

General 原始碼管理 建置觸發程序 建置環境 建置 建置後動作

專案 名稱 Test01

描述 My Test

[Plain text] 預覽

☐ GitHub project
☐ Throttle builds
☐ 參數化建置
☐ 忽略舊Builds
☐ 停用Build
☐ 必要時同時執行多個建構

原始碼管理

☐ 無
☒ Git

(3) 各種不同建置排程



(4) 定期建置



11. 外掛程式

Jenkins - 管理 Jenkins - 管理外掛程式

Jenkins

- 新增作業
- 使用者
- 建置歷程
- 管理 Jenkins
- 我的視景
- Credentials

建置佇列

佇列中沒有建置作業。

建置執行程式狀態

- 1 閒置
- 2 閒置

管理 Jenkins

⚠ 新版的 Jenkins (2.90) 已經可以下載

- 設定系統
設定全域設定及路徑。
- 設定全域安全性
保護 Jenkins，定義誰可以
- Configure Credentials
Configure the credential
- Global Tool Configuration
Configure tools, their loca
- 從磁碟重新載入設定
放棄所有在記憶體裡的資料
- 管理外掛程式
外掛程式是能擴充 Jenkins
- 系統資訊

(1) 更新 - 可用的 - 已安裝 - 進階

Jenkins 外掛程式管理

回到儀表板

管理 Jenkins

更新 可用的 已安裝 進階

安裝

☐ [Ant Plugin](#)
Adds Apache Ant support to J

(2) 搜尋可用的 plug-in

Jenkins 外掛程式管理

回到儀表板

管理 Jenkins

過濾條件: java

更新 可用的 已安裝 進階

安裝

名稱
FindBugs Plug-in This plug-in generates the trend report for FindBugs, an open source program which uses static analysis to look for bugs in Java code. Warning: This plugin requires dependent plugins be upgraded and at least one of these dependent plugins claims to use a different setting the installed version. Jobs using that plugin may need to be reconfigured, and/or you may not be able to cleanly revert to the prior version manually restoring old settings. Consult the plugin release notes for details.
Javascript Widgets Plugin Allows embedding various statistics available from Jenkins in your via javascript-snippets ala ohloh.net

已安裝

Jenkins > 外掛程式管理

☒	Pipeline: Stage View Plugin Pipeline Stage View Plugin.	2.0
☒	Pipeline: Step API API for asynchronous build step primitive.	2.3
☒	Pipeline: Supporting APIs Common utility implementations to build Pipeline Plugin	2.5
☒	Plain Credentials Plugin Allows use of plain strings and files as credentials.	1.2
☒	SCM API Plugin This plugin provides a new enhanced API for interacting with SCM systems.	1.3
☒	Script Security Plugin Allows Jenkins administrators to control what in-process scripts can be run by less-privileged users.	1.22
☒	SonarQube Scanner for Jenkins This plugin allows an easy integration of SonarQube , the open source platform for Continuous Inspection of code quality.	2.4.4
☒	SSH Credentials Plugin Allows storage of SSH credentials in Jenkins	1.12

(3) 安裝相對應的 plug-in

Jenkins

外掛程式管理

↑ 回到儀表板

⚙ 管理 Jenkins

更新

可用的

已安裝

進階

已啟用

☒

[Ant Plugin](#)

Adds Apache Ant support to Jen

☒

[bouncycastle API Plugin](#)

This plugin provides an stable AI

☒

[Branch API Plugin](#)

This plugin provides an API for n

☒

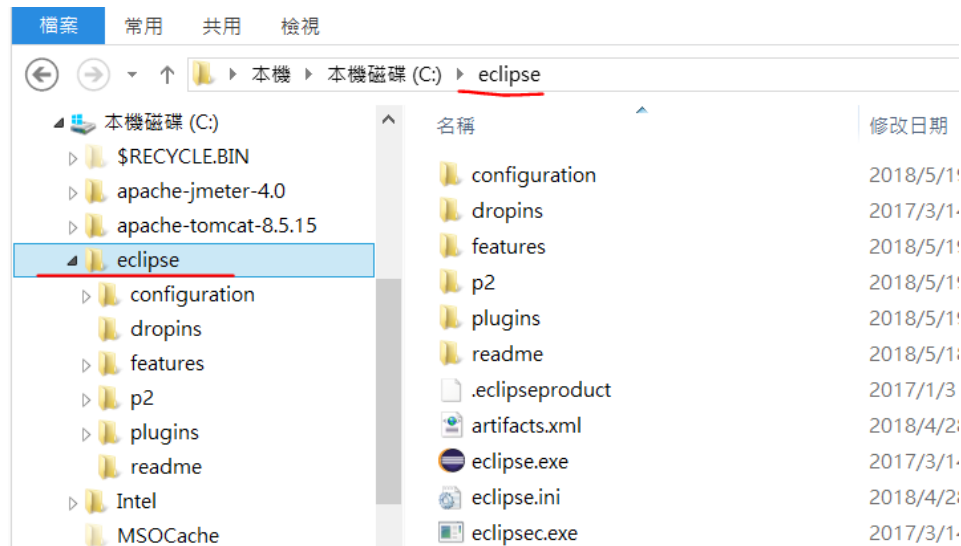
[Build Timeout](#)

六、CLIENT端操作

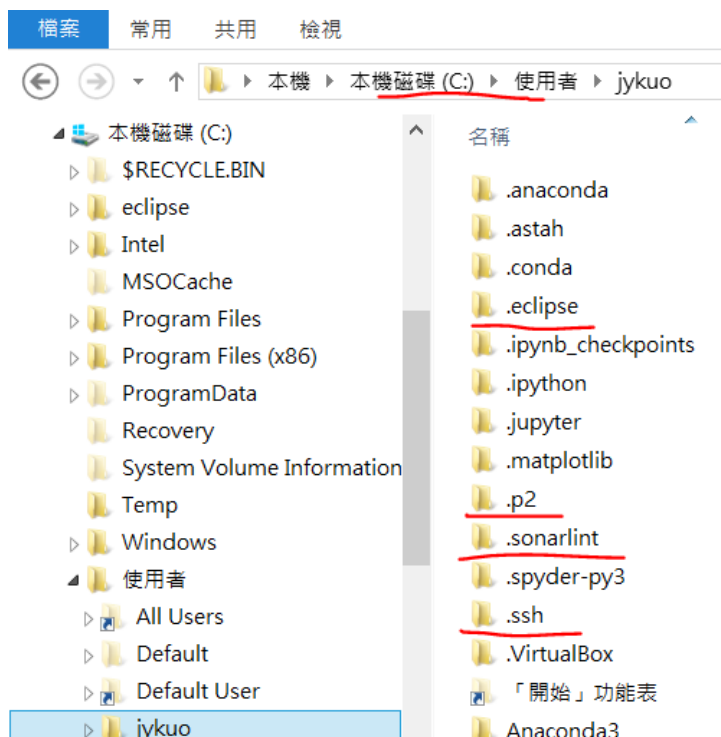
1. 下載 Eclipse

下載完解壓縮到 C:\

https://www.dropbox.com/s/esec3znqovdml05/eclipse-jee-neon-3-win32-x86_64.zip?dl=0



2. 若有舊版本, 清除相關舊檔案



3. 從遠端 Git repository 複製 clone 專案檔案

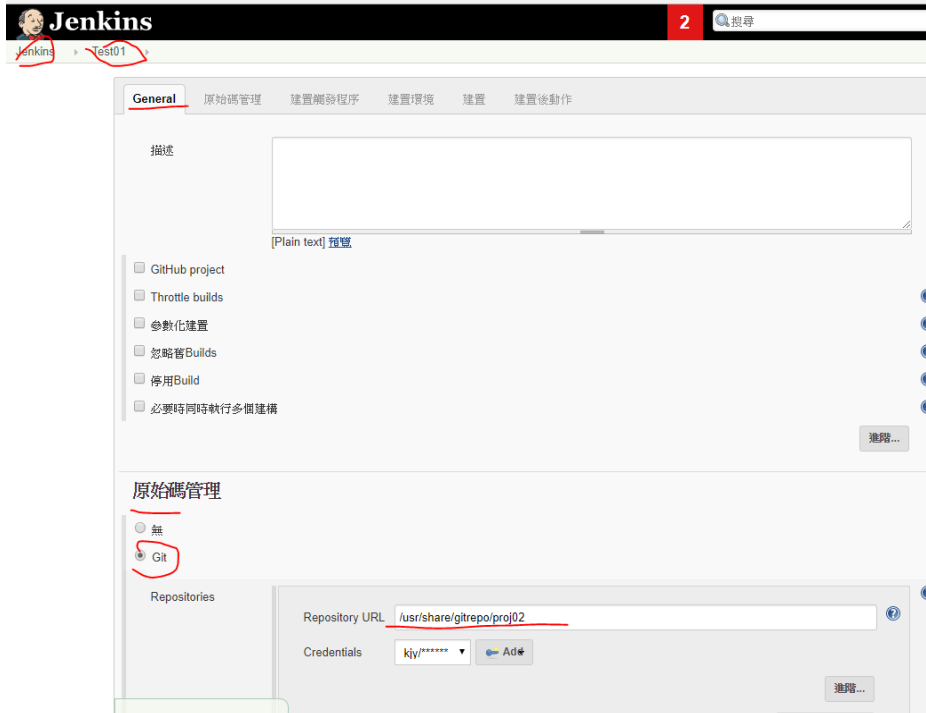
參考講義 https://myweb.ntut.edu.tw/~jykuo/train/EGit_Ubuntu_2017.pdf

遠端 Git 路徑



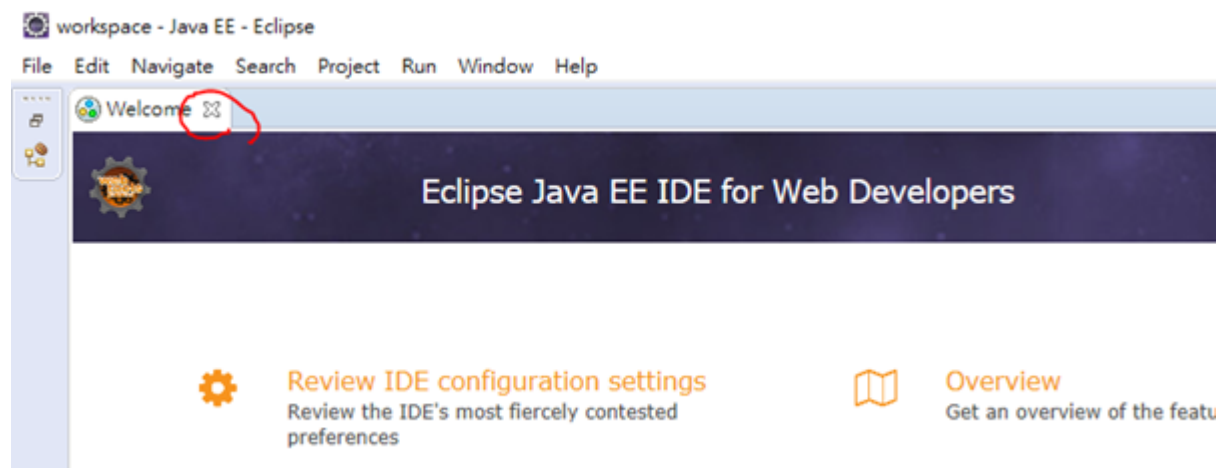
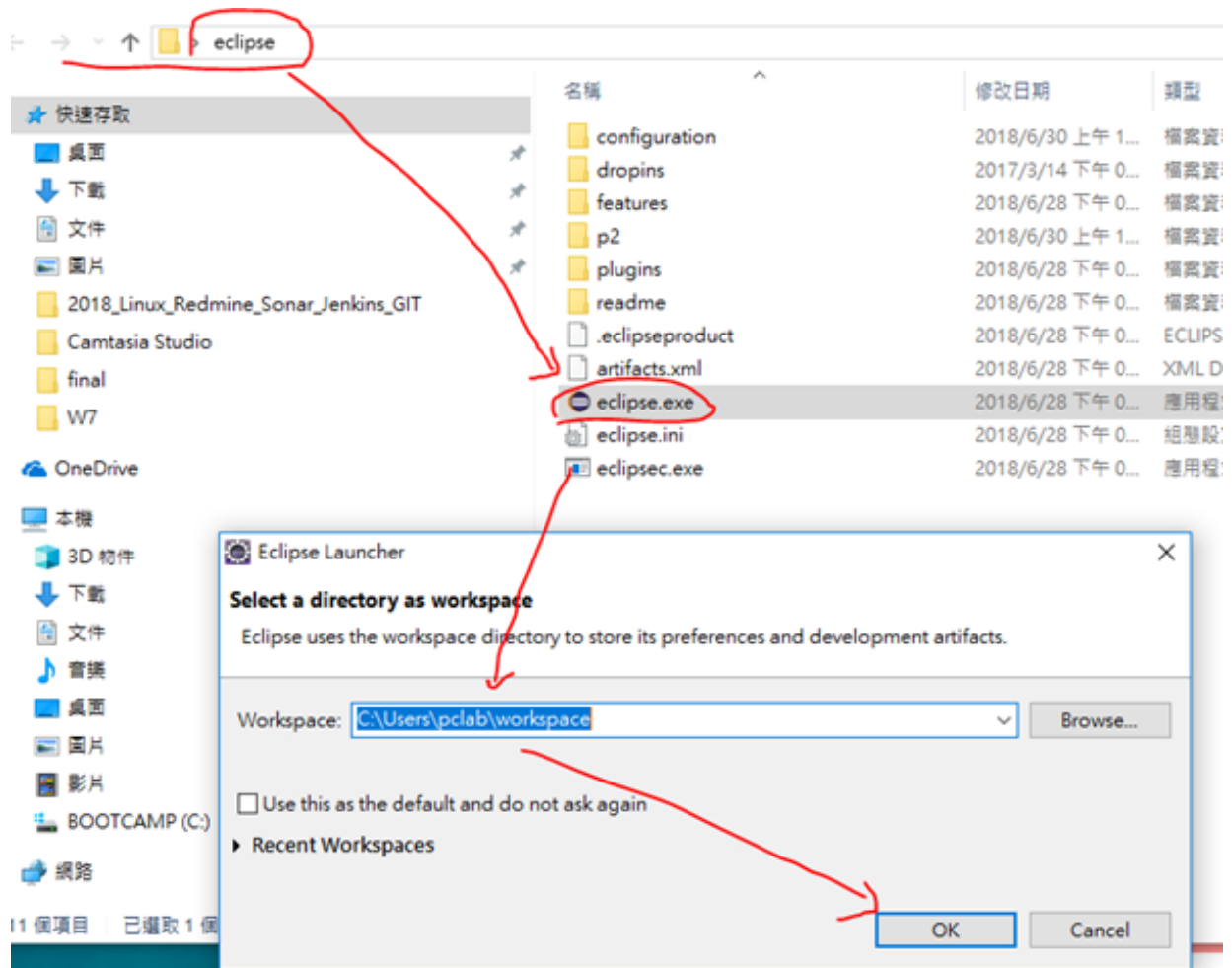
The screenshot shows the Jenkins Test01 dashboard. On the left, a sidebar contains navigation links: 回到儀表板, 狀態, 變更, 工作目錄, 馬上建置, 刪除專案, 組態 (highlighted with a red circle), SonarQube, Rename, and Coverage Trend. The main area is titled '專案 Test01' and features a SonarQube status bar on the right with a green progress line. Below the title, there are links for '工作區' and '最近變更'. A '永久連結' section lists several build links, including '最新建置 (#14), 38 分 以前' and 'Last completed build (#14), 38 分 以前'. At the bottom left, a '建置歷程' table lists builds #11 through #14 with their respective timestamps and status icons.

Build #	Timestamp	Status
#14	2018/5/22 下午 6:58	Success
#13	2018/5/22 下午 6:51	Failure
#12	2018/3/7 下午 9:07	Success
#11	2018/3/7 下午 8:55	Success



The screenshot shows the Jenkins Test01 configuration page. The 'General' tab is selected, displaying a description field and a list of checkboxes for build options: GitHub project, Throttle builds, 參數化建置, 忽略舊Builds, 停用Build, and 必要時同時執行多個建構. Below this, the '原始碼管理' (Source Code Management) section is active, showing 'Git' as the selected provider. The 'Repositories' list contains one entry with the 'Repository URL' set to '/usr/share/gitrepo/proj02' and 'Credentials' set to 'kiv*****'. The '進階...' (Advanced) button is visible at the bottom right.

4. 啟動 Eclipse, 設定空 workspace



5. 找到 Git Server 的目錄位置

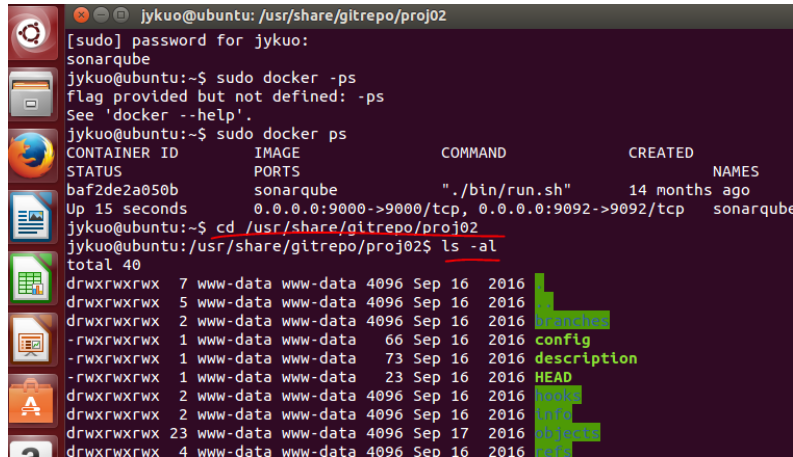
(1) VM Linux Terminal (找 Git 目錄結構)

chang directory

cd /usr/share/gitrepo/proj02

list 檔案目錄詳細資料清單

ls -al

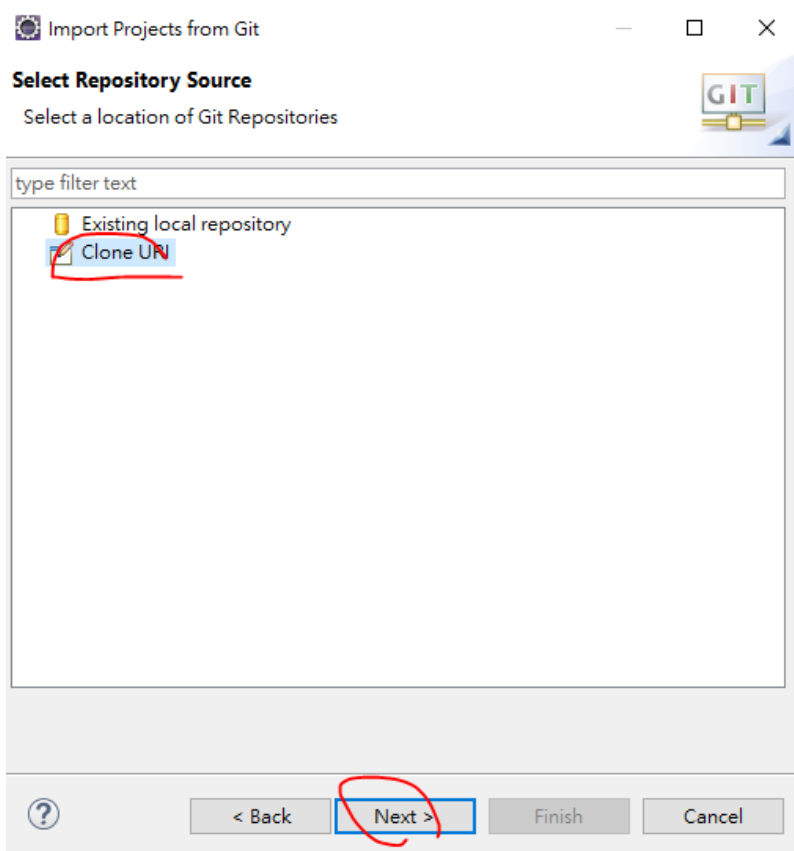
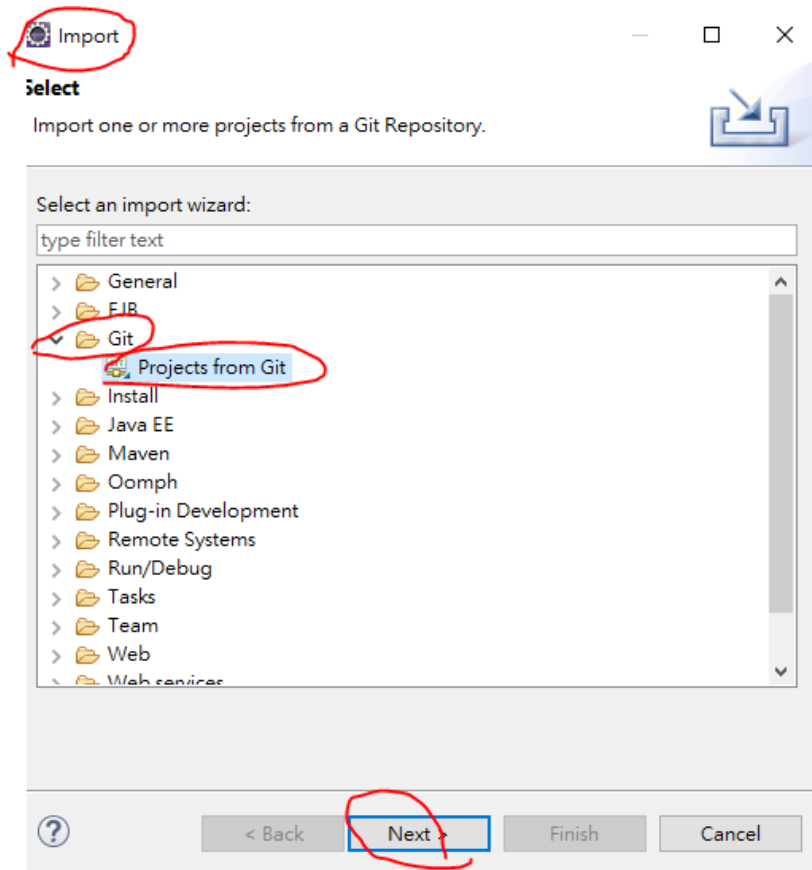


```
jykuo@ubuntu: /usr/share/gitrepo/proj02
[sudo] password for jykuo:
sonarqube
jykuo@ubuntu:~$ sudo docker -ps
flag provided but not defined: -ps
See 'docker --help'.
jykuo@ubuntu:~$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             NAMES
baf2de2a050b        sonarqube           "/bin/run.sh"       14 months ago       sonarqube
Up 15 seconds
0.0.0.0:9000->9000/tcp, 0.0.0.0:9092->9092/tcp
jykuo@ubuntu:~$ cd /usr/share/gitrepo/proj02
jykuo@ubuntu: /usr/share/gitrepo/proj02$ ls -al
total 40
drwxrwxrwx 7 www-data www-data 4096 Sep 16 2016
drwxrwxrwx 5 www-data www-data 4096 Sep 16 2016
drwxrwxrwx 2 www-data www-data 4096 Sep 16 2016
-rwxrwxrwx 1 www-data www-data 66 Sep 16 2016
-rwxrwxrwx 1 www-data www-data 73 Sep 16 2016
-rwxrwxrwx 1 www-data www-data 23 Sep 16 2016
drwxrwxrwx 2 www-data www-data 4096 Sep 16 2016
drwxrwxrwx 2 www-data www-data 4096 Sep 16 2016
drwxrwxrwx 23 www-data www-data 4096 Sep 17 2016
drwxrwxrwx 4 www-data www-data 4096 Sep 16 2016
```

參考資料 - http://www.cc.ntut.edu.tw/~jykuo/train/Ubuntu14_23.pdf

6. 本地端電腦 ECLIPSE, 從 Git server 抓下 proj02專案檔案

Eclipse - File - Import



Import Projects from Git

Source Git Repository

Enter the location of the source repository.

Location

URI:

✓ Host:

✓ Repository path:

Connection

✓ Protocol:

Port:

Authentication

User: ✓

Password: ✓

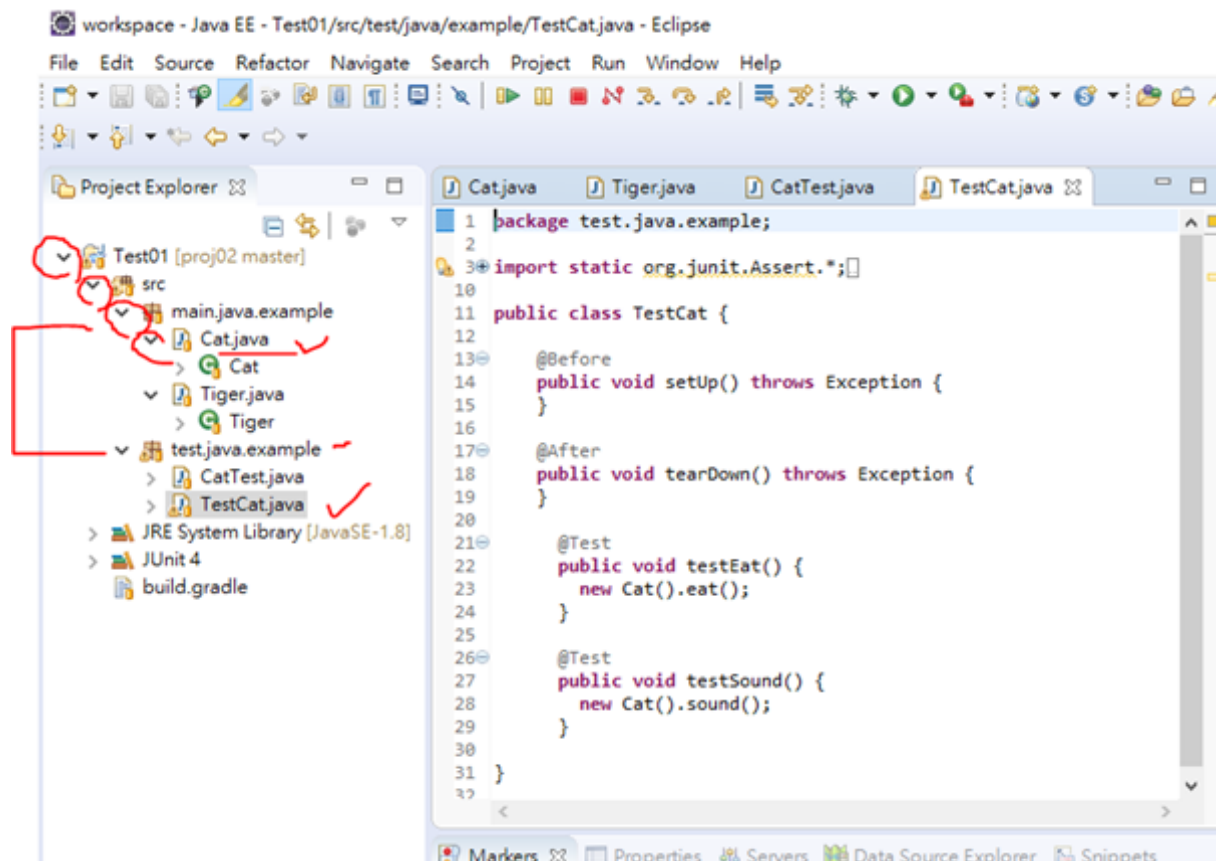
☐ Store in Secure Store

Host: 192.168.xx.xx

Protocol: ssh

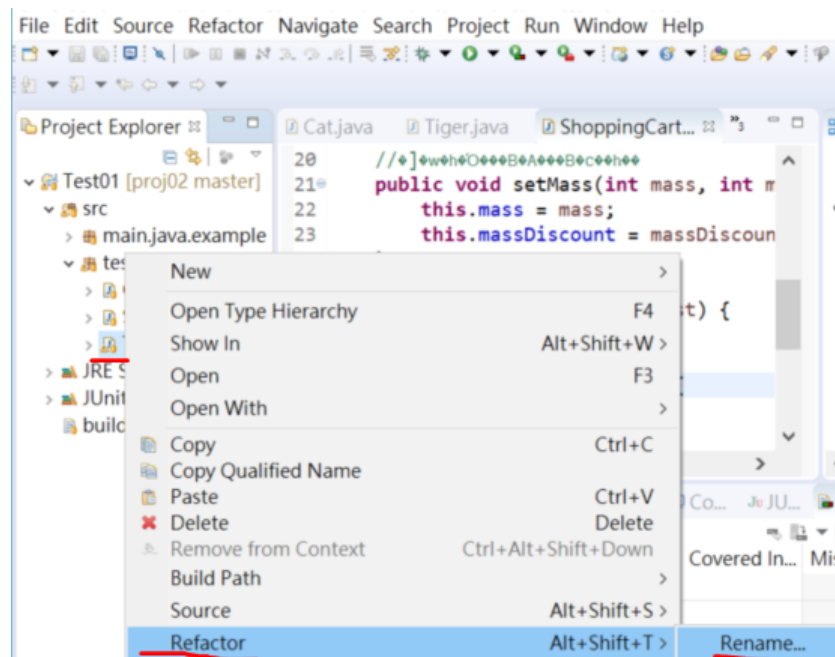
/usr/share/gitrepo/proj02, 按 ok, yes, next finish,

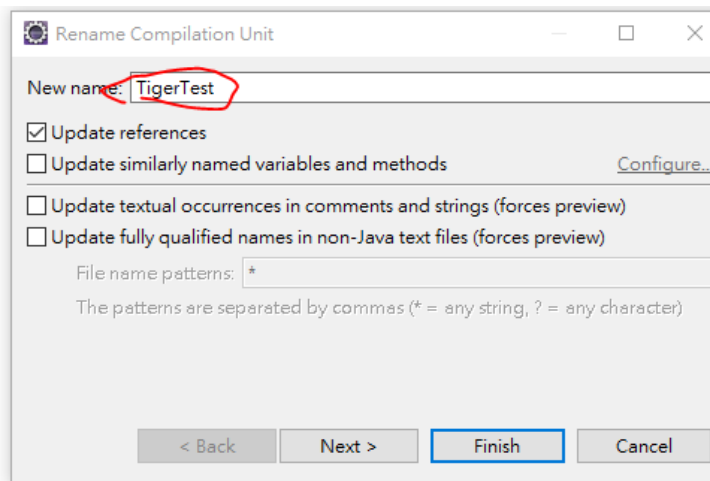
7. 從 **Git** 抓回專案程式碼, 到 **Eclipse** 個人桌面 IDE
編輯



8. 修改測試 Test Driver

(1) File - Refactor - Rename

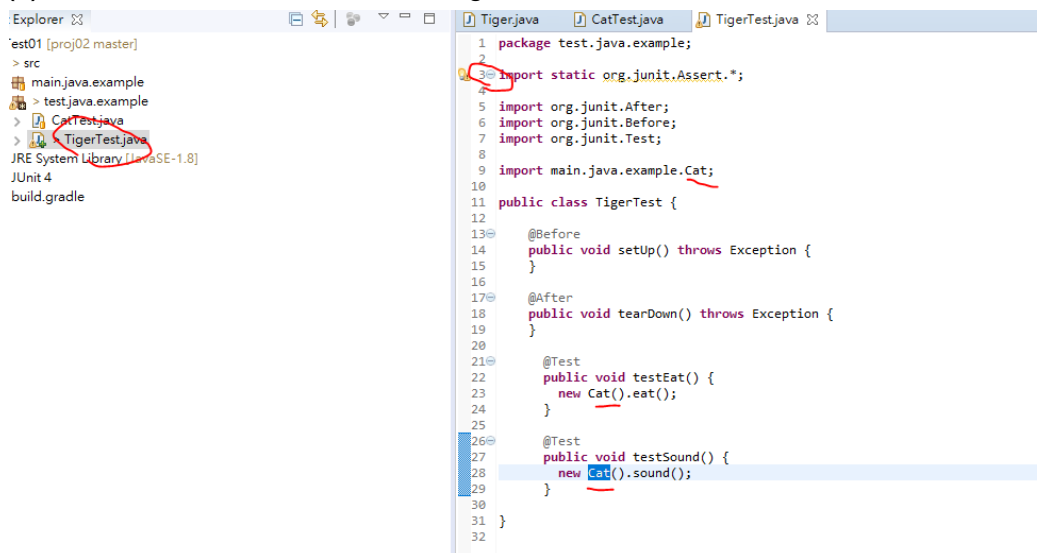




(2) 修改 test driver, 增加測試 code coverage

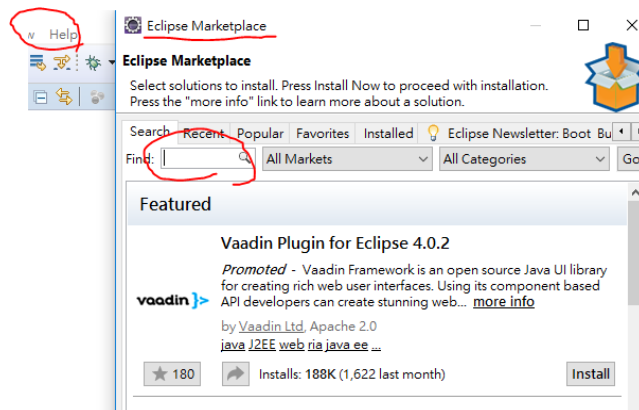
(a) 將 TestCat.java, 改成 TigerTest, 按右鍵 refactor - name

(b) 找到三隻cat, 將 3 個 cat 改成 Tiger,

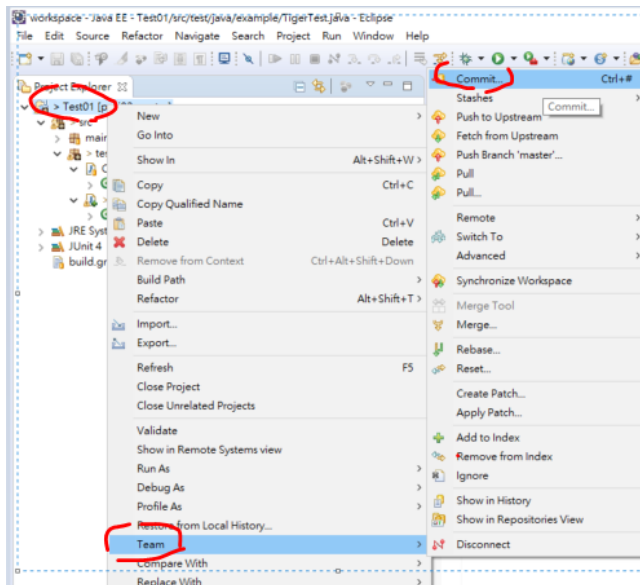


9. 安裝 Eclemma

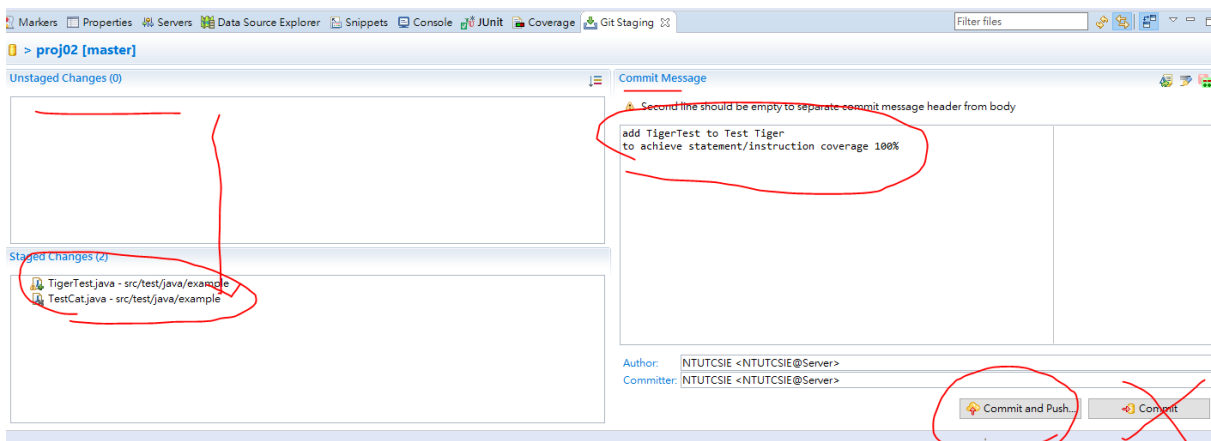
Eclipse Help Eclipse marketplace, eclemma, install



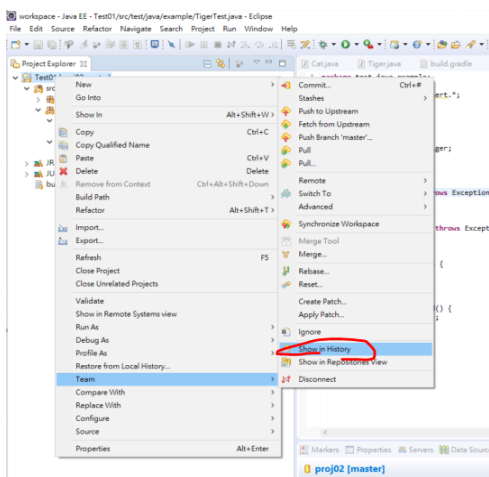
Add to index (將原本工作區的檔案, 移到暫存區 staged)

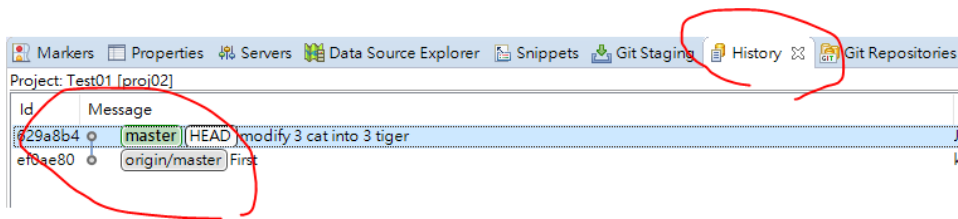


(2) 輸入版本控管中這個版本的說明, commit&Push(存到本地端儲存區、遠端伺服器儲存區)

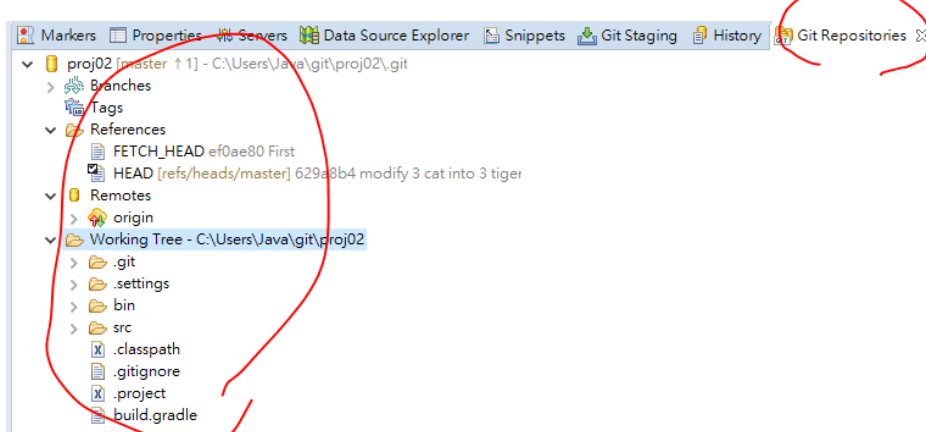


(3) 顯示 commit 歷史, 右鍵 - Team - show in History





(4) 右鍵 - Team - show in Reposit... view

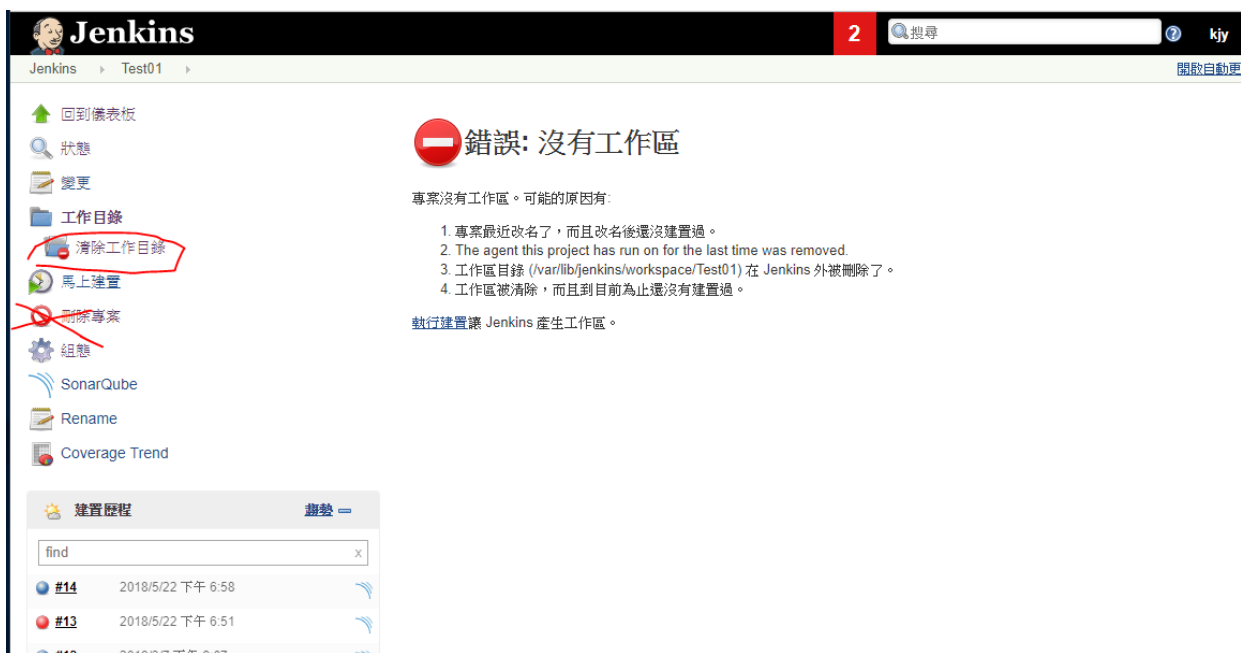


(5) 若剛剛只有按到 commit, 則在程式任意處加空白鍵, 再commit and push 到遠端

12. 到 Jenkins 查看工作目錄內的檔案

(1) Jenkins 第二層選單 - 工作目錄

(馬上建置 -> Jenkins 會到 Git 拉下最新 push 程式碼)

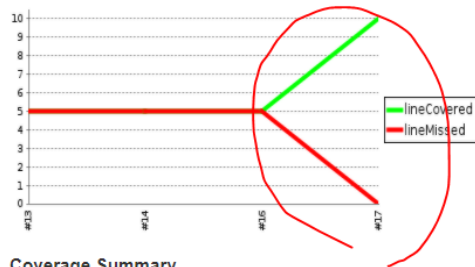


(2) 檢視 Code Coverage

Jenkins - Test01 - 馬上建置

檢視 JaCoCo Code coverage Trend, GREEN

Package: main.java.example



Coverage Summary

name	instruction	branch	complexity
main.java.example	M: 0 C: 22 100%	M: 0 C: 0 0%	M: 0 C: 6 100%

Coverage Breakdown by Source File

name	instruction	branch	complexity	
Cat	M: 0 C: 11 100%	M: 0 C: 0 0%	M: 0 C: 3 100%	M: 0 100%
Tiger	M: 0 C: 11 100%	M: 0 C: 0 0%	M: 0 C: 3 100%	M: 0 100%

13. 到 Jenkins 查看工作目錄內的檔案

Jenkins 第二層選單 - 工作目錄

(馬上建置 -> Jenkins 會到 Git 拉下最新 push 程式碼)

Workspace of Test01 on r

src / main / java / example

- [Cat.java](#) 189 B [檢視](#)
- [Tiger.java](#) 198 B [檢視](#)

(全部打包成 Zip 檔)

14. 觀察編譯狀況

Jenkins > Test01 > #17

回到專案
狀態
變更
Console Output
View as plain text
編輯建構資訊
刪除這次建構
Git Build Data
No Tags
Coverage Report
前一次建構

終端機輸出

```
Started by user jykuo
Building in workspace /var/lib/jenkins/workspac
> git rev-parse --is-inside-work-tree # timeou
Fetching changes from the remote Git repository
> git config remote.origin.url /usr/share/gitr
Fetching upstream changes from /usr/share/gitr
> git --version # timeout=10
using GIT_ASKPASS to set credentials
> git fetch --tags --progress /usr/share/gitr
> git rev-parse refs/remotes/origin/master^(co
> git rev-parse refs/remotes/origin/origin/mas
Checking out Revision 308b1841ed3504514d3617212
> git config core.sparsecheckout # timeout=10
> git checkout -f 308b1841ed3504514d3617212bd0
> git rev-list ef0ae80c729e10a7bde8b2c0816dc1b
[Test01] $ /bin/sh -xe /tmp/hudson6232284822441
+ gradle build
Starting a Gradle Daemon (subsequent builds wil
:compileJava UP-TO-DATE
:processResources UP-TO-DATE
:classes UP-TO-DATE
:jar UP-TO-DATE
:assemble UP-TO-DATE
:compileTestJava
:processTestResources UP-TO-DATE
:testClasses
:test
:check
:build

BUILD SUCCESSFUL

Total time: 8.985 secs
[Test01] $ /bin/sh -xe /tmp/hudson2626269128475
[Test01] $ /var/lib/jenkins/tools/hudson.plugin
Dsoner host.url=http://192.168.22.130:9090/ -o
```

15. 到 Redmine 中，觀察連結 Git Server Code

網站首頁 帳戶首頁 專案清單 網站管理 說明

Test01

概觀 活動 問題清單 建立新問題 甘特圖 日曆 新聞 文件 Wiki 檔案清單 設定

建立新儲存機制

版本控管 Git

主要儲存機制 ☒

代碼 test01
長度必須介於 1 至 255 個字元之間。僅允許使用小寫英文字母 (a-z)，阿拉伯數字，虛線與底線。
一旦儲存之後，代碼便無法再次被更改。

儲存機制路徑 * /usr/share/gitrepo/proj02
儲存機制是本機的空(bare)目錄(即：/gitrepo, c:\gitrepo)

路徑編碼 UTF-8
預設：UTF-8

報告最後認可的文件和目錄 ☐

建立 取消

Test01

概觀 活動 問題清單 建立新問題 甘特圖 日曆 新聞 文件 Wiki 檔案清單 儲存機制 設定

test01 @ master

		名稱
📁	.settings	
📁	src	
📄	.classpath	
📄	.gitignore	
📄	.project	
📄	build.gradle	

最新的修訂版清單

#	日期	作者	
eac786e0	2018-05-04 20:55	Student	add tigertes
c8eb8a54	2018-03-07 20:41	jykuo	PUSH REMOTE

檢視差異

[檢視所有的修訂版清單](#) | [檢視修訂版清單](#)

購物網站

概觀 活動 問題清單 建立新問題 甘特圖 日曆 新聞 文件 Wiki 檔案清單 儲存機制 設定

proj01 / src / main / java / example / Cat.java @ master

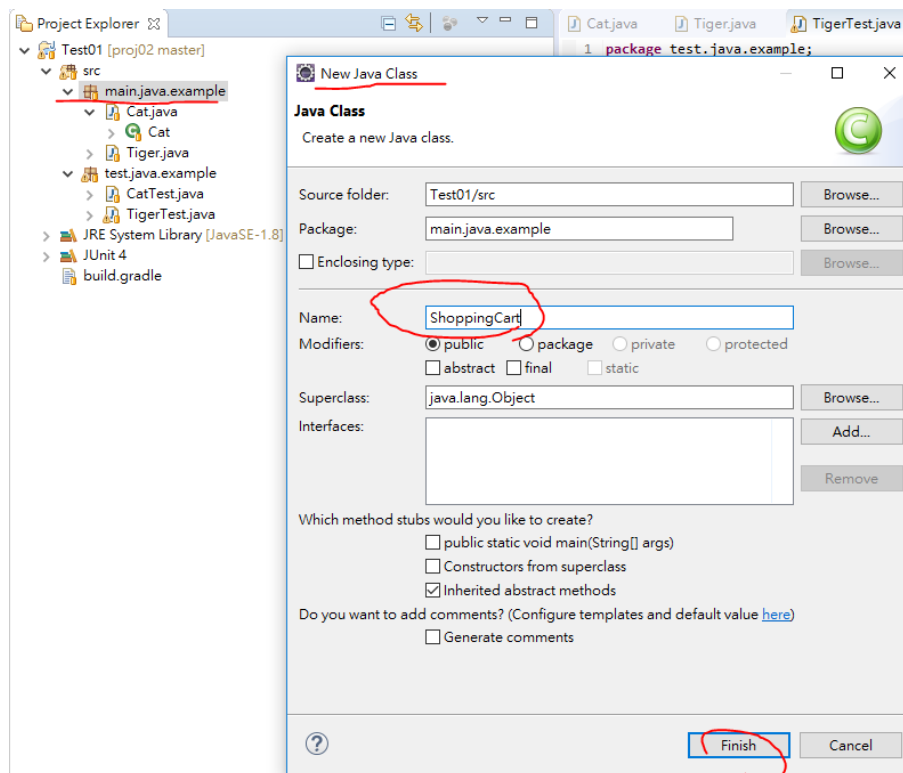
歷史 檢視 注釋 | 下載 (179 位元組 (B))

```
1 package example;
2
3 public class Cat {
4     public void eat() {
5         System.out.println("Cat eating.");
6     }
7
8     public void sound() {
9         System.out.println("Cat meow meow meow.");
10    }
11 }
```

Git

七、案例練習 ---- ShoppingCart

https://myweb.ntut.edu.tw/~jykuo/train/WebShoppingSpec_V_25.pdf



1. 類別需求描述

(1)資料屬性

- cost, 原始價格。
- vip=1, 會員, vipDiscount 會員折扣; vip=0, 非會員。
- 滿額 mass, massDiscount 滿額折扣。

(2)方法

- 設定資料屬性, 折扣0~100為0~100折。
- 計算總價
會員, 總價, 打 vipDiscount 折。非會員無折扣。
滿額 mass, 總價, 再打 massDiscount 折扣。

增加程式 production code, 在main.java.example, 按右鍵new
ShoppingCart 類別

#Eclipse 編碼修改, 正確存檔中文註解

Windows - Preference - General - Workspace - Text file encoding - Other: UTF-8

ShoppingCart.java

```
public class ShoppingCart {
    private int vip;
    private int vipDiscount;
    private int cost;
    private int mass;
    private int massDiscount;
    private int newCost;
    public ShoppingCart() {
        vip=0;    vipDiscount=0;
        cost = 0; newCost=0;
        mass = 0; massDiscount=100;
    }
    //設定是否為VIP, 若是則打多少折扣
    public void setVip(int vip, int vipDiscount) {
        this.vip = vip;
        this.vipDiscount = vipDiscount;
    }
    //設定多少是滿額, 滿額後打折多少
    public void setMass(int mass, int massDiscount) {
        this.mass = mass;
        this.massDiscount = massDiscount;
    }
    //設定原始價格
```

```

public void setCost(int cost) {
    this.cost = cost;
}
public void computeCost() {
    newCost = cost;
    //如果 VIP則可以打折
    if (vip==1) {
        newCost = newCost*vipDiscount;
    }
    //如果滿額則可以再打折
    if (cost>=mass) {
        newCost = newCost*massDiscount;
    }
}
public int getCost() {
    return newCost;
}
}

```

=====

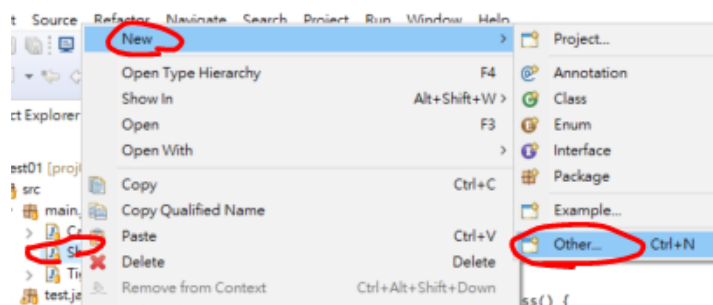
2. JUNIT 單元測試

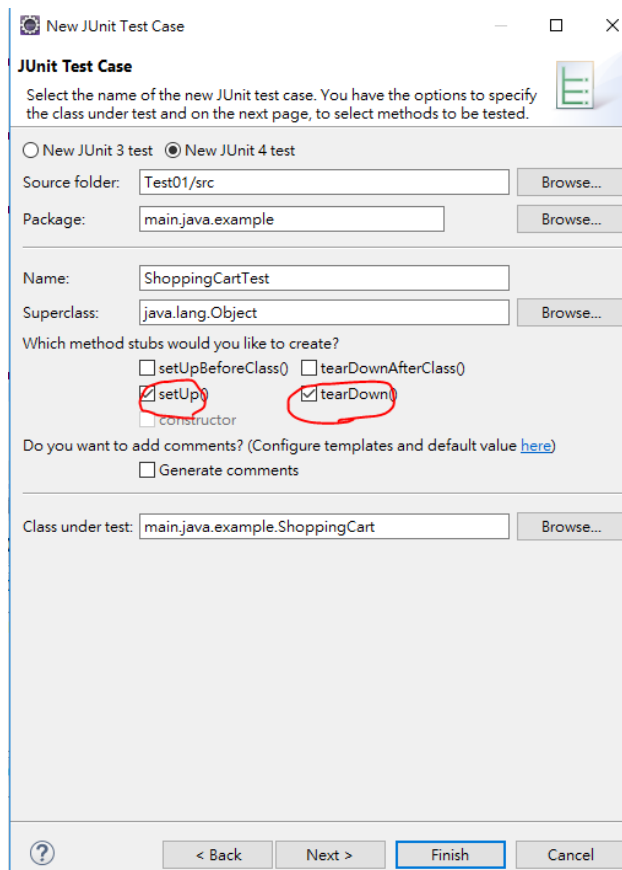
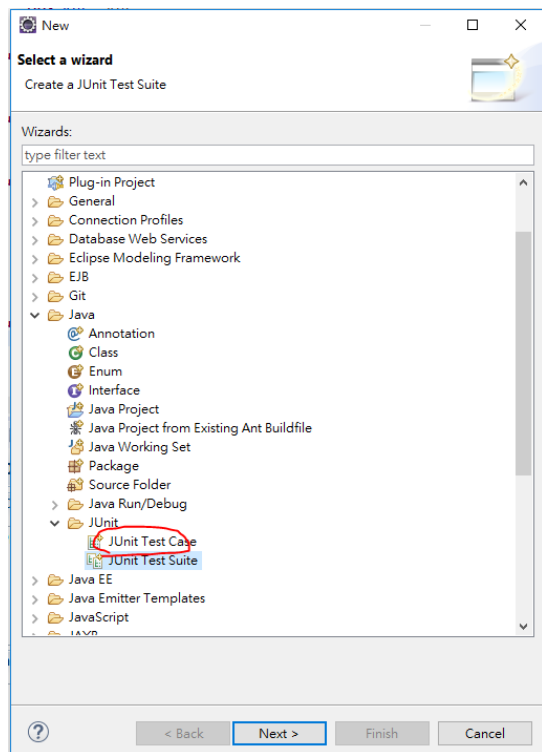
重點摘要

- (1) Run as - JUnit Test 測試程式 production code是否正確
若程式production code不正確(紅色), 要加以修改
- (2) Coverage as - JUnit Test 檢驗 Test Driver寫得是否完整 (測試程式是否完整)
看 production code 是否涵蓋度 100%且綠色, 否則要修改(增加) Test driver。

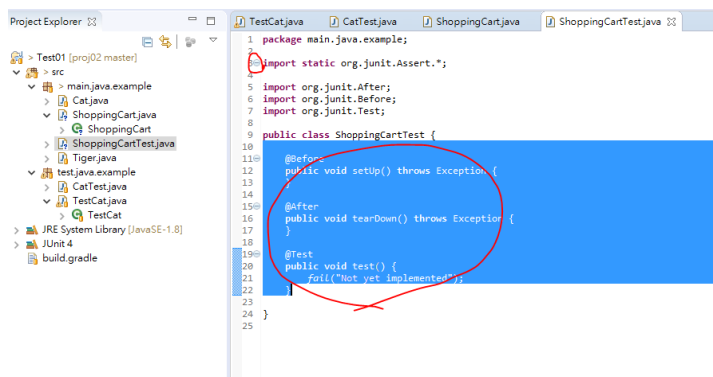
(1) 使用 JUnit 產生 test driver 框架

(a)點選 ShoppingCart.java按右鍵 (選 other)





(b) 替換程式碼



```
public class ShoppingCartTest {
    private ShoppingCart myCart;
    @Before
    public void setUp() throws Exception {
        myCart = new ShoppingCart();
    }
    @After
    public void tearDown() throws Exception {
        myCart = null;
    }
    @Test
```

```

public void test() {
    myCart.setCost(100);
    myCart.setMass(100, 90);
    myCart.setVip(0, 100);
    myCart.computeCost();
    assertEquals(90, myCart.getCost());
}
}

```

說明

setUp 測試初始化, 配置產生物件空間. 資料連線資源

tearDown 釋放空間、資源

@test 一個獨立方法測試

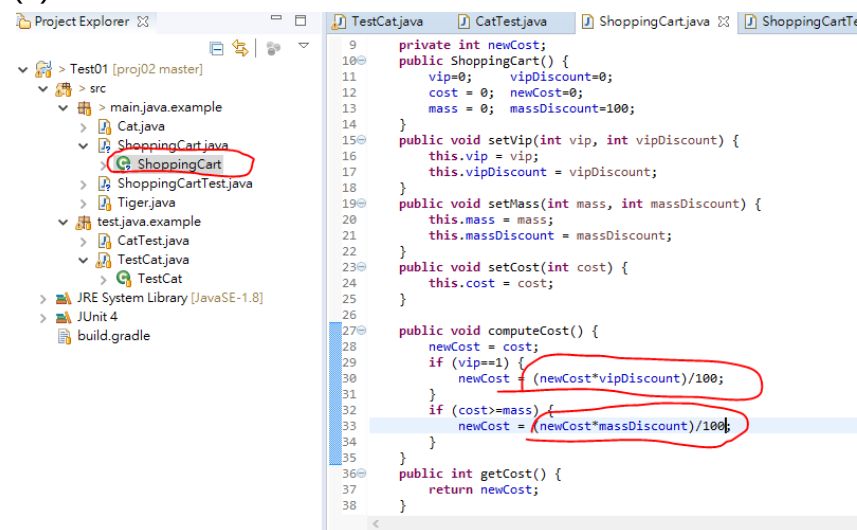
assertEquals(期望的正確執行結果, 呼叫要測試的method, 比對小數精確)

(c) 執行

ShoppingCartTest.java 按右鍵, Run as - Junit Test case

紅色 - production code 有錯

(d) 修改程式

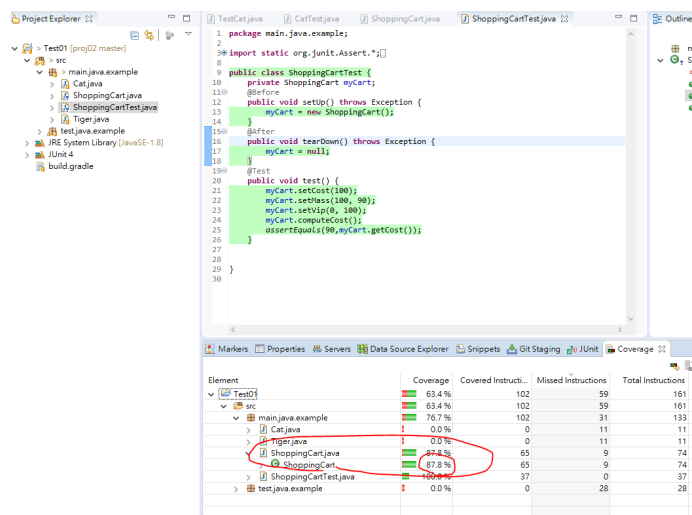


3. Test Code Coverage 練習

(1) 安裝Eclipse plug-in Junit + EClemma

(2) 查詢 main package 看 code coverage

按右鍵, Coverage As - Junit Test

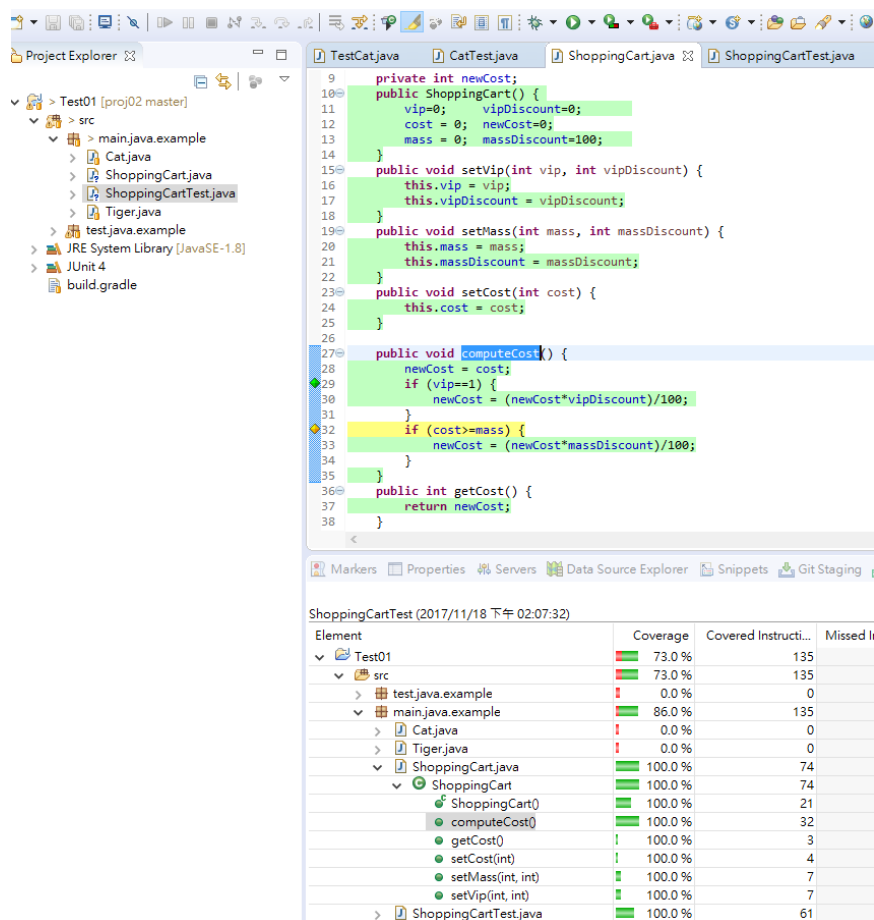


(3) 若coverage不足, 增修 test package 程式, 增加 test code coverage

(a)Eclipse 消除紅色

(b)Eclipse 消除黃色 (可代表 (1))

(c)查看 Jenkins JoCoCo

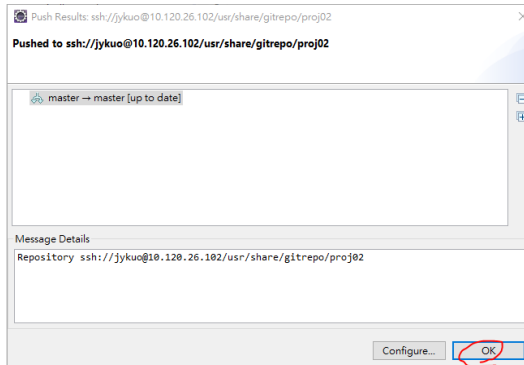
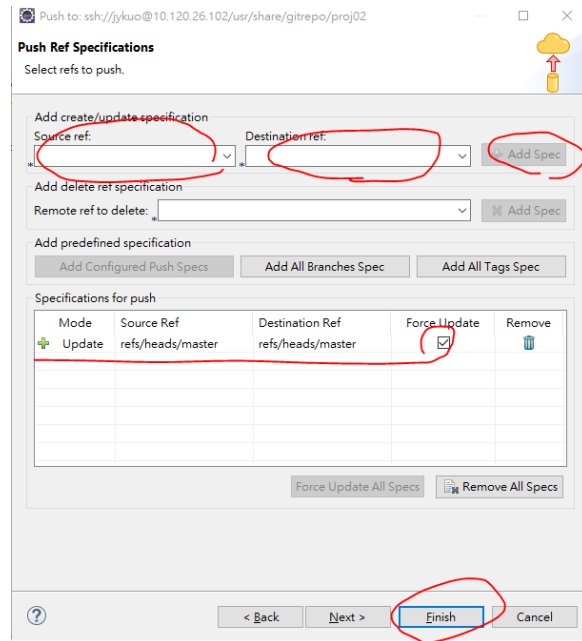
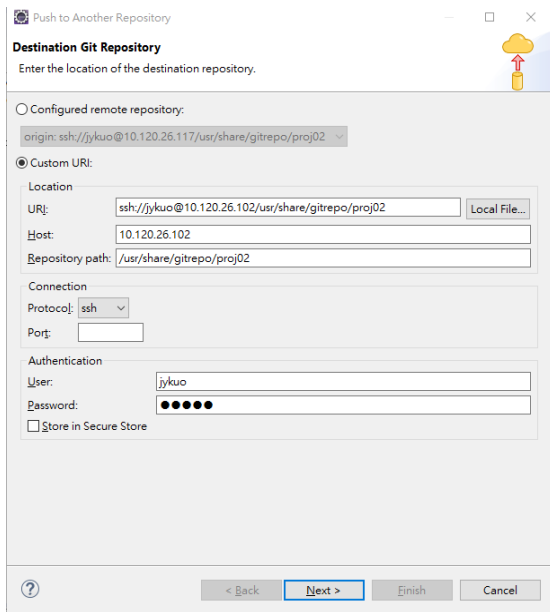


The screenshot shows an IDE with a Java project. The code editor displays the `ShoppingCartTest` class. A red circle highlights the `testVip` method. Below the code editor, a table shows the coverage report for the project.

Element	Coverage	Covered Instructi...	Missed Instruct
Test01	63.4 %	102	
src	63.4 %	102	
main.java.example	76.7 %	102	
Cat.java	0.0 %	0	
Tiger.java	0.0 %	0	
ShoppingCart.java	87.8 %	65	
ShoppingCart	87.8 %	65	
computeCost()	71.9 %	23	
ShoppingCart()	100.0 %	21	
getCost()	100.0 %	3	
setCost(int)	100.0 %	4	
setMass(int, int)	100.0 %	7	
setVip(int, int)	100.0 %	7	
ShoppingCartTest.java	100.0 %	37	

(4) 直接 remote push 到远端 Git

The screenshot shows the IDE's Git menu. The 'Team' menu item is highlighted, and the 'Push...' option is selected. A red circle highlights the 'Push...' option in the 'Team' menu.



(5)Git

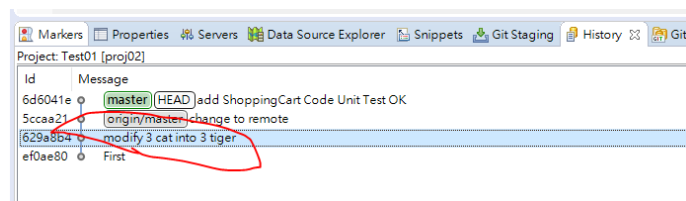
? working directory

+ 綠色 staged(add to index)

直筒黃色 (commit)本地端儲存區(個人電腦)

(push) 遠端儲存區 (中心控制, 企業伺服器)

點選第二版, 右鍵 create branch- Tiger



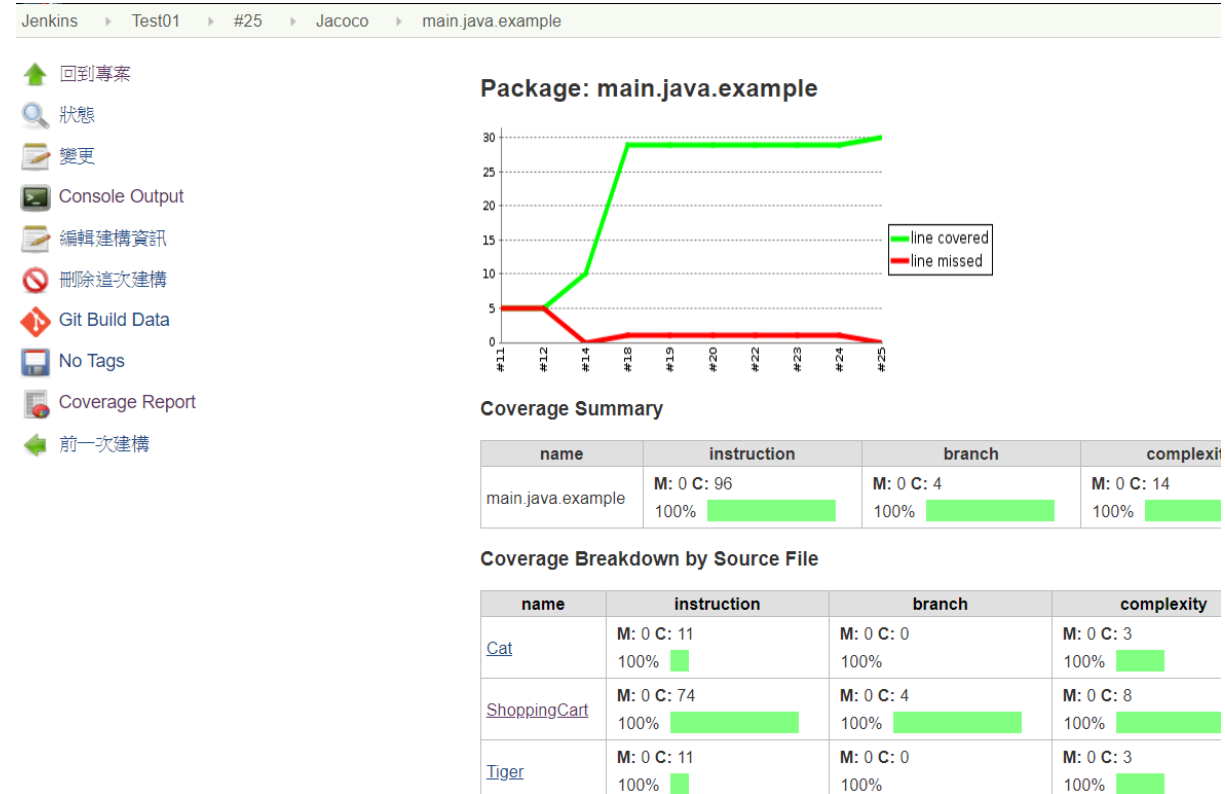
點選ShoppingCart版, 右鍵 create branch - ShoppingCart

點選 BRANCH - 右鍵 - CHECKOUT

跳到那個版本

跳到最新版本 push 到 remote git

(6) push 到遠端, 使用 Jenkins 建置, 查看 Code Coverage



@Test

```
public void testNoVIPMass() {  
    myCart.setCost(100);  
    myCart.setMass(100, 90);  
    myCart.setVip(0, 100);  
    myCart.computeCost();  
    assertEquals(90, myCart.getCost());  
}
```

@Test

```
public void testVIPMass() {  
    myCart.setCost(100);  
    myCart.setMass(100, 90);  
    myCart.setVip(1, 90);  
    myCart.computeCost();  
    assertEquals(81, myCart.getCost());  
}
```

4. ShoppingCart 新增 foo method

```
public int foo(int A, int B, int X) {  
    int Y = 0;  
    if ((A>1) && (B==0)) {  
        Y=A;  
    }  
    if ((A==2) || (X>1)) {  
        Y=X;  
    }  
    return Y;  
}
```

寫 test method 達到 instruction, branch 100%

5. 白箱與黑箱測試配合

若程式變成以下 code

vip =2 會變成 newCost=0 不用錢

單純以 black box 設計的測試案例無法發現問題

必須使用 code coverage 才能抓出程式問題

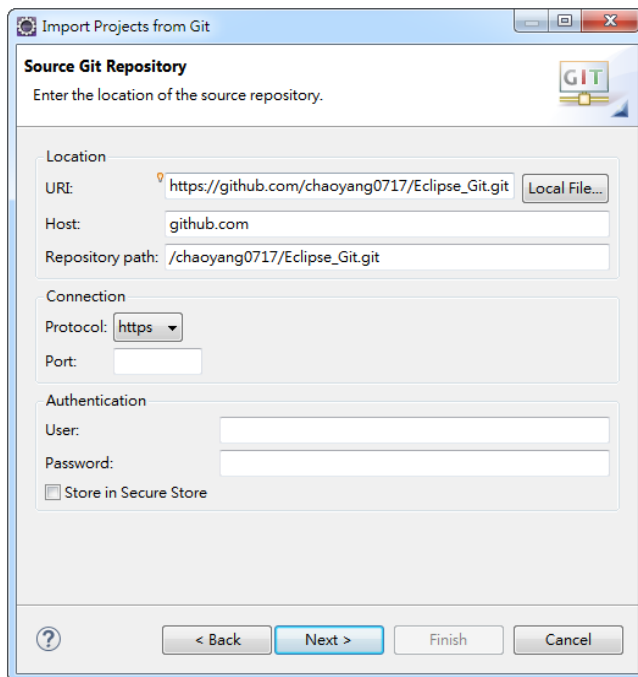
```
public void computeCost() {  
    newCost = cost;  
    if (vip==2) {  
        newCost =0;  
        return;  
    }  
    //如果 VIP則可以打折  
    if (vip==1) {  
        newCost = (newCost*vipDiscount)/100;  
    }  
    //如果滿額則可以再打折  
    if (cost>=mass) {  
        newCost = (newCost*massDiscount)/100;  
    }  
}
```

6. 遠端主機可為 各種 Git 儲存庫

(1) 遠端主機可為 Dropbox

參考資料http://www.cc.ntut.edu.tw/~jykuo/train/022EGIT2017_66.pdf, page 56

(2) 遠端主機可為 GitHub



摘要

1. 工程師 在自己個人電腦 eclipse 開發程式, 改好CODE
push & commit -> Jenkins (CI-get Git Newest Code)->Sonarqube
 2. 品管人員、專案經理, 登入 Sonarqube查看團隊的程式碼品質
專案 - Code - 程式碼 - Issue
-

http://www.cc.ntut.edu.tw/~jykuo/train/022EGIT2017_66.pdf

mkdir Test

cd Test

git init

ls -la

cd .git

ls -al

cd ..

gedit product.java

git status

git add Product.java

git status

git commit Product.java -m 'First version'

git status

http://www.cc.ntut.edu.tw/~jykuo/train/05GIT2017_77.pdf

git commit --amend

ctrl-O

ctrl-X

git log

sudo apt-get update

sudo apt-get upgrade

=====

七、TDD

=====

參考資料

http://www.cc.ntut.edu.tw/~jykuo/train/003TDD_Bowling_2017_23.pdf

<http://www.cc.ntut.edu.tw/~jykuo/train/VMwareVmdk.pdf>

規格文件

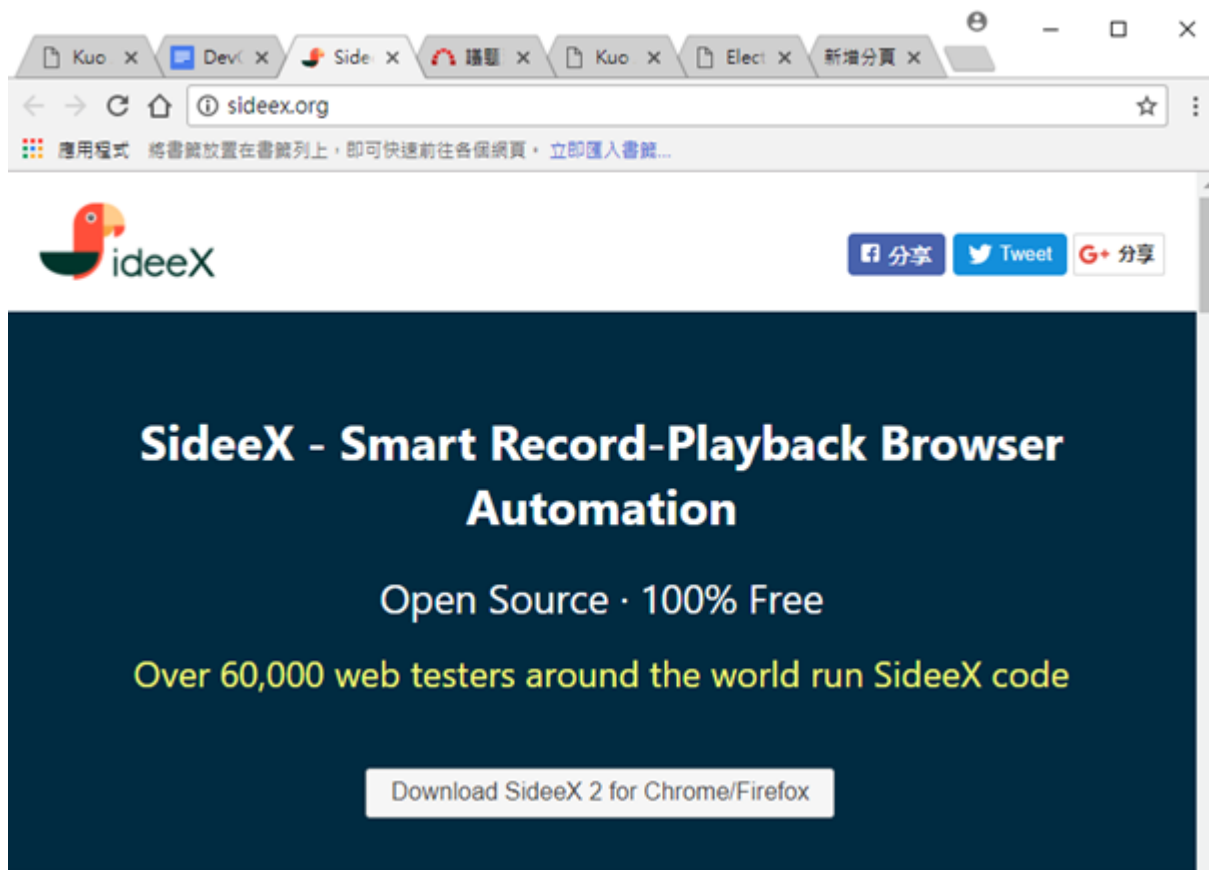
http://www.cc.ntut.edu.tw/~jykuo/train/WebShoppingSpec_V_25.pdf

=====

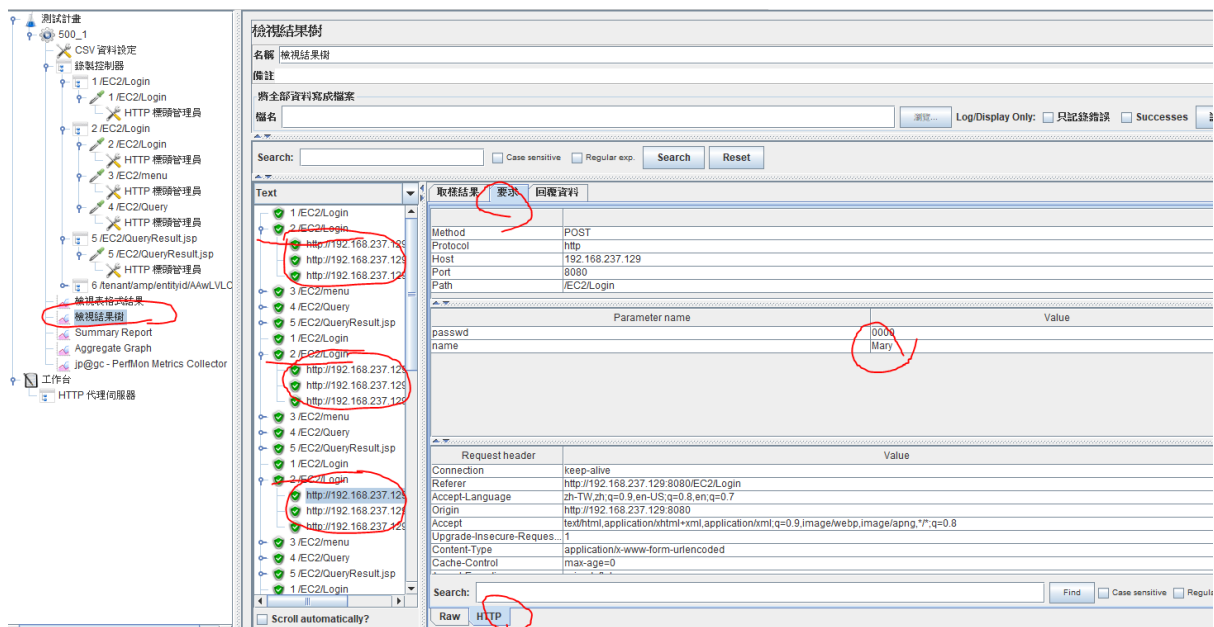
八、Web 功能整合測試工具

<http://sideex.org/>

=====



http://www.cc.ntut.edu.tw/~jykuo/train/106JMeter02_98.pdf



九、測試計畫書

=====

<http://www.cc.ntut.edu.tw/~jykuo/train/performancePlan.pdf>

<http://www.cc.ntut.edu.tw/~jykuo/train/VMware-player-14.1.1-7528167.exe>

http://www.cc.ntut.edu.tw/~jykuo/train/107Redmine_99.pdf

安全 | <https://jmeter-plugins.org/wiki/PerfMon/>

書籤置在書籤列上，即可快速前往各個網頁，立即匯入書籤...

JMeter Plugins > Documentation > PerfMon



jmeter-plugins.org

Every load test needs some sexy features!

Install

Browse Plugins

Documentation

Usage Statistics

Support Forums

Servers Performance Monitoring

Download

Introduction

During a load test, it is important to know the health of the servers loaded. It is also nice to see if you are targeting a cluster if the load is correctly dispatched. To address this, the plugin package now supports server monitoring! Using it, you can monitor CPU, Memory, Swap, Disks I/O and Networks I/O on almost all platforms!

Here is how the plugin looks like. It shows the CPU usage of 4 servers involved in the load test:

Servers Performance Monitoring

listeners. After running the test you may load saved file into GUI and see the values timeline.

JMeter Properties

- `jmeterPlugin.perfmon.interval` - metrics collection interval in milliseconds
- `jmeterPlugin.perfmon.useUDP` - true/false, enabling UDP connection try after failed TCP connection attempt
- `jmeterPlugin.perfmon.label.useHostname` - true/false, enable using "short" hostnames, default pattern is `(([\w\.-]+)\.)*`
- `jmeterPlugin.perfmon.label.useHostname.pattern` - string (escaped), regular expression to extract hostname (first group is matched)
- e.g. Default pattern would be: `jmeterPlugin.perfmon.label.useHostname.pattern=([\w\.-]+)\.)*`
- e.g. Pattern for EC2 us-east/west subdomain matching: `jmeterPlugin.perfmon.label.useHostname.pattern=(([\w\.-]+)\.us-(east|west)-[0-9])\.)*`
- `forcePerfmonFile` - true/false, enabling it makes JMeter to write JTL file with perfmon metrics in the current directory

Common Considerations

PerfMon Server Agent did support only few metrics in versions up to 0.4.2. The old agent still supported in PerfMon Metrics Collector version 0.5.0+. However, version 0.5.0 ships new **ServerAgent** which provide over 75 separate metrics, support per-process CPU and Memory metrics and even custom metrics for measuring whatever you want: file sizes, database row counts, Java heap sizes and garbage collections.

Specifying Metric Params

On this

1. In
2. M
3. H

4. U

2. M
3. H

4. U

5. C

Follow

undera / perfmon-agent

Watch

11

Star

46

Fork

<> Code

Issues 2

Pull requests 0

Projects 0

Insights

Branch: master

perfmon-agent / README.md

Find file

Copy

undera Update doc

47c3dcc on 9 Aug

1 contributor

337 lines (273 sloc) | 11.5 KB

Raw

Blame

History

build passing

PerfMon Server Agent

Download

ServerAgent-2.2.3.zip

previous ServerAgent-2.2.1.zip)

Installation

You do not need any root/admin privilege. You can just unzip the the ServerAgent-X.X.X.zip somewhere on the server. Then launch the agent using `startAgent.sh` script on Unix, or `startAgent.bat` script on Windows.

cd Downloads
ls -al
unzip ServerAgent
source

```
ec2@ec2-virtual-machine: ~/Downloads
ec2@ec2-virtual-machine:~$ glances
ec2@ec2-virtual-machine:~$ cd Downloads
ec2@ec2-virtual-machine:~/Downloads$ ls -al
total 16252
drwxr-xr-x  4 ec2 ec2    4096  5月  6 11:16 .
drwxr-xr-x 24 ec2 ec2    4096  5月  6 14:06 ..
drwxrwxr-x  9 ec2 ec2    4096  4月 19 2017 apache-tomcat-8.0.43
-rw-rw-r--  1 ec2 ec2 9323896  4月 19 2017 apache-tomcat-8.0.43.tar.gz
-rw-rw-r--  1 ec2 ec2  10821  2月 25 2013 CMDRunner.jar
drwxr-xr-x  2 ec2 ec2    4096  2月 25 2013 lib
-rw-rw-r--  1 ec2 ec2  85433  2月 25 2013 LICENSE
-rw-rw-r--  1 ec2 ec2 3676668  5月  6 11:16 ServerAgent-2.2.1.zip
-rw-rw-r--  1 ec2 ec2 3447815  5月  6 11:16 ServerAgent-2.2.3.zip
-rw-rw-r--  1 ec2 ec2  62848  2月 25 2013 ServerAgent.jar
-rwxr-xr-x  1 ec2 ec2    63  2月 25 2013 startAgent.bat
-rwxr-xr-x  1 ec2 ec2    74  2月 25 2013 startAgent.sh
ec2@ec2-virtual-machine:~/Downloads$ unzip ServerAgent-2.2.1.zip
Archive:  ServerAgent-2.2.1.zip
replace startAgent.sh? [y]es, [n]o, [A]ll, [N]one, [r]ename: N
ec2@ec2-virtual-machine:~/Downloads$ source startAgent.sh
```

```
-rw-rw-r--  1 ec2 ec2  62848  2月 25 2013 ServerAgent.jar
-rwxr-xr-x  1 ec2 ec2    63  2月 25 2013 startAgent.bat
-rwxr-xr-x  1 ec2 ec2    74  2月 25 2013 startAgent.sh
ec2@ec2-virtual-machine:~/Downloads$ source startAgent.sh
INFO 2018-05-06 14:12:12.797 [kg.apc.p] (): Binding UDP to 4444
INFO 2018-05-06 14:12:13.828 [kg.apc.p] (): Binding TCP to 4444
INFO 2018-05-06 14:12:13.840 [kg.apc.p] (): JP@GC Agent v2.2.0 started
INFO 2018-05-06 14:12:19.744 [kg.apc.p] (): Accepting new TCP connection
INFO 2018-05-06 14:12:19.770 [kg.apc.p] (): Yep, we received the 'test' comma
nd
INFO 2018-05-06 14:12:19.810 [kg.apc.p] (): Starting measures: cpu:
INFO 2018-05-06 14:12:43.587 [kg.apc.p] (): Client disconnected
INFO 2018-05-06 14:14:05.166 [kg.apc.p] (): Accepting new TCP connection
INFO 2018-05-06 14:14:05.182 [kg.apc.p] (): Yep, we received the 'test' comma
nd
INFO 2018-05-06 14:14:05.223 [kg.apc.p] (): Starting measures: cpu:
INFO 2018-05-06 14:14:42.744 [kg.apc.p] (): Client disconnected
INFO 2018-05-06 14:17:19.489 [kg.apc.p] (): Accepting new TCP connection
INFO 2018-05-06 14:17:19.536 [kg.apc.p] (): Yep, we received the 'test' comma
nd
INFO 2018-05-06 14:17:19.558 [kg.apc.p] (): Starting measures: disks i/o: c
pu:
memory:
INFO 2018-05-06 14:17:31.738 [kg.apc.p] (): Client disconnected
```

<http://www.cc.ntut.edu.tw/~jykuo/train/106IssueSOP.pdf>

http://www.cc.ntut.edu.tw/~jykuo/train/106REQM_SOP.pdf

<http://www.cc.ntut.edu.tw/~jykuo/train/ansibleLab2017.txt>

http://www.cc.ntut.edu.tw/~jykuo/train/05GIT2017_77.pdf

http://www.cc.ntut.edu.tw/~jykuo/train/022EGIT2017_66.pdf

<https://www.dropbox.com/s/ve852bgmyouvf3l/GitHubSetup.exe?dl=0>

(1)Git命令列(2)Source Tree(3)TortoiseGit