
DevOps

自動化持續整合測試與建置

郭忠義

jykuo@ntut.edu.tw

臺北科技大學資訊工程系

DevOps 目標

❑ 企業組織開發產品目標

- 面對多種使用者、具備多樣功能。
- 頻繁交付產品、部署週期短。
- 虛擬化和雲端運算基礎設施。

❑ 開發和業務紛爭

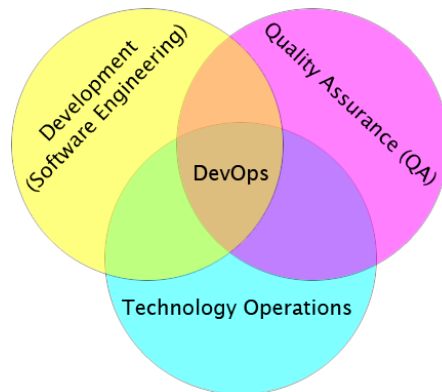
- 業務為滿足客戶，要求開發人員做「Change」。

❑ 開發和維運紛爭

- 開發人員為滿足使用者需求，不斷更新版本，不管硬體網路部署問題，但新版本會造成新問題，增加維運人員工作量。
- 維運人員為成功部署，可能變更設定，造成上線責任歸屬紛爭。
- 維運遇到問題，軟體開發人員不一定協助處理，兩者間產生「困惑牆Wall of Confusion」。

DevOps 目標

- ❑ “開發Development”與“維運Operations”整合。
 - 強調溝通 (Communication)、合作(Collaboration)、整合(Integration)及自動化(Automation)加強開發人員與維運和品質控制人員合作。
 - DevOps指導原則幫助解決Wall of Confusion問題，提高軟體品質、加快開發速度和避免重工。
- ❑ 敏捷軟體開發 (Agile Software Development) 方法。
 - 透過跨部門合作及自動化實現快速、頻繁發佈軟體。



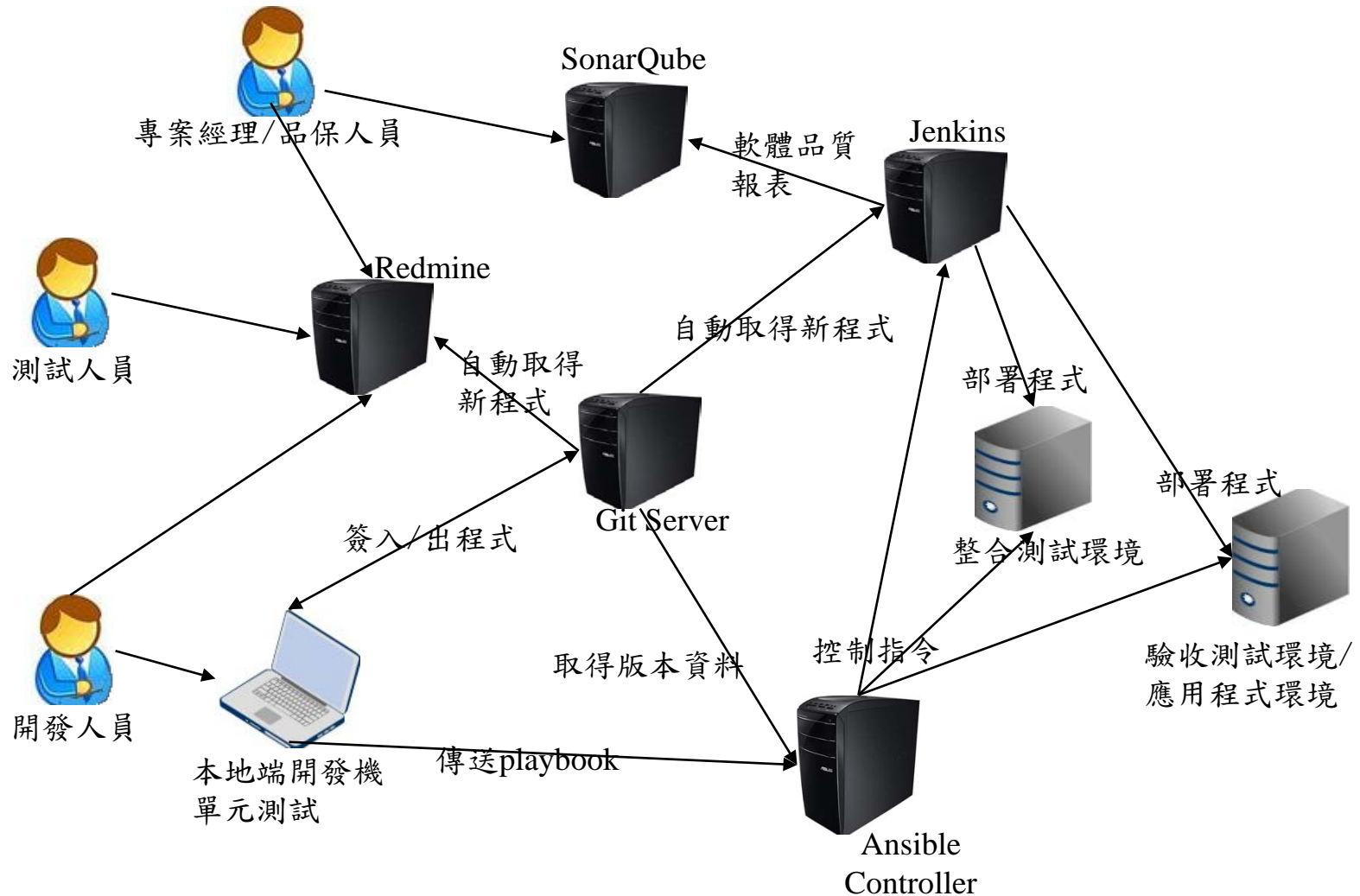
DevOps 目標

- ❑ CI/CD持續整合與持續發展
 - 扮演角色
 - 與傳統開發流程的不同
- ❑ 系統上線後如何持續監控狀態以修復程式
- ❑ 建立Git版本控管帳號
- ❑ 在Jenkins建立CI/CD的工作任務
 - 透過Jenkins自動將新版本部署至環境上

自動化建置與工具整合



DevOps開發流程



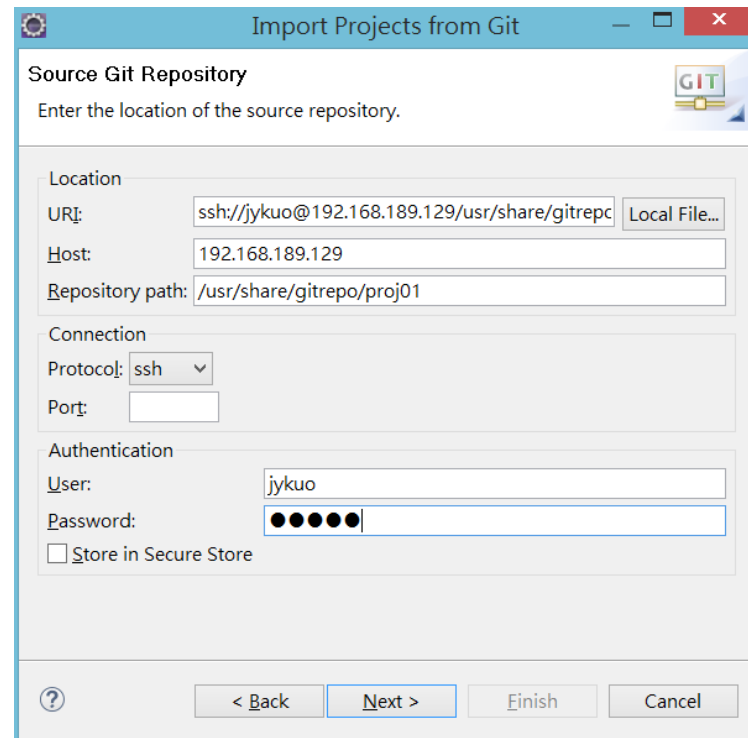
DevOps開發流程

□ DevOps開發流程

- 簽入(Commit)：程式在本地端開發機測試後，簽入版本控制系統Git。
- 建置 (Build)：持續整合系統Jenkins CI，偵測到版本更新，自動從Git取得最新程式進行建置與測試。。
- 單元測試：Jenkins 完成建置後，自動執行指定的單元測試。
- 部署到測試環境 (Deploy to Test Environment)：完成單元測試後，Jenkins 將程式部署到跟應用環境 (Production Environment) 相近的驗收測試環境進行測試。
- 驗收測試：在驗收測試環境進行自動化功能測試，例如 Selenium 測試。開發人員或客戶手動進行跟實際情況類似的驗收測試。
- 部署到應用環境 (Deploy to Production Environment)：通過所有測試後，將最新版本部署到實際應用環境。

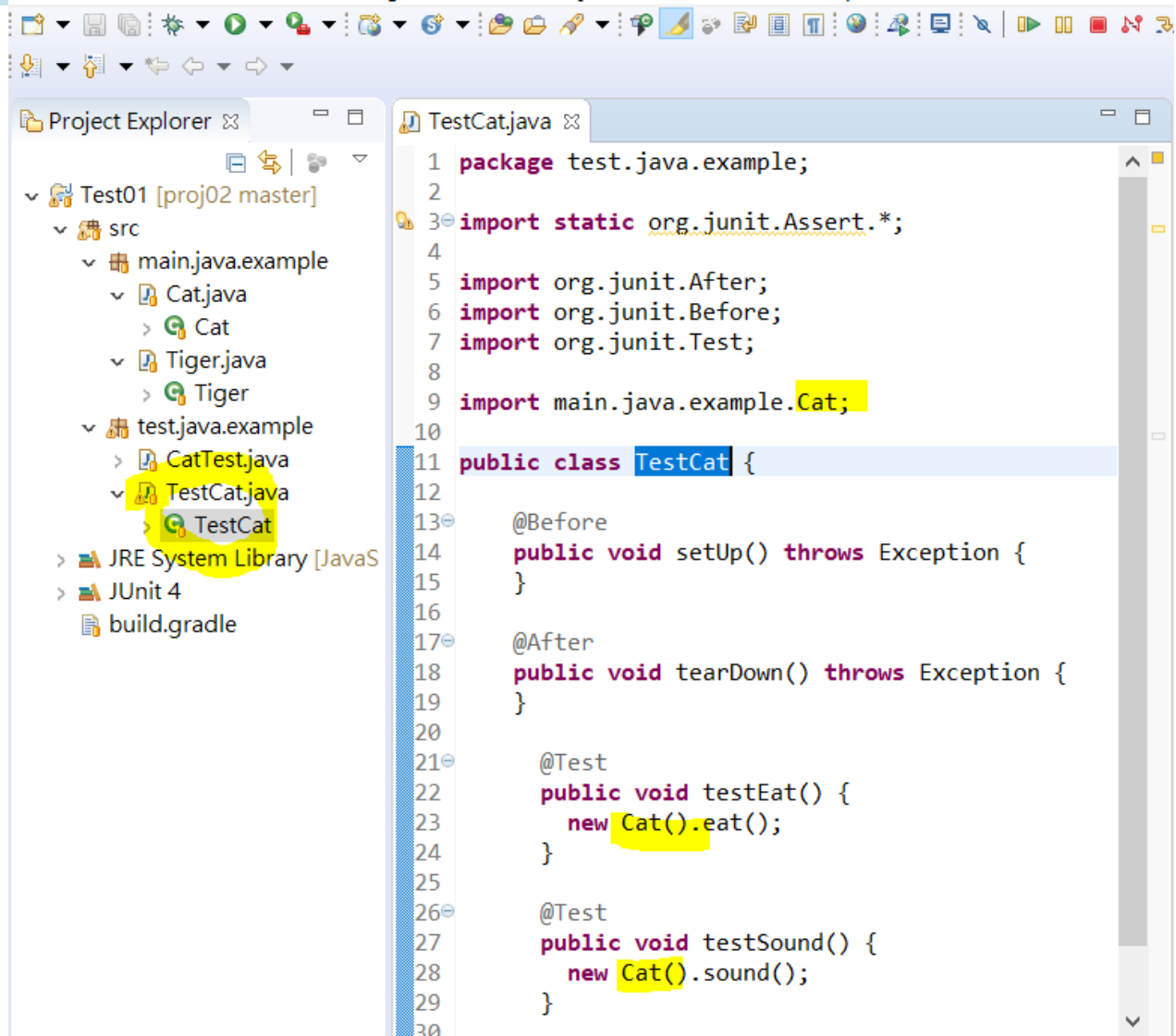
Eclipse 簽出程式

- ❑ 啟動虛擬機器
- ❑ 使用Eclipse，以新的workspace，從Git簽出程式
 - Host: 在這裡鍵入方程式。192.168.xx.xx
 - Repository path: /usr/share/gitrepo/proj02



Eclipse改修程式

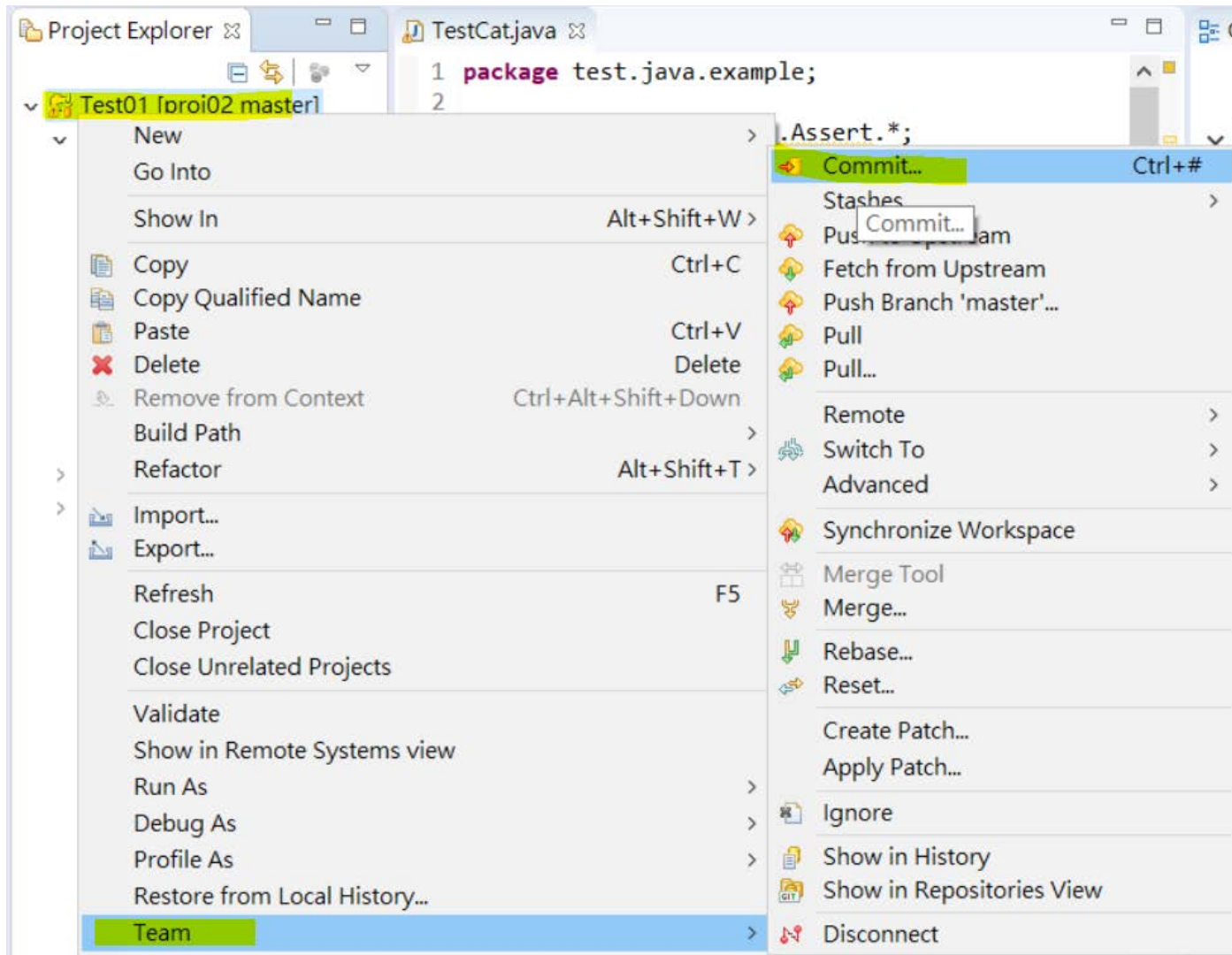
- ❑ TestCat
- ❑ 測試程式
 - 三個Cat
 - 改成Tiger



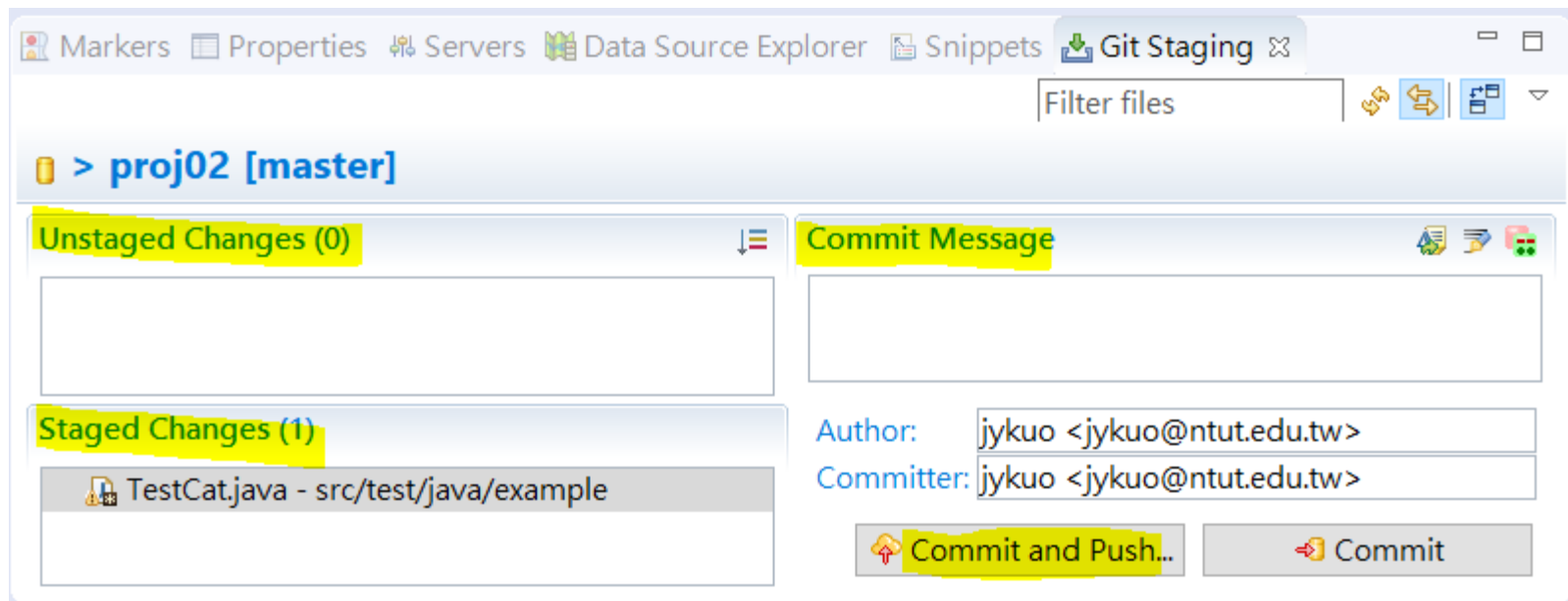
The screenshot shows the Eclipse IDE interface. On the left, the Project Explorer displays a project named 'Test01 [proj02 master]'. Under the 'src' folder, there are two packages: 'main.java.example' and 'test.java.example'. The 'test.java.example' package contains three files: 'CatTest.java', 'TestCat.java', and 'TestCat'. The 'TestCat.java' file is selected and highlighted in yellow. On the right, the editor window shows the code for 'TestCat.java'. The code is as follows:

```
1 package test.java.example;
2
3 import static org.junit.Assert.*;
4
5 import org.junit.After;
6 import org.junit.Before;
7 import org.junit.Test;
8
9 import main.java.example.Cat;
10
11 public class TestCat {
12
13     @Before
14     public void setUp() throws Exception {
15     }
16
17     @After
18     public void tearDown() throws Exception {
19     }
20
21     @Test
22     public void testEat() {
23         new Cat().eat();
24     }
25
26     @Test
27     public void testSound() {
28         new Cat().sound();
29     }
30 }
```

簽入 Git





簽入 Git





持續整合測試查看Coverage


Jenkins


 新增作業

 使用者


 建置歷程

 **管理 Jenkins**


 我的視景


 Credentials


管理 Jenkins




您的 Reverse Proxy 設



新版的 Jenkins (2.73) 



設定系統
設定全域設定



設定全域安全
保護 Jenkins

SonarQube servers

Environment variables

☐ Enable injection of SonarQube server configuration as build environment variables

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

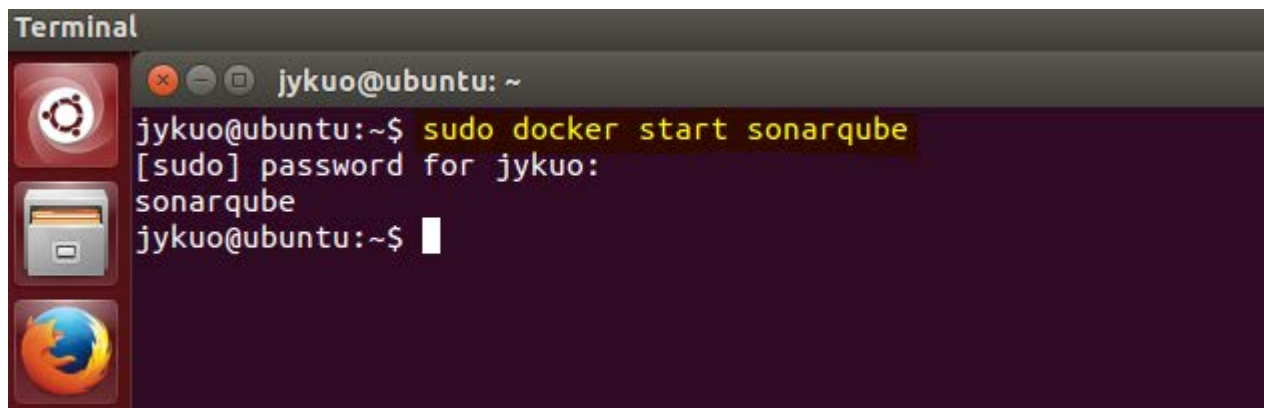
SonarQube installations

Name	MySonar
Server URL	http://192.168.78.128:9000/

儲存

Apply

持續整合測試查看Coverage

A terminal window titled "Terminal" with a dark background. The window has a sidebar on the left with icons for Dash, Home, and Firefox. The terminal text shows the user "jykuo@ubuntu: ~" entering the command "sudo docker start sonarqube". It prompts for a password, which is entered as "sonarqube", and then returns to the shell prompt "jykuo@ubuntu:~\$".

```
Terminal
jykuo@ubuntu: ~
jykuo@ubuntu:~$ sudo docker start sonarqube
[sudo] password for jykuo:
sonarqube
jykuo@ubuntu:~$
```

持續整合測試查看Coverage

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

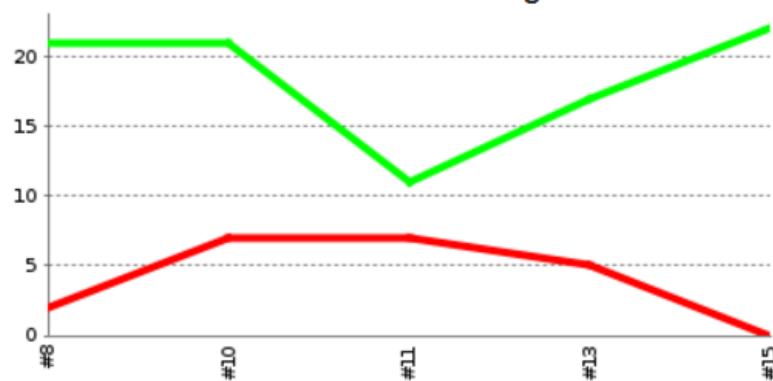
My Test



永久連結

- [最新建置 \(#15\), 49 秒 以前](#)
- [最新穩定建置 \(#15\), 49 秒 以前](#)
- [最新成功建置 \(#15\), 49 秒 以前](#)
- [最新失敗建置 \(#14\), 9 分 51 秒 以前](#)
- [最新不成功建置 \(#14\), 9 分 51 秒 以前](#)

Code Coverage Trend



建置歷程

趨勢

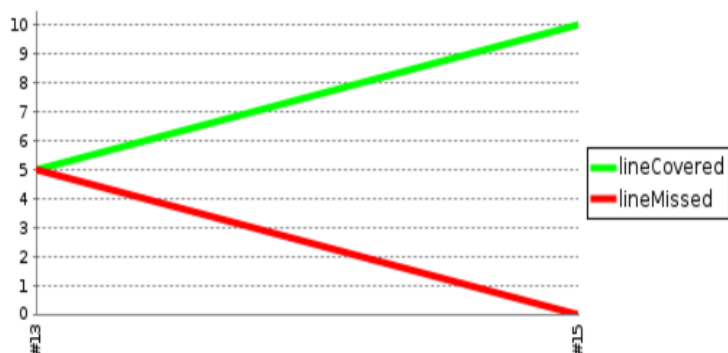
find

x

#15 2017/8/11 上午 6:51

持續整合測試查看Coverage

Package: main.java.example



Coverage Summary

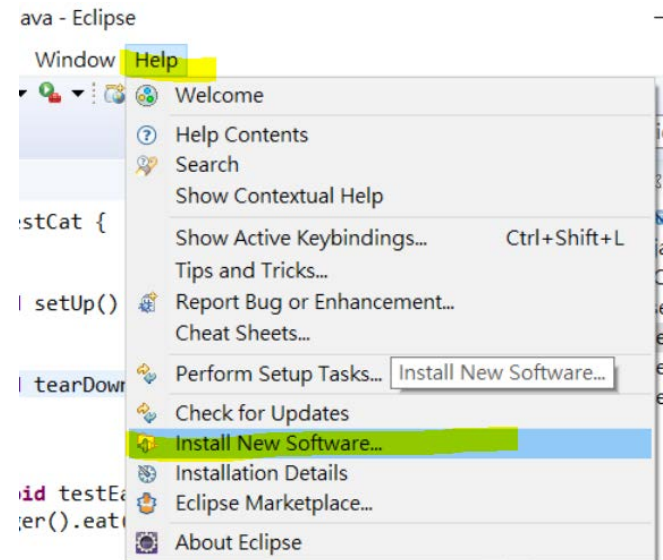
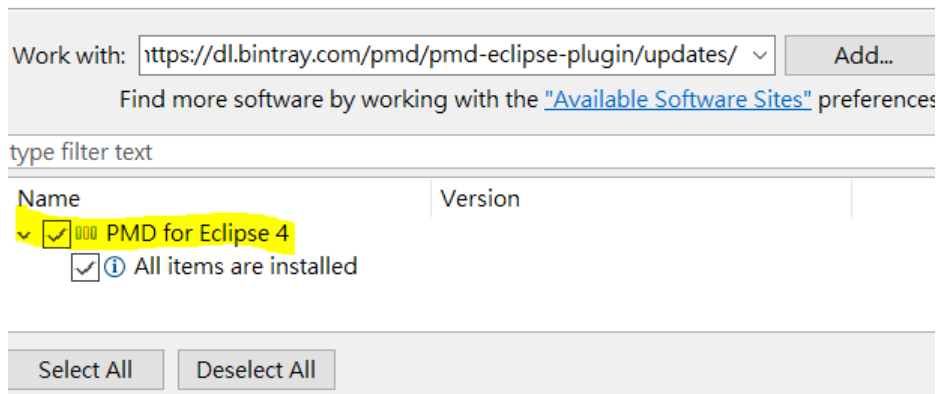
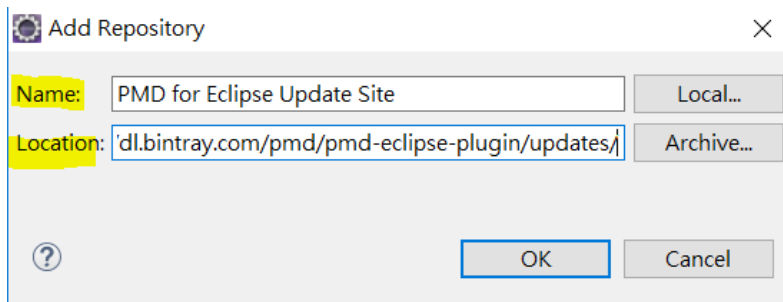
name	instruction	branch	complexity	line	method	class
main.java.example	M: 0 C: 22 100% <div></div>	M: 0 C: 0 0%	M: 0 C: 6 100% <div></div>	M: 0 C: 10 100% <div></div>	M: 0 C: 6 100% <div></div>	M: 0 C: 2 100% <div></div>

Coverage Breakdown by Source File

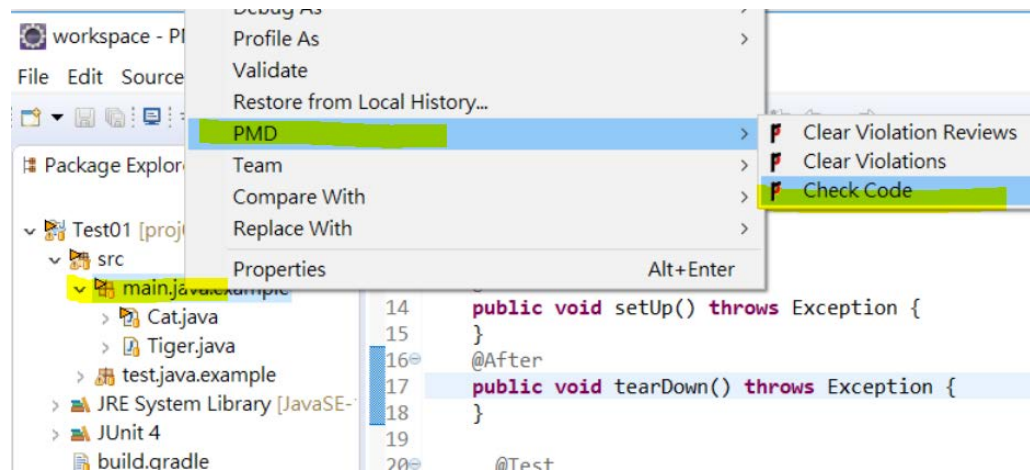
name	instruction	branch	complexity	line	method	class
Cat	M: 0 C: 11 100% <div></div>	M: 0 C: 0 0%	M: 0 C: 3 100% <div></div>	M: 0 C: 5 100% <div></div>	M: 0 C: 3 100% <div></div>	M: 0 C: 1 100% <div></div>
Tiger	M: 0 C: 11 100% <div></div>	M: 0 C: 0 0%	M: 0 C: 3 100% <div></div>	M: 0 C: 5 100% <div></div>	M: 0 C: 3 100% <div></div>	M: 0 C: 1 100% <div></div>

靜態程式碼分析 – PMD

- ❑ Help – > Install New Software...
 - **Name:** PMD for Eclipse Update Site
 - <https://dl.bintray.com/pmd/pmd-eclipse-plugin/updates/>



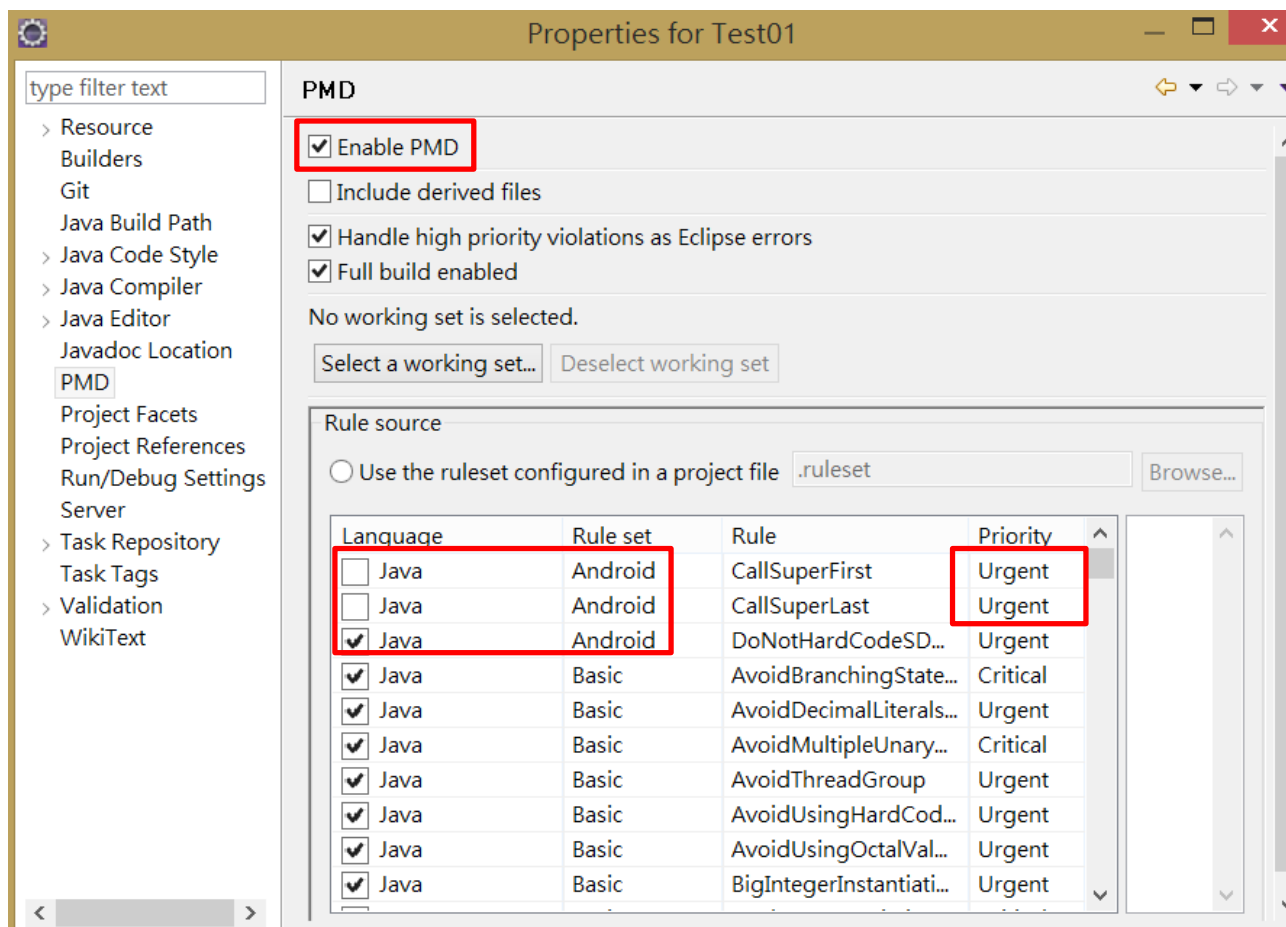
靜態程式碼分析 – PMD



Violations Overview				
Element	# Violations	# Violation...	# Violation...	Project
main.java.example	13	1625.0	3.25	Test01
Tiger.java	6	1500.0	3.00	Test01
CommentRequired	3	750.0	1.50	Test01
AtLeastOneConstructor	1	250.0	0.50	Test01
SystemPrintln	2	500.0	1.00	Test01
Cat.java	7	1750.0	3.50	Test01
ShortClassName	1	250.0	0.50	Test01
CommentRequired	3	750.0	1.50	Test01
AtLeastOneConstructor	1	250.0	0.50	Test01
SystemPrintln	2	500.0	1.00	Test01

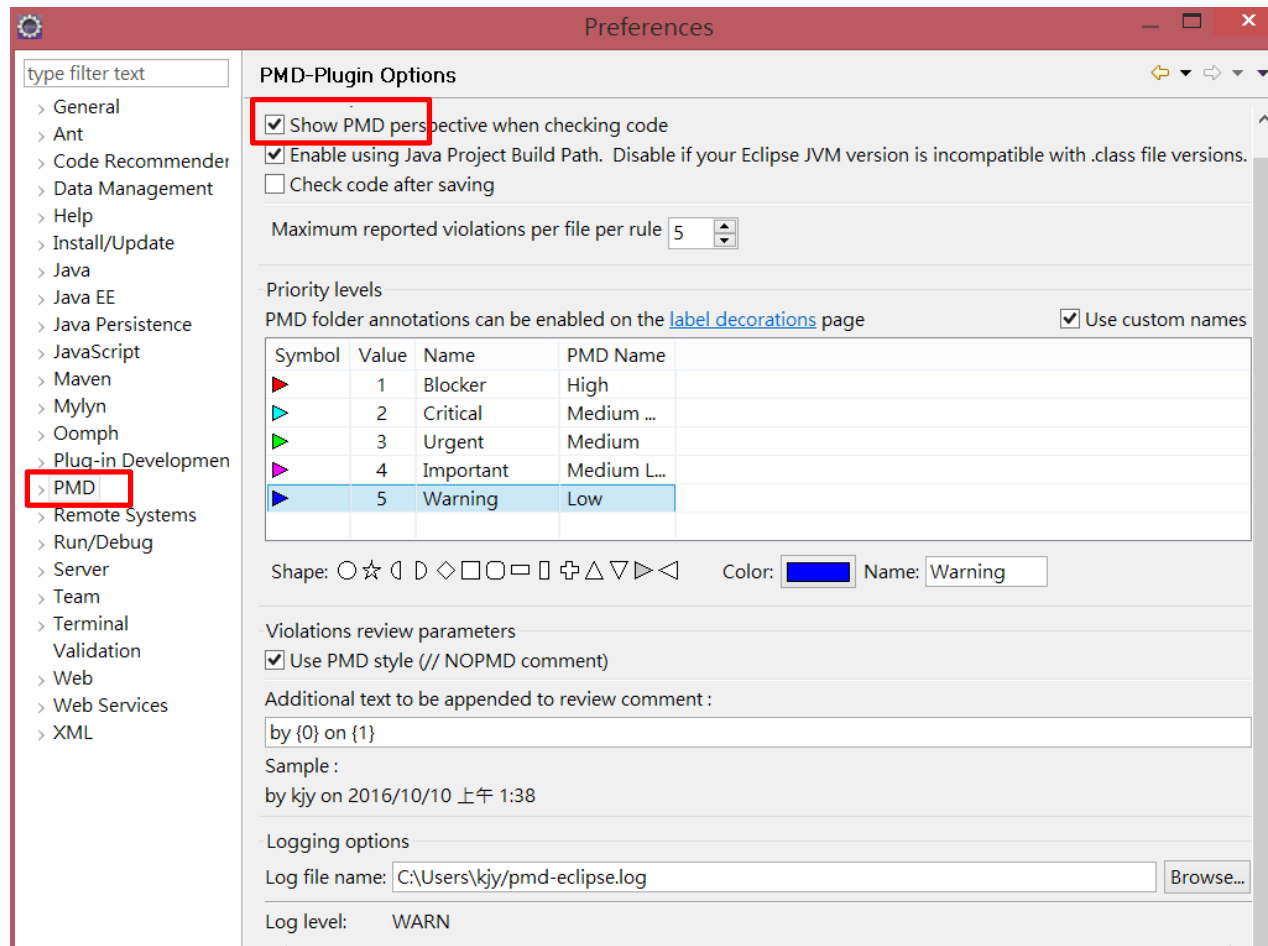
靜態程式碼分析 – PMD

- ❑ 點選Project名稱，按右鍵，選擇 Properties，設定Rules



靜態程式碼分析 – PMD

❑ Window – Preferences



靜態程式碼分析 – Find Bugs

❑ Help - Eclipse Marketplace... - Findbug

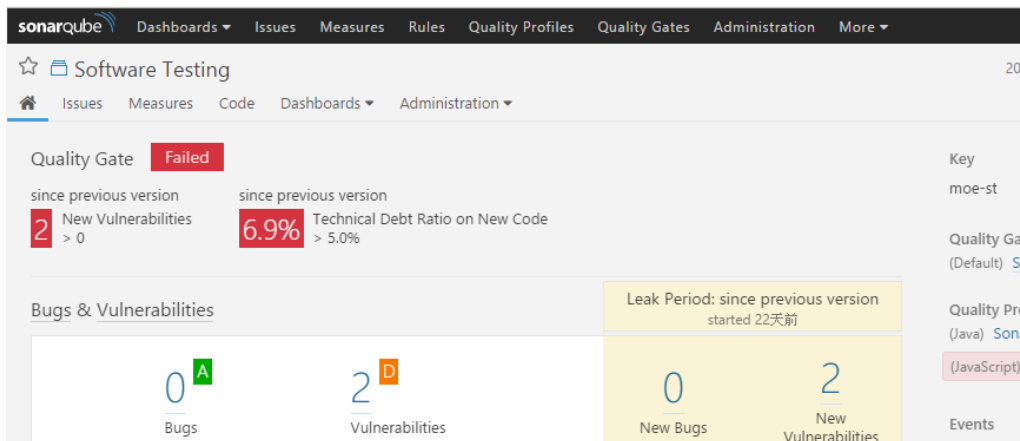
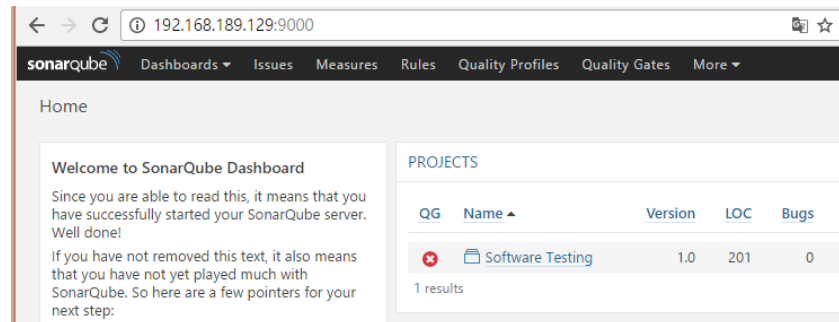
- 右鍵 – Find Bugs

The screenshot shows the Eclipse IDE interface. At the top, the 'Eclipse Marketplace' window is open, displaying the 'FindBugs Eclipse Plugin 3.0.2'. The search bar contains 'findbug', and the 'Install Pending' button is highlighted. Below the marketplace window, the 'Find Bugs' menu is open, showing options like 'Find Bugs' and 'Clear Bug Markers'. The 'Markers' tab is selected in the bottom panel, showing a table of errors and warnings.

Description	Resource	Path	Location	Type
0 errors, 1 warning, 0 others				
Java Problems (1 item)				
The import org.junit.Assert is never used	TestCat.java	/Test01/src/test/j...	line 3	Java Problem

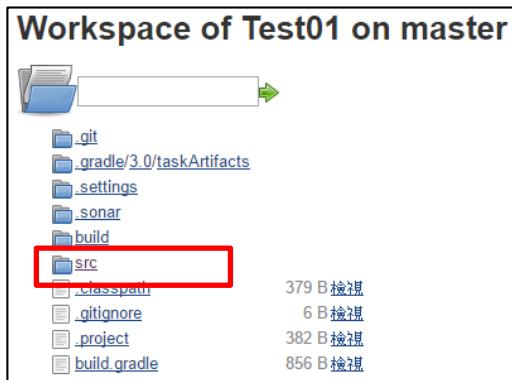
Sonarqube品質管理

- ❑ 啟動sonarqube，sudo docker start sonarqube
 - 查看sonarqube docker是否啟動，sudo docker ps
 - 192.168.x.x:9000，登入 admin/admin，Software Testing



Jenkins 查看 Git 工作區 Code

- ❑ 點選 Test01，或回到專案；點選工作目錄
- ❑ 點選src、example；有ShoppingCart.java



- ❑ 點選ShoppingCart.java，可以看到Code

Eclipse新增程式

- 增加程式 production code，在main.java.example，右鍵new
 - 購物車 ShoppingCart 類別
 - 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。
 - 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。

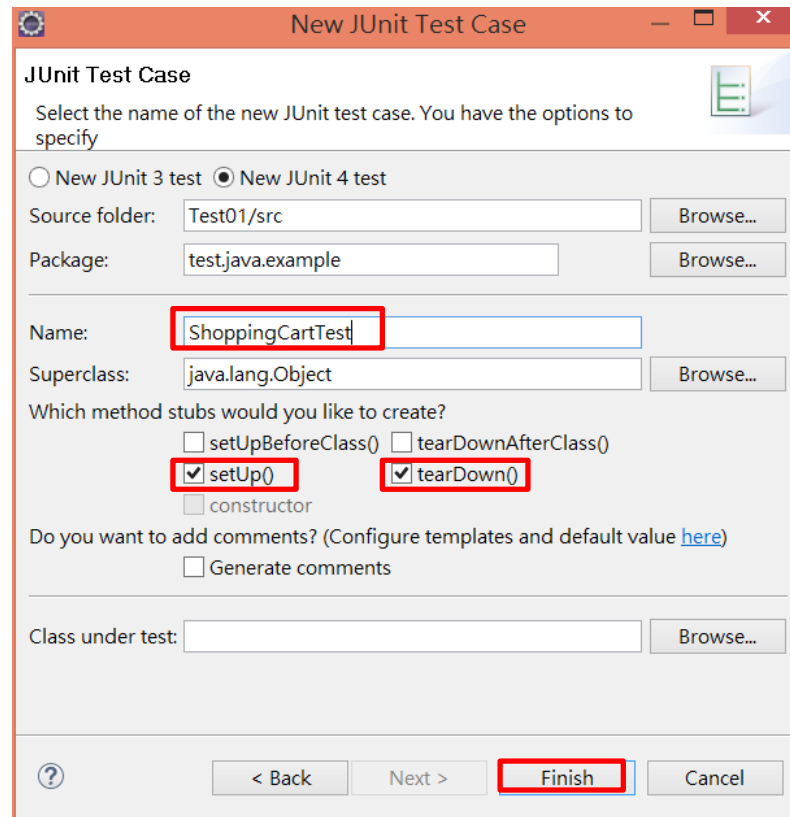
Eclipse新增程式

```
package main.java.example;
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;
    }
    public void setVip(int vip, int vipDiscount) {
        this.vip = vip;
        this.vipDiscount = vipDiscount;
    }
}
```

```
    public void setMass(int mass, int massDiscount) {
        this.mass = mass;
        this.massDiscount = massDiscount;
    }
    public void setCost(int cost) {
        this.cost = cost;
    }
    public void computeCost() {
        newCost = cost;
        if (vip==1) {
            newCost = newCost*vipDiscount;
        }
        if (cost>=mass) {
            newCost = newCost*massDiscount;
        }
    }
    public int getCost() {
        return newCost;
    }
}
```

Eclipse新增程式

- ❑ 新增 test Driver，測試計算總價方法，test.java.example 右鍵
 - New – Other... – Java – JUnit – JUnit Test Case [next]
 - Name: ShoppingCartTest [Finish]



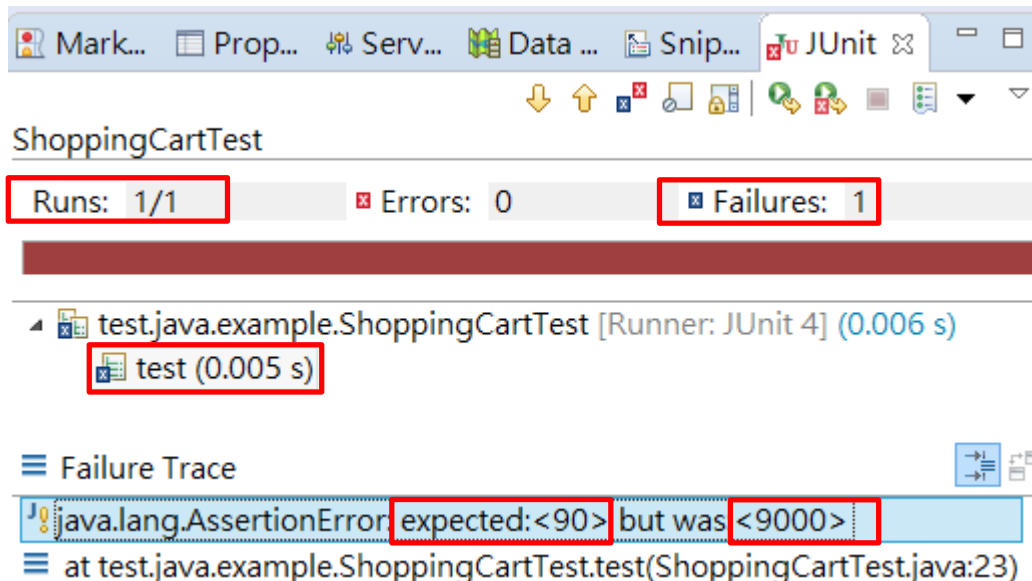
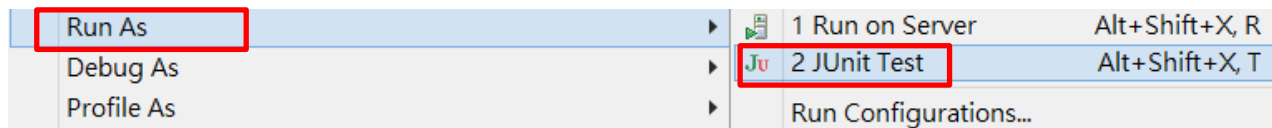
Eclipse新增程式

```
package test.java.example;
import static org.junit.Assert.*;
import org.junit.After;
import org.junit.Before;
import org.junit.Test;
import main.java.example.*;
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());
}
}
```

Eclipse新增程式

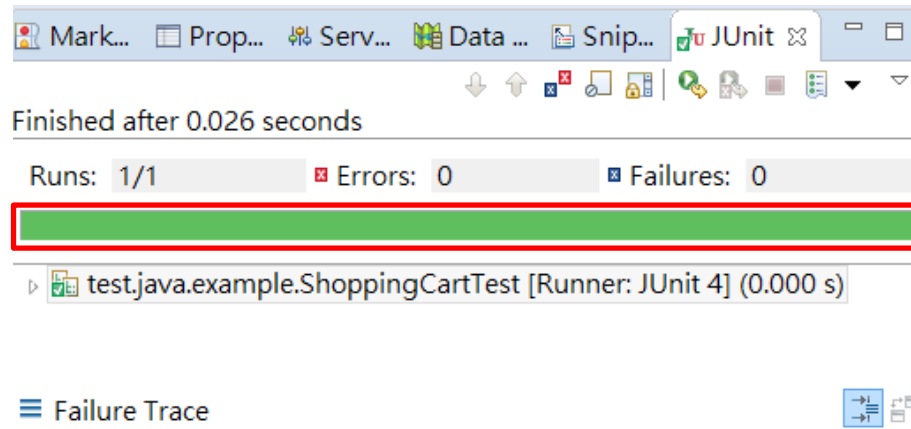
❑ 執行測試，ShoppingCartTest 右鍵



Eclipse改修程式

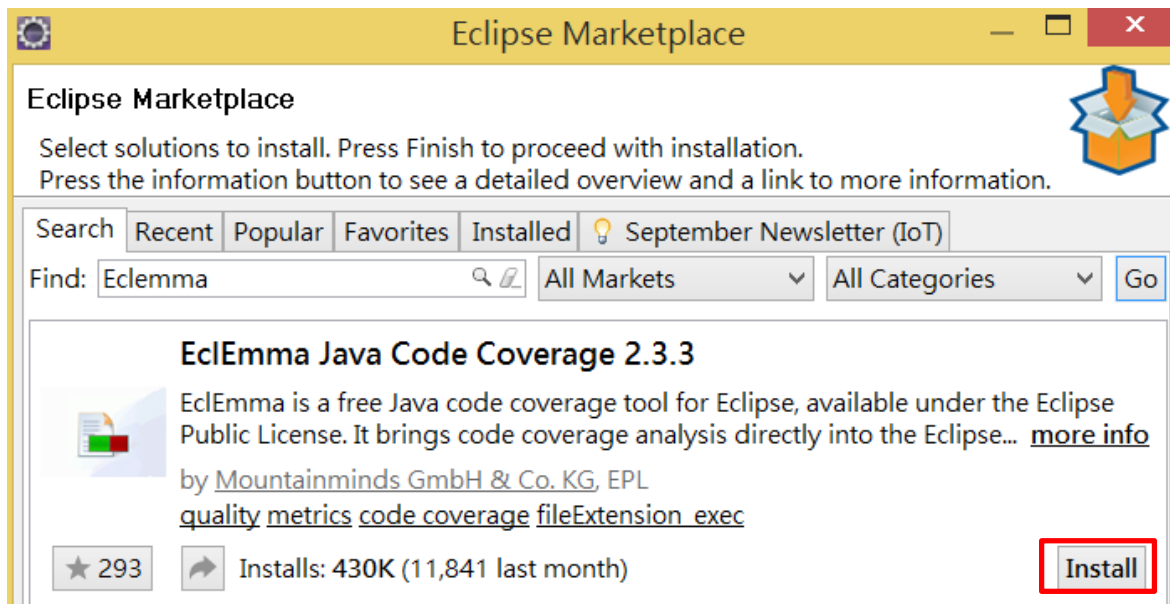
❑ 修改程式，重新測試，變成綠色

```
public void computeCost() {  
    newCost = cost;  
    if (vip==1) {  
        newCost = (newCost*vipDiscount)/100;  
    }  
    if (cost>=mass) {  
        newCost = (newCost*massDiscount)/100;  
    }  
}
```



Eclipse測試涵蓋度Coverage

- ❑ 安裝Eclipse plugin，Eclemma
 - Help – Eclipse Marketplace
 - Find: Eclemma [Go][Install][Accept]
 - 重新啟動Eclipse



Eclipse測試涵蓋度Coverage

- 重新測試，查看 Coverage；右鍵，點選 Coverage As

The screenshot shows the Eclipse IDE interface. At the top, the 'Coverage As' dropdown is set to '1 JUnit Test'. Below it, the 'Validate' button is visible. The code editor displays a Java class with the following code:

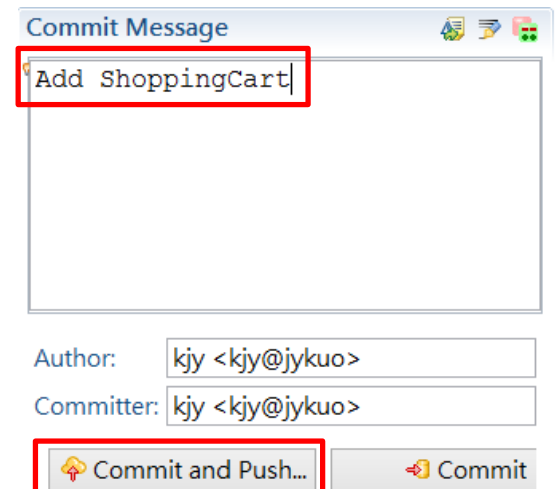
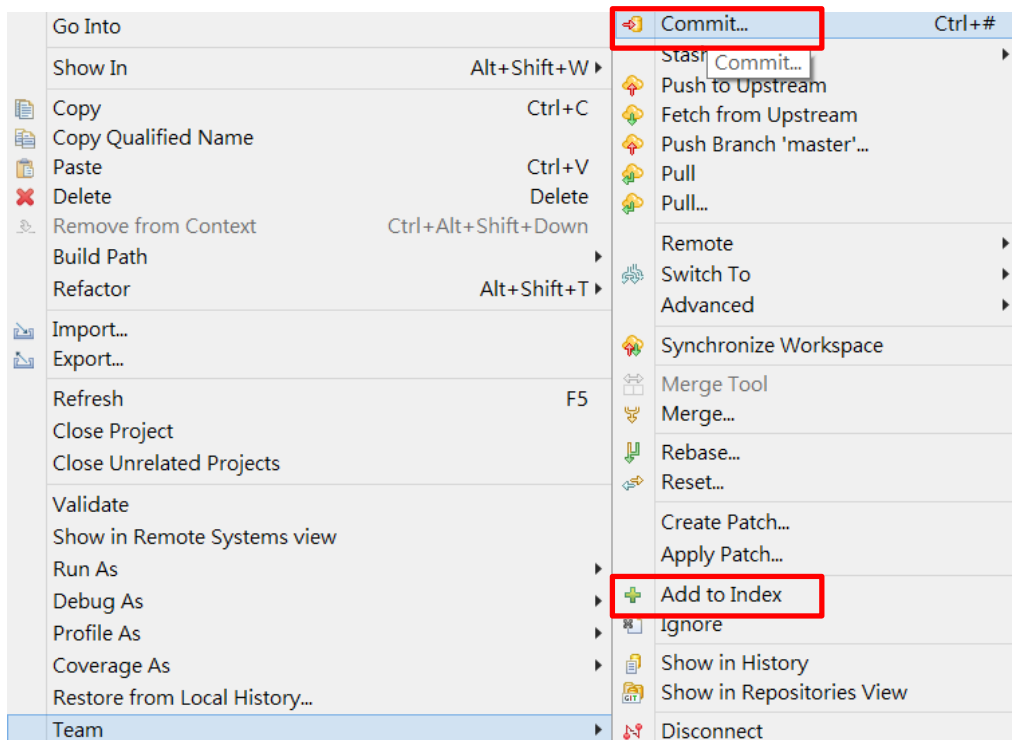
```
28 if (vip==1) {
29     newCost = (newCost*vipDiscount)/100;
30 }
31 if (cost>=mass)
32     newCost = (newCost*massDiscount)/100;
33 }
34 }
35 public int getCost() {
36     return newCost;
37 }
38 }
```

The 'Coverage' view at the bottom shows the following table:

Element	Coverage	Covered Instr...	Missed Instru...	Total Instructi...
Test01	63.4 %	102	59	161
src	63.4 %	102	59	161
main.java.example	67.7 %	65	31	96
Cat.java	0.0 %	0	11	11
Tiger.java	0.0 %	0	11	11
ShoppingCart.java	87.8 %	65	9	74
ShoppingCart	87.8 %	65	9	74
computeCost()	71.9 %	23	9	32
ShoppingCart()	100.0 %	21	0	21
getCost()	100.0 %	3	0	3
setCost(int)	100.0 %	4	0	4
setMass(int, int)	100.0 %	7	0	7
setVip(int, int)	100.0 %	7	0	7

Eclipse 簽入修改程式碼

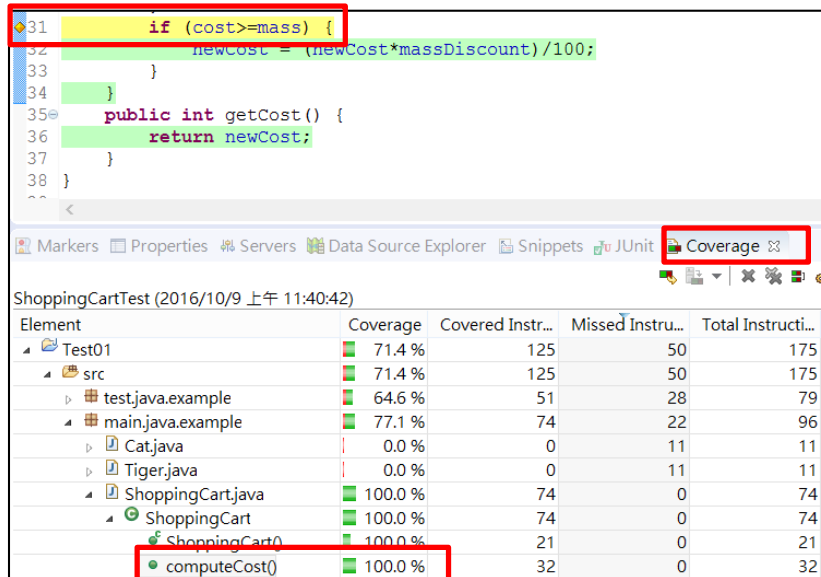
- ❑ 加入暫存區、簽入本地儲存區、遠端儲存區
 - 輸入密碼、[ok]



Eclipse改修測試程式

- ❑ 新增 test ， ShoppingCartTest ， 增加Code Coverage
- ❑ ShoppingCartTest.java
 - 右鍵 ， Run As-JUnit Test
 - 通過 ， 綠色
 - 右鍵 ， Coverage As-JUnit Test

```
public void test() {  
    myCart.setCost(100);  
    myCart.setMass(100, 90);  
    myCart.setVip(0, 100);  
    myCart.computeCost();  
    assertEquals(90, myCart.getCost());  
    myCart.setVip(1, 90);  
    myCart.computeCost();  
    assertEquals(81, myCart.getCost());  
}
```



Element	Coverage	Covered Instr...	Missed Instru...	Total Instructi...
Test01	71.4 %	125	50	175
src	71.4 %	125	50	175
test.java.example	64.6 %	51	28	79
main.java.example	77.1 %	74	22	96
Cat.java	0.0 %	0	11	11
Tiger.java	0.0 %	0	11	11
ShoppingCart.java	100.0 %	74	0	74
ShoppingCart	100.0 %	74	0	74
ShoppingCart()	100.0 %	21	0	21
computeCost()	100.0 %	32	0	32

Eclipse改修測試程式

❑ 新增 test ， ShoppingCartTest ， 增加Code Coverage

❑ ShoppingCartTest.java

- 右鍵 ， Run As-Junit Test
 - 通過 ， 綠色
- 右鍵 ， Coverage As-Junit Test

The screenshot shows the Eclipse IDE with the ShoppingCartTest.java file open. The code is as follows:

```
26 public void computeCost() {
27     newCost = cost;
28     if (vip==1) {
29         newCost = (newCost*vipDiscount)/100;
30     }
31     if (cost>=mass) {
32         newCost = (newCost*massDiscount)/100;
33     }
34 }
35 public int getCost() {
36     return newCost;
37 }
38 }
```

The code is highlighted with a red box. Below the code, the Coverage report is displayed for the test run on 2016/10/9 at 11:48:07. The report shows the following data:

Element	Coverage	Covered Instr...	Missed Instru...	Total Instructi...
Test01	73.5 %	139	50	189
src	73.5 %	139	50	189
test.java.example	69.9 %	65	28	93
main.java.example	77.1 %	74	22	96
Cat.java	0.0 %	0	11	11
Tiger.java	0.0 %	0	11	11
ShoppingCart.java	100.0 %	74	0	74
ShoppingCart	100.0 %	74	0	74
ShoppingCart	100.0 %	21	0	21
computeCost()	100.0 %	32	0	32

```
public void test() {
    myCart.setCost(100);
    myCart.setMass(100, 90);
    myCart.setVip(0, 100);
    myCart.computeCost();
    assertEquals(90,myCart.getCost());
    myCart.setVip(1, 90);
    myCart.computeCost();
    assertEquals(81,myCart.getCost());
    myCart.setMass(200, 90);
    myCart.computeCost();
    assertEquals(90,myCart.getCost());
}
```