

개발도구와 단위 테스트

허광남

kenu@okjsp.net

개발도구와 테스트 자동화

1. 개발도구에서 지원하는 테스트케이스
2. JUnit 이해하기
3. 테스트케이스와 리팩토링

개발도구에서 지원하는 테스트케이스

- 테스트케이스는 입력값과 출력값을 지정해서 테스트 되는 함수가 그 조건을 충족하는지 여부를 확인 가능한 프로그램
- assertEquals(예상출력, 함수(입력));

테스트의 자동화

- 테스트케이스는 프로그램으로 자동화해야 한다.
- 품질은 테스트의 범위와 실행 횟수에 영향을 받는다.
- 수작업의 테스트는 기한 내 테스트 실행 횟수를 감소시킨다.
- 테스트는 빨리 쉽게 수행 가능해야 한다.

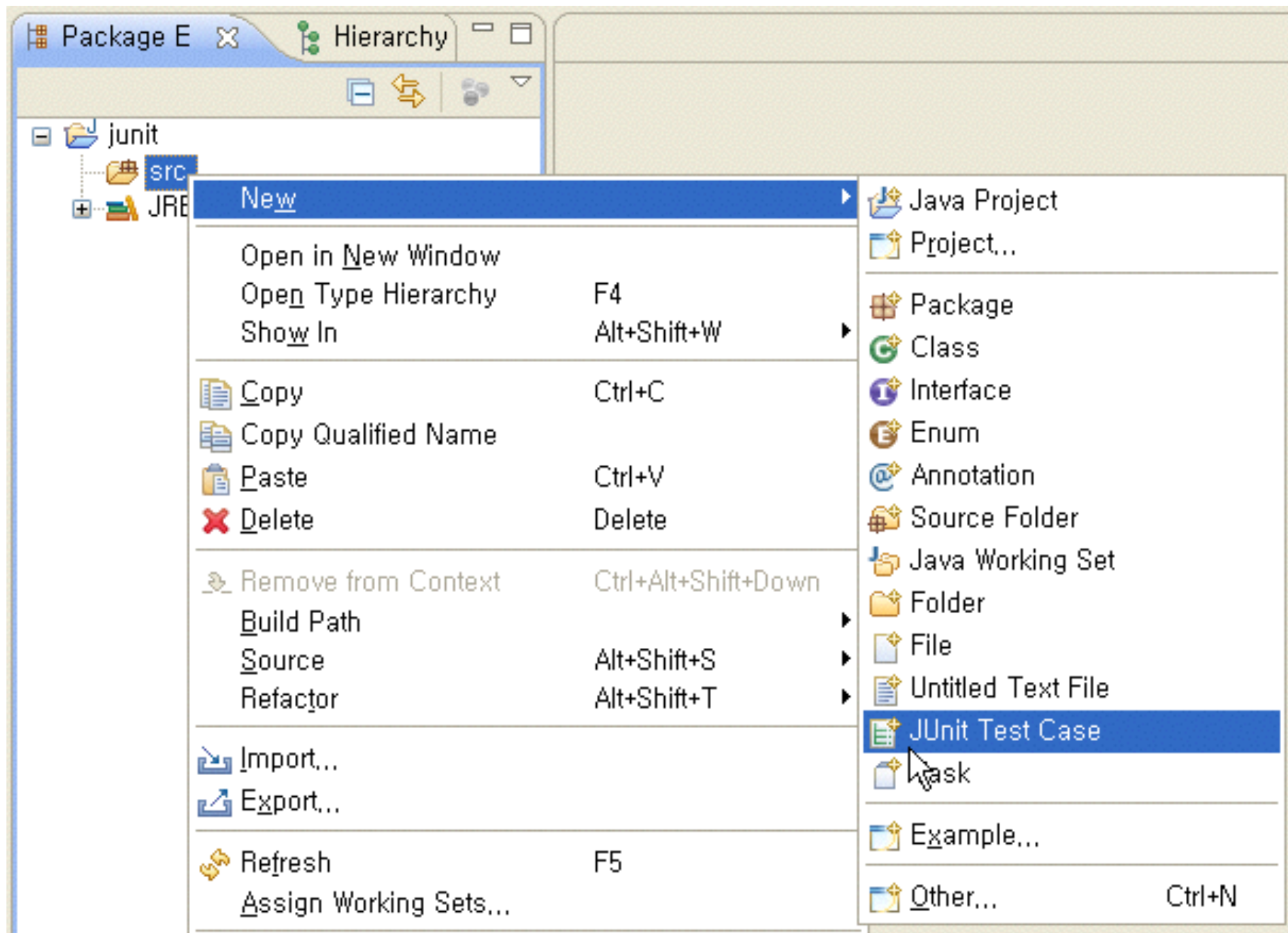
Test Harness

- Scaffold
- Outside of Application
- Fine grained TestCases
- It's useful for changing software
- Acrobatic with safety net





TestCase 제작



TestCase 제작

New JUnit Test Case

JUnit Test Case
⚠ Superclass does not exist.

☒ New JUnit 3 test ☐ New JUnit 4 test

Source folder:

Package:

Name:

Superclass:

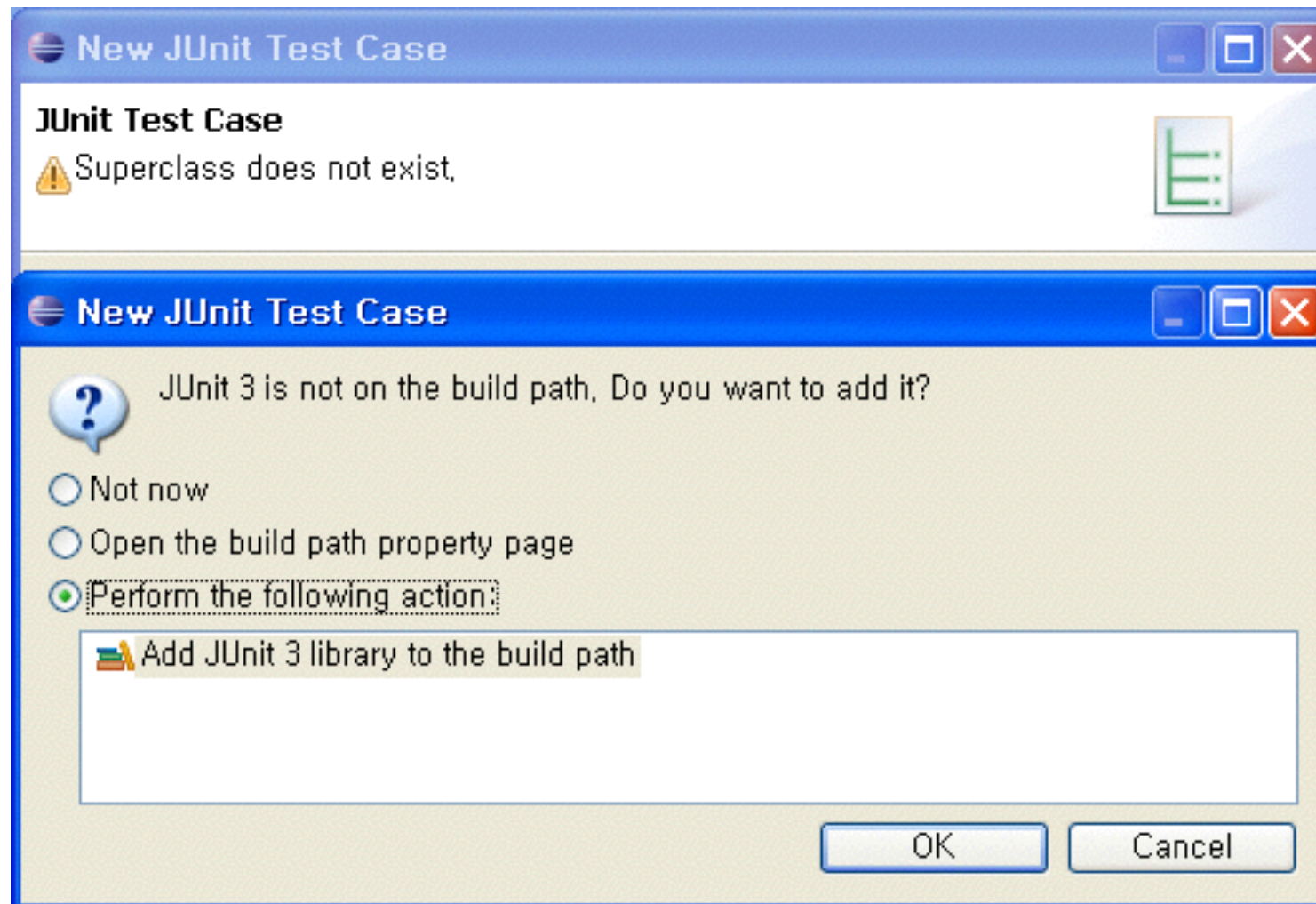
Which method stubs would you like to create?

☐ setUpBeforeClass() ☐ tearDownAfterClass()
☐ setUp() ☐ tearDown()
☐ constructor

Do you want to add comments? (Configure templates and default value [here](#))
☐ Generate comments

Class under test:

TestCase 제작



TestCase 제작

```
package net.okjsp;  
import junit.framework.TestCase;  
public class HelloWorldTest extends  
    TestCase {  
}
```


테스트 메소드

A screenshot of a Java IDE window titled 'HelloWorldTest.java'. The code is written in Java and uses syntax highlighting. It defines a package 'net.okjsp', imports 'junit.framework.TestCase', and defines a public class 'HelloWorldTest' that extends 'TestCase'. Inside the class, there is a public void method 'testGetMessage()' which calls 'HelloWorld.getMessage()' and asserts that the result is equal to 'Hello World'.

```
package net.okjsp;

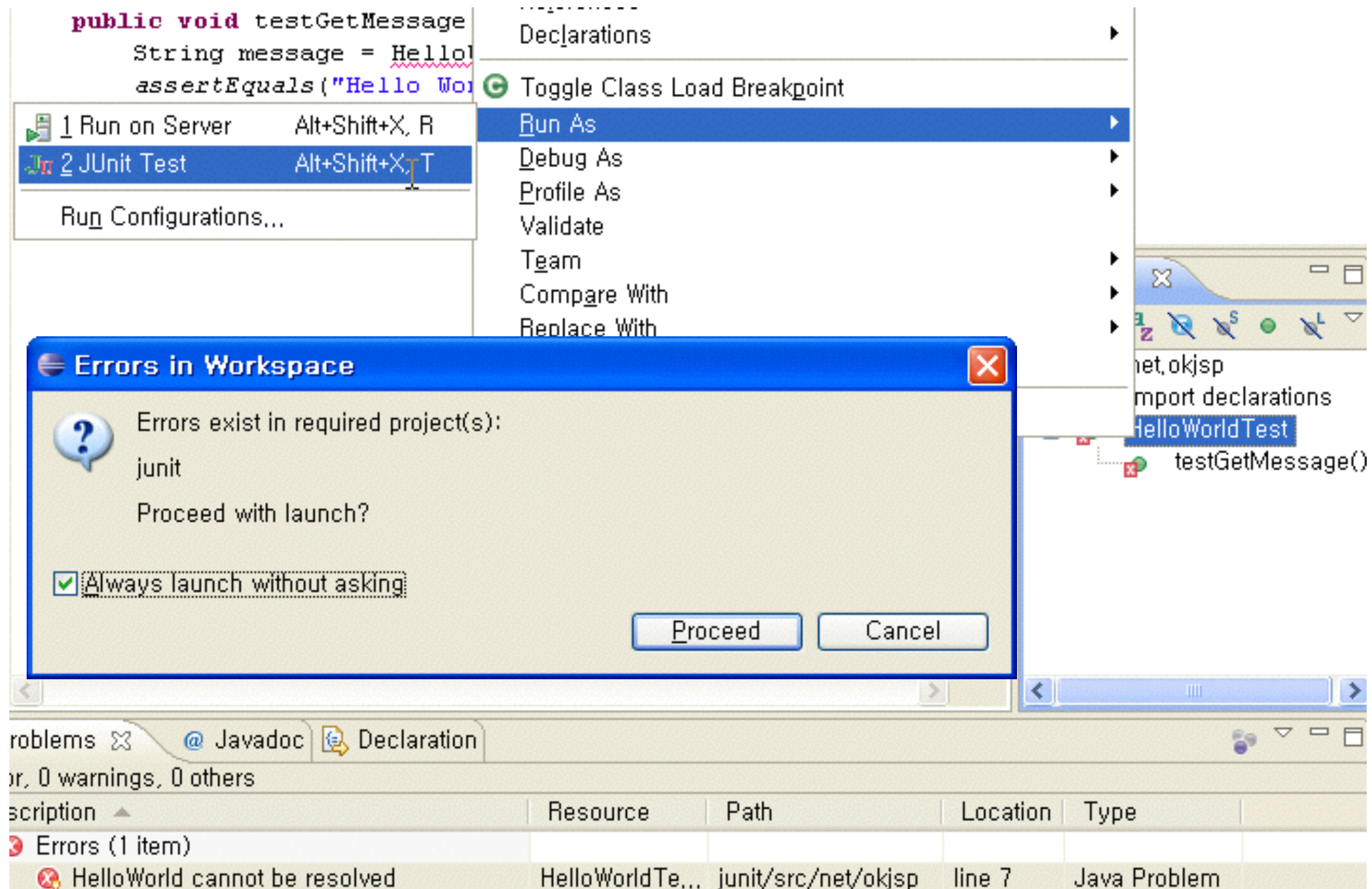
import junit.framework.TestCase;

public class HelloWorldTest extends TestCase {
    public void testGetMessage() {
        String message = HelloWorld.getMessage();
        assertEquals("Hello World", message);
    }
}
```

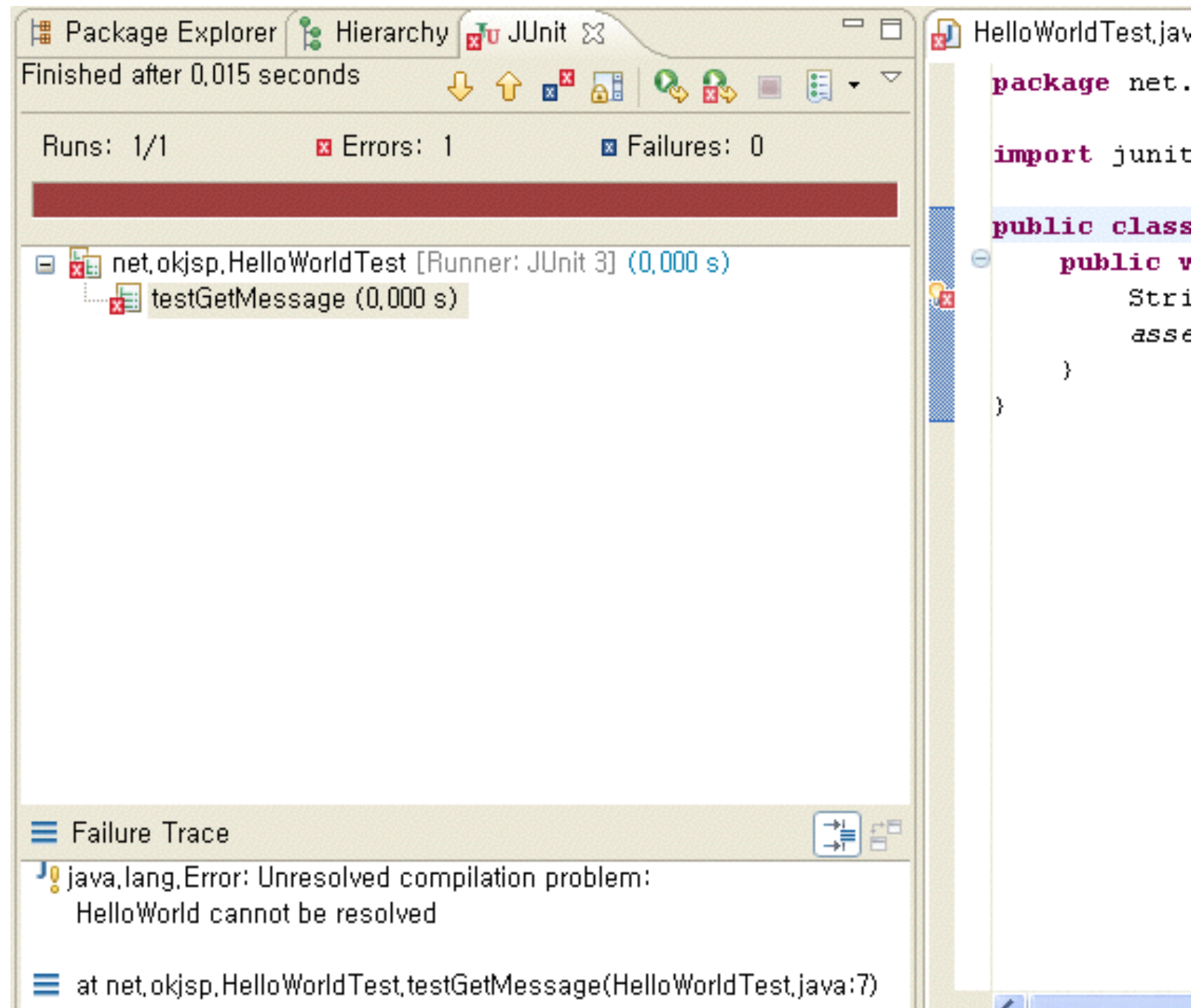
assert...

- assertTrue(실행결과);
- assertFalse(실행결과);
- assertEquals(목적값, 실행결과);
- assertEquals(float목적값, 실행결과, 오차범위);

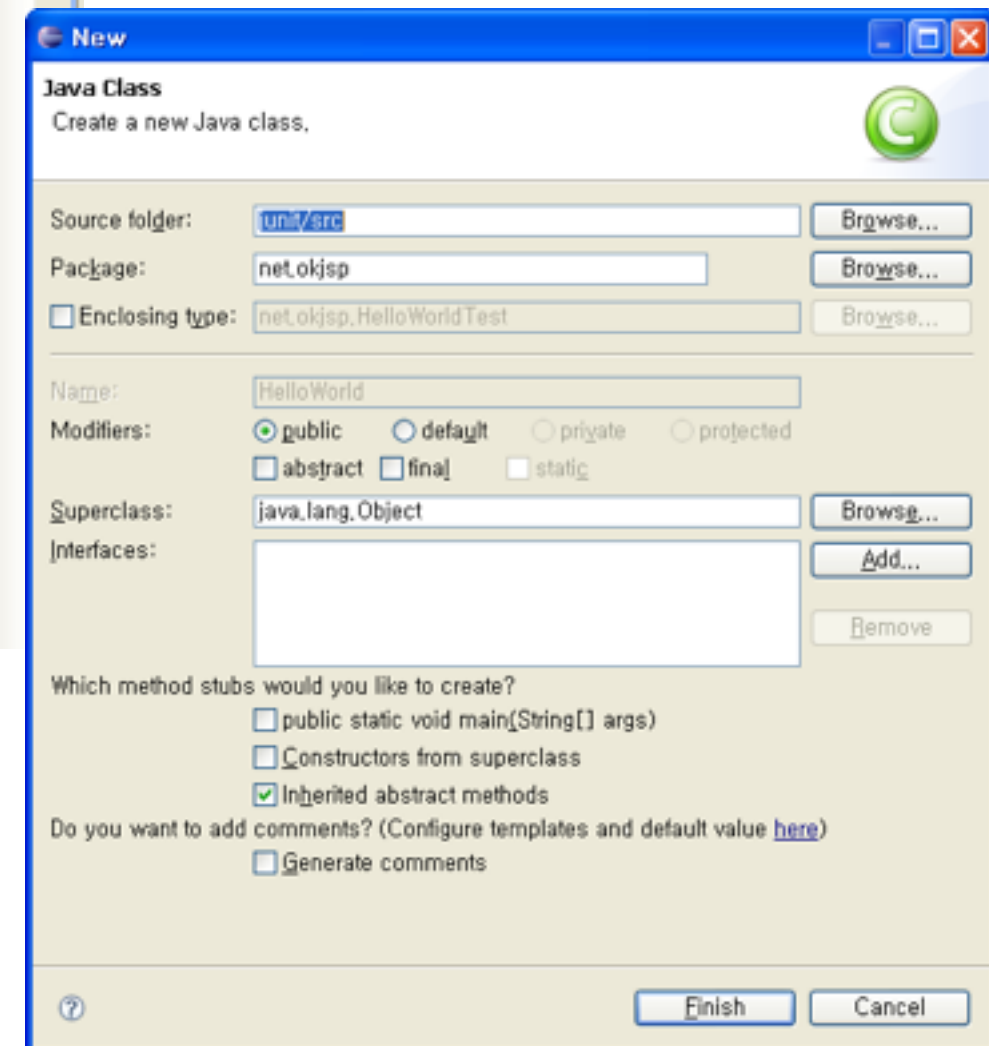
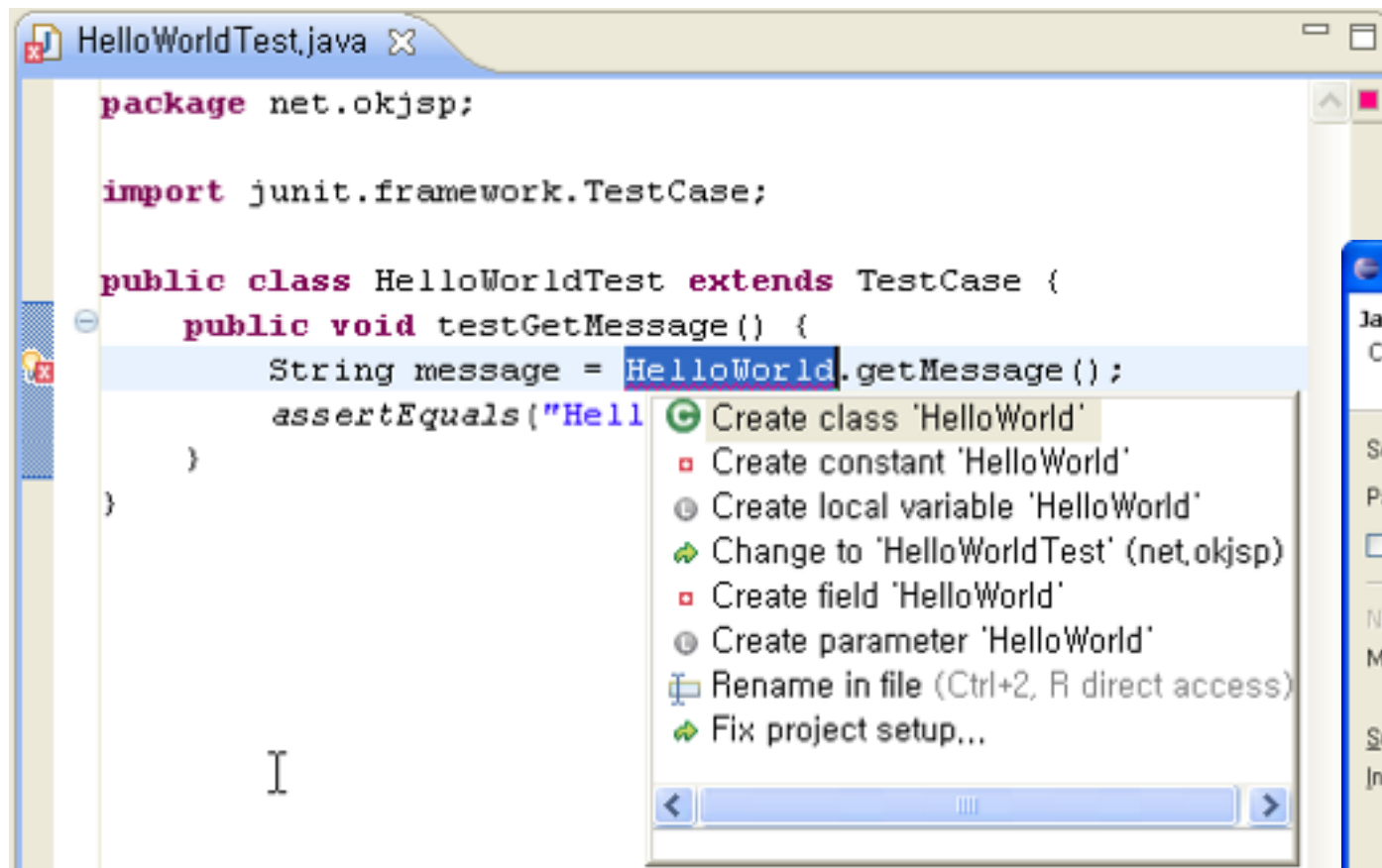
일단 테스트 고



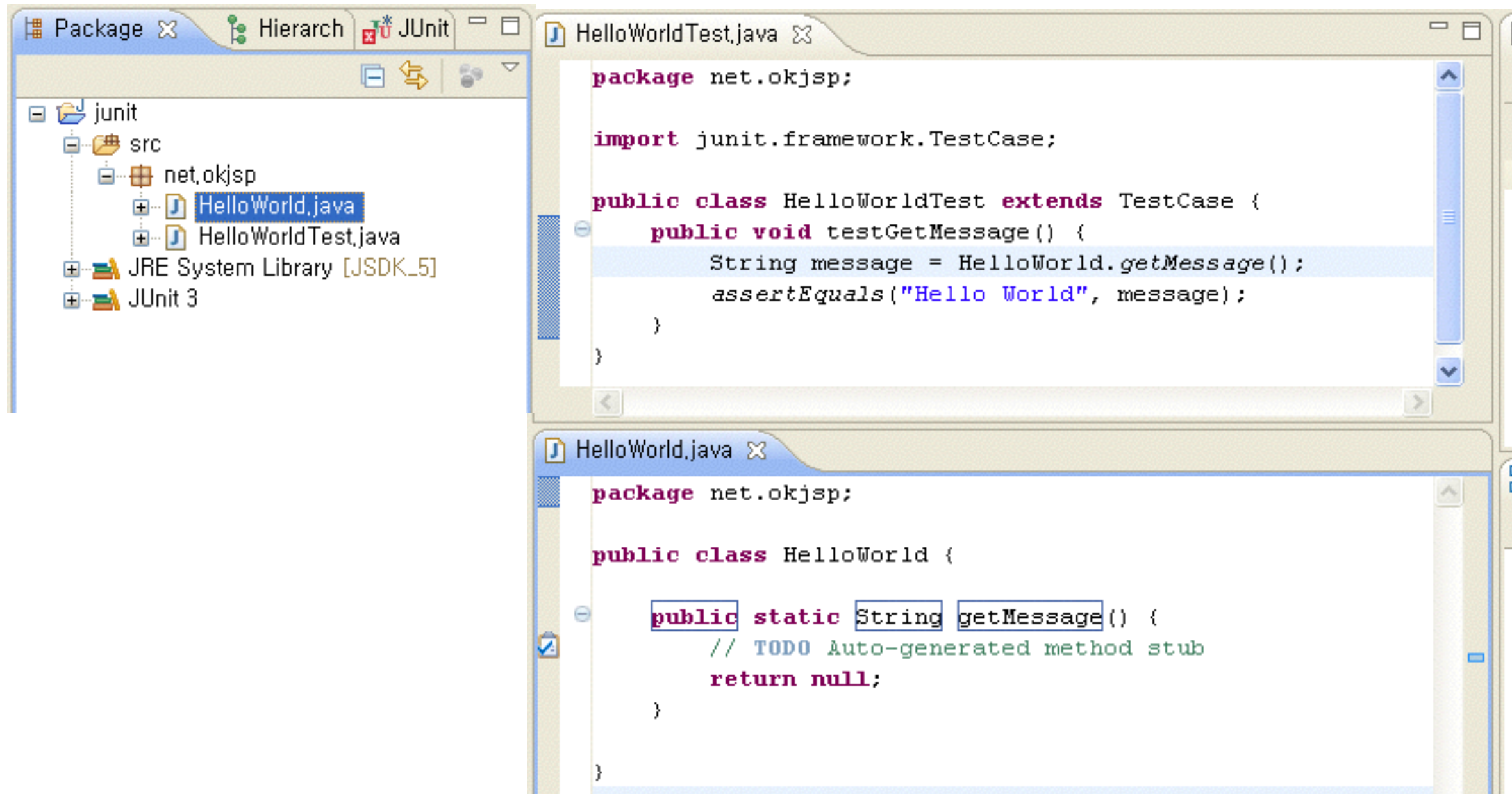
JUnit Result



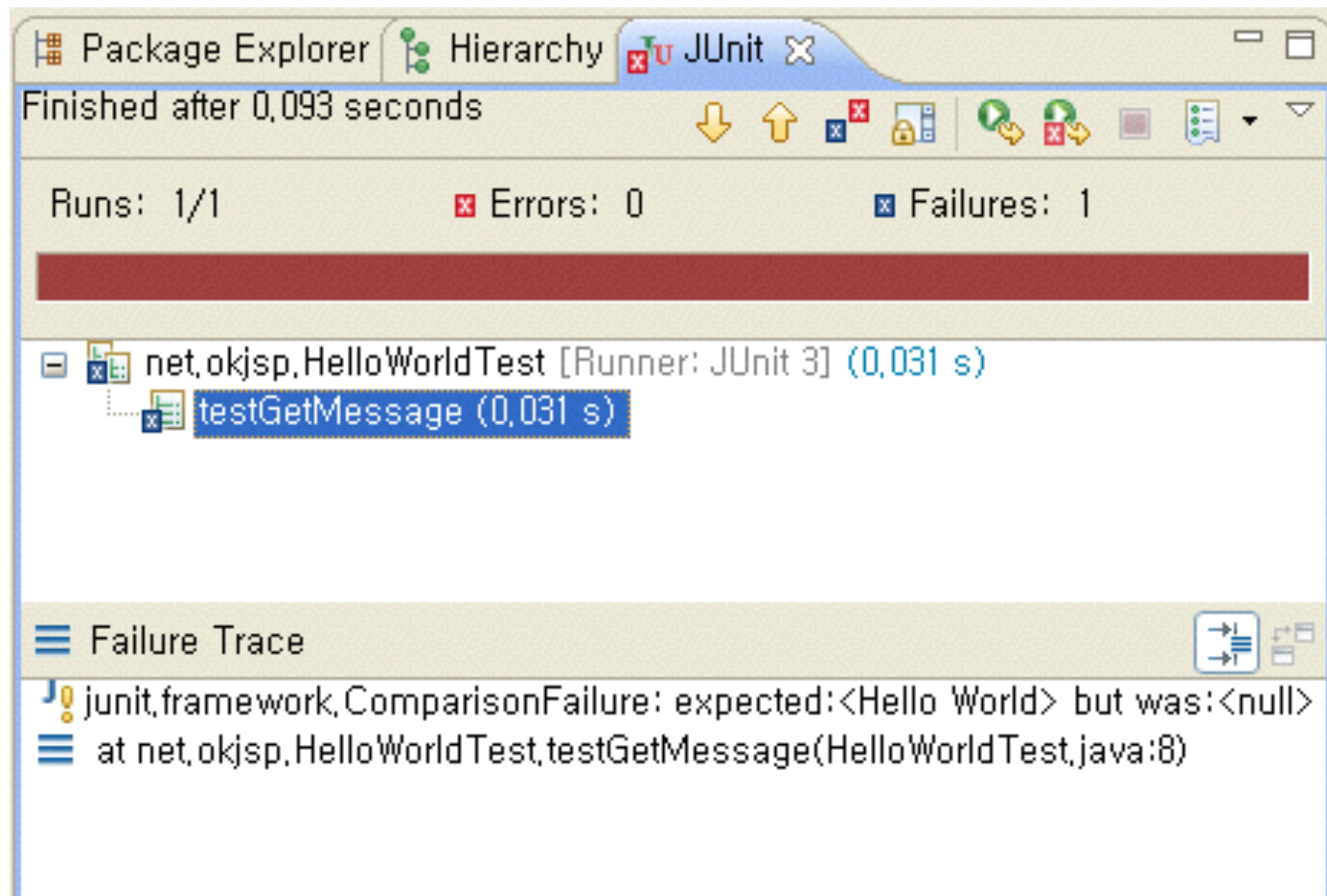
컴파일 되게 만들기



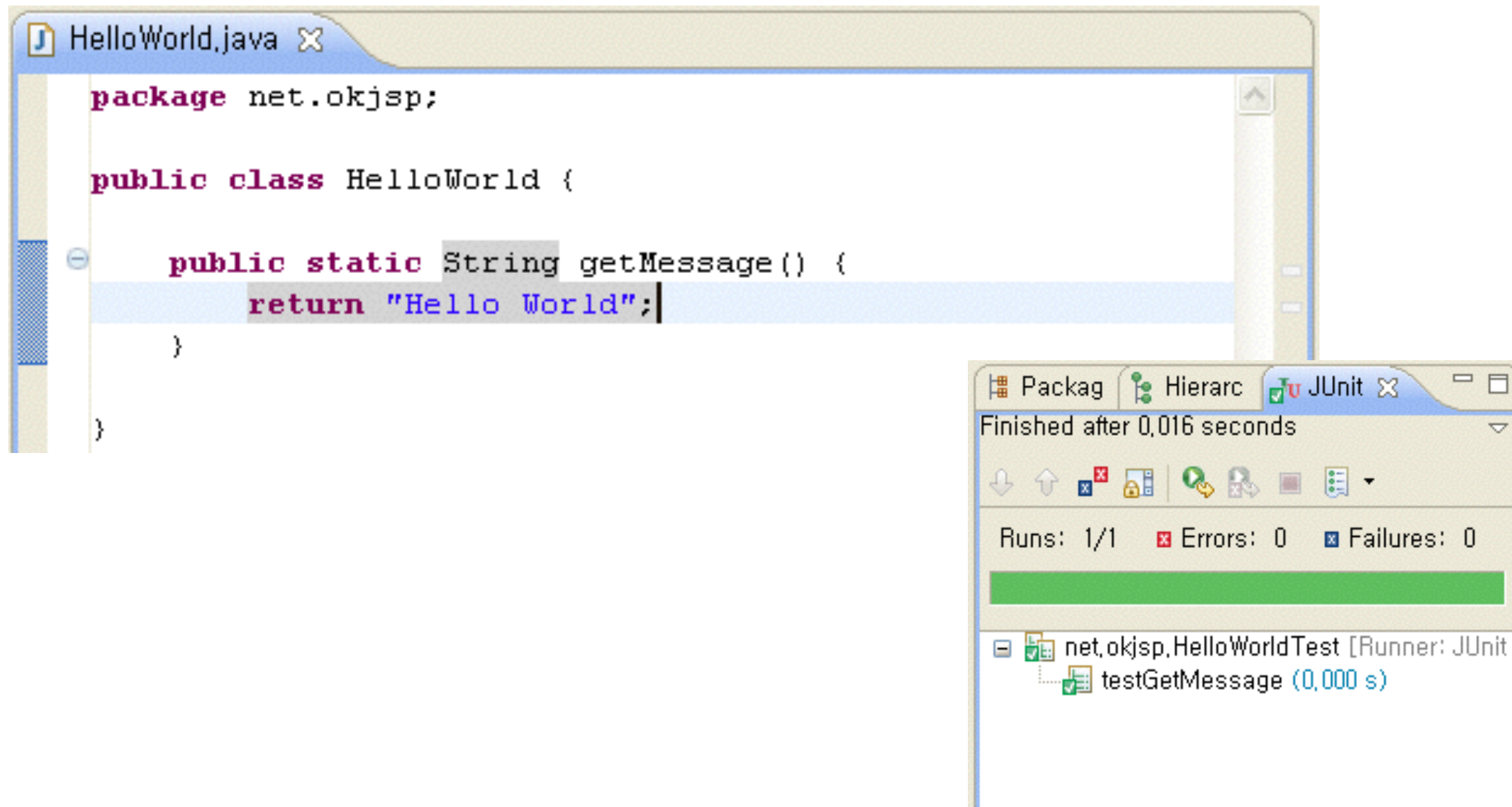
컴파일 되게 만들기



다시 테스트 고



테스트 통과하게 하기



기능 추가하기



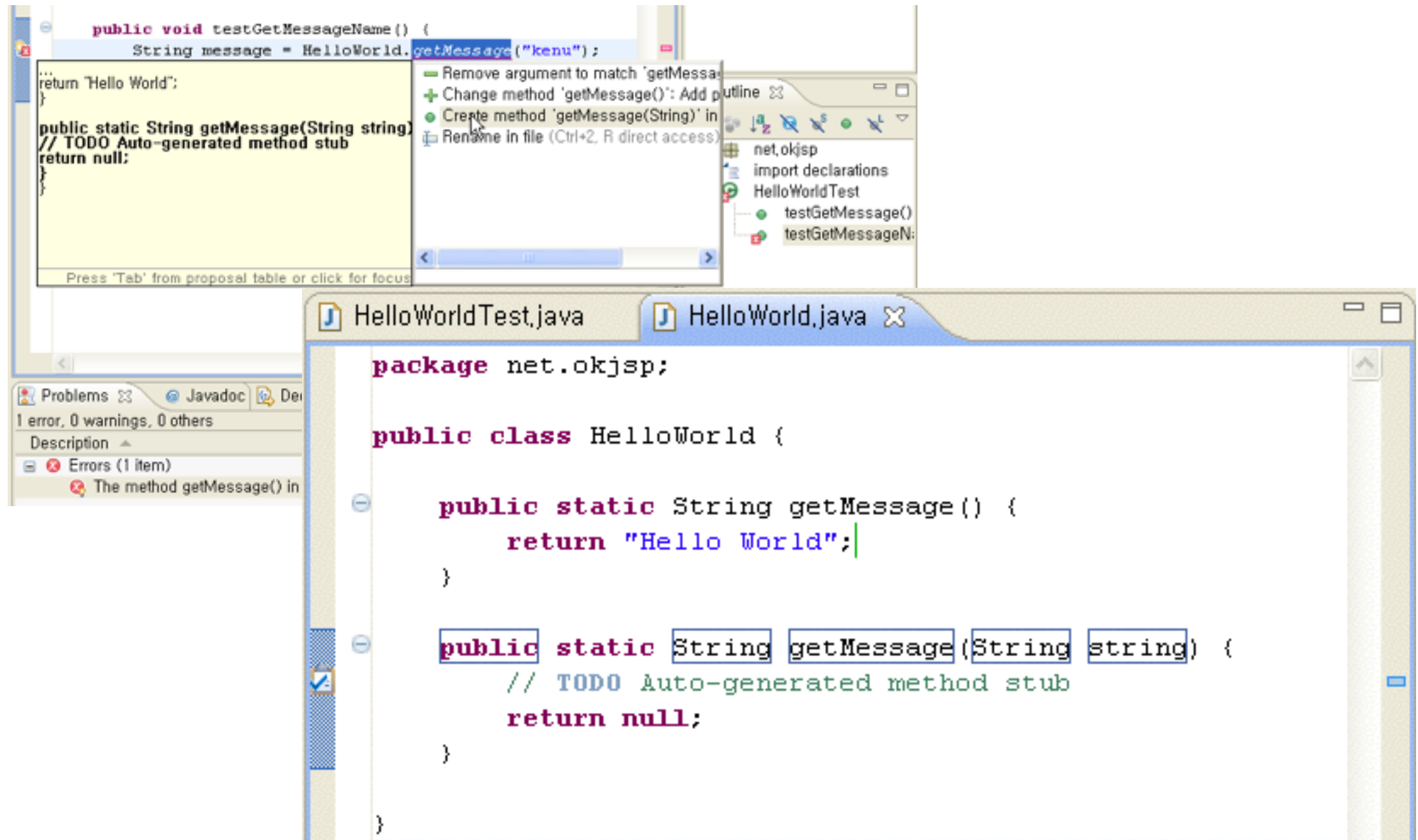
```
package net.okjsp;

import junit.framework.TestCase;

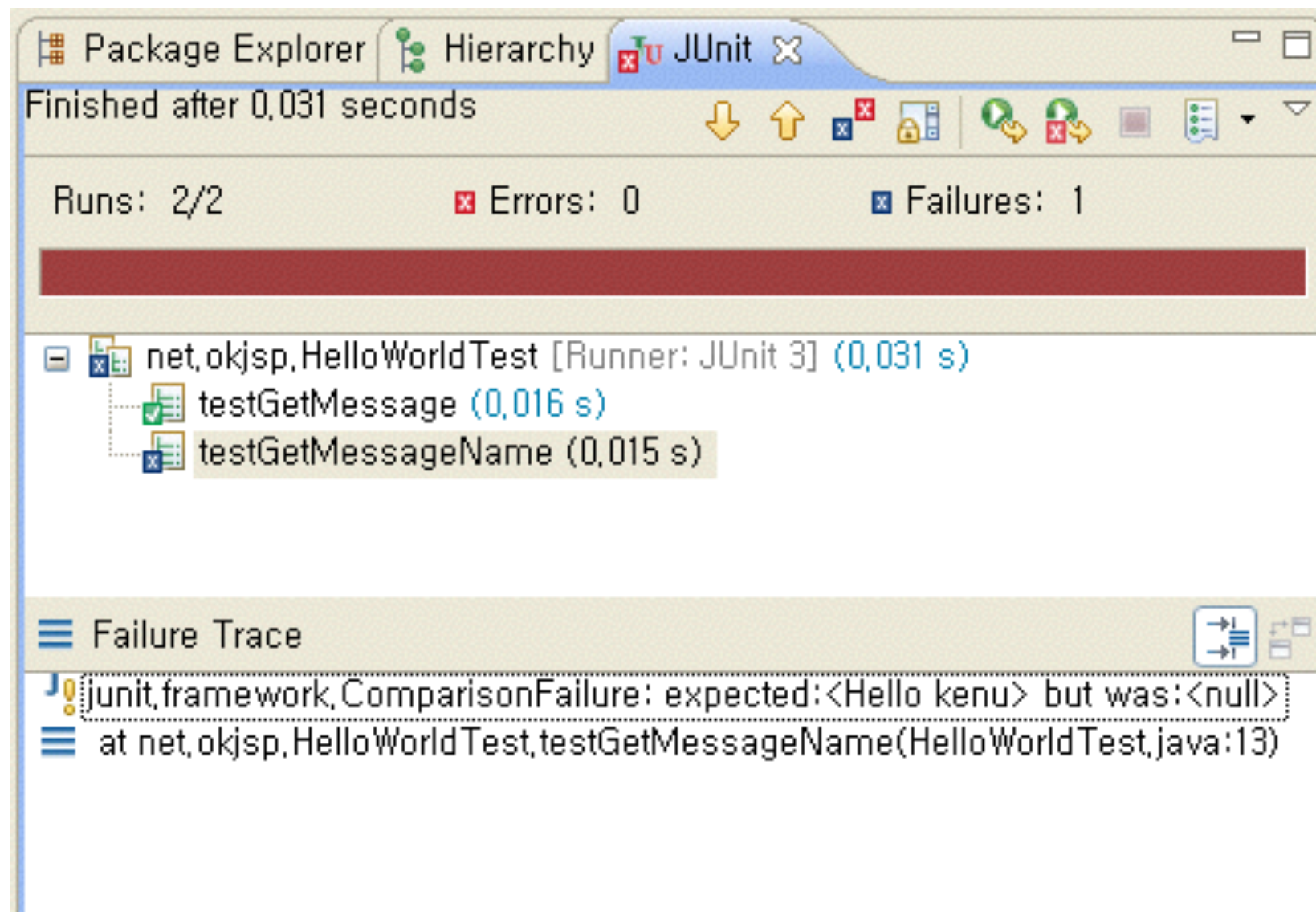
public class HelloWorldTest extends TestCase {
    public void testGetMessage() {
        String message = HelloWorld.getMessage();
        assertEquals("Hello World", message);
    }

    public void testGetMessageName() {
        String message = HelloWorld.getMessage("kenu");
        assertEquals("Hello kenu", message);
    }
}
```

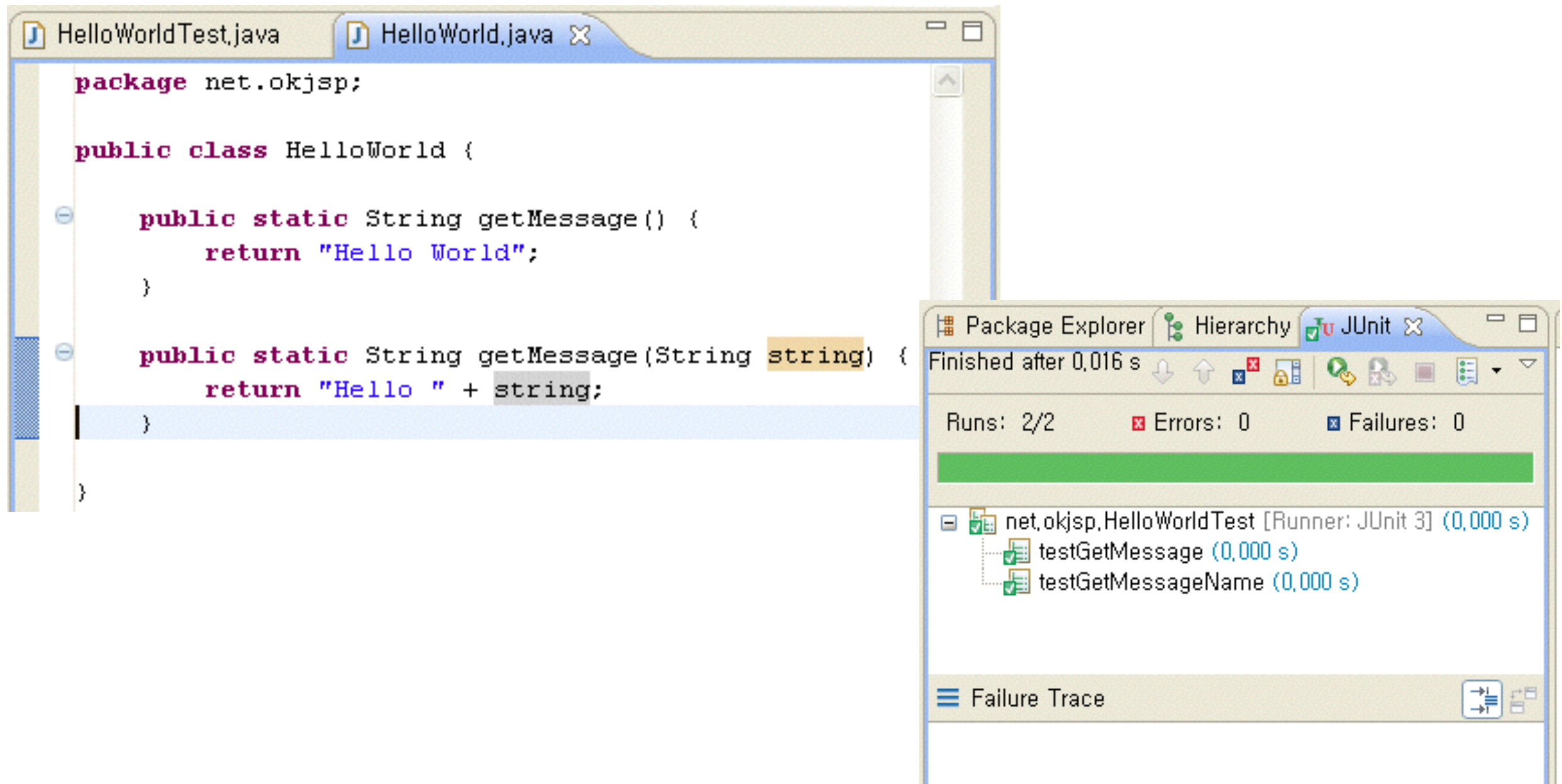

컴파일 되게 만들기



다시 테스트 고



테스트 통과하게 하기



The image shows an IDE window with two tabs: `HelloWorldTest.java` and `HelloWorld.java`. The `HelloWorldTest.java` tab is active, displaying the following Java code:

```
package net.okjsp;

public class HelloWorld {

    public static String getMessage() {
        return "Hello World";
    }

    public static String getMessage(String string) {
        return "Hello " + string;
    }
}
```

The `getMessage(String string)` method is highlighted. To the right, the `JUnit` test runner window is open, showing the results of the test run:

- Package Explorer: `net.okjsp.HelloWorldTest` [Runner: JUnit 3] (0,000 s)
- testGetMessage (0,000 s)
- testGetMessageName (0,000 s)

The test results indicate that the tests passed successfully, with 2/2 runs, 0 errors, and 0 failures.

리팩토링 하기

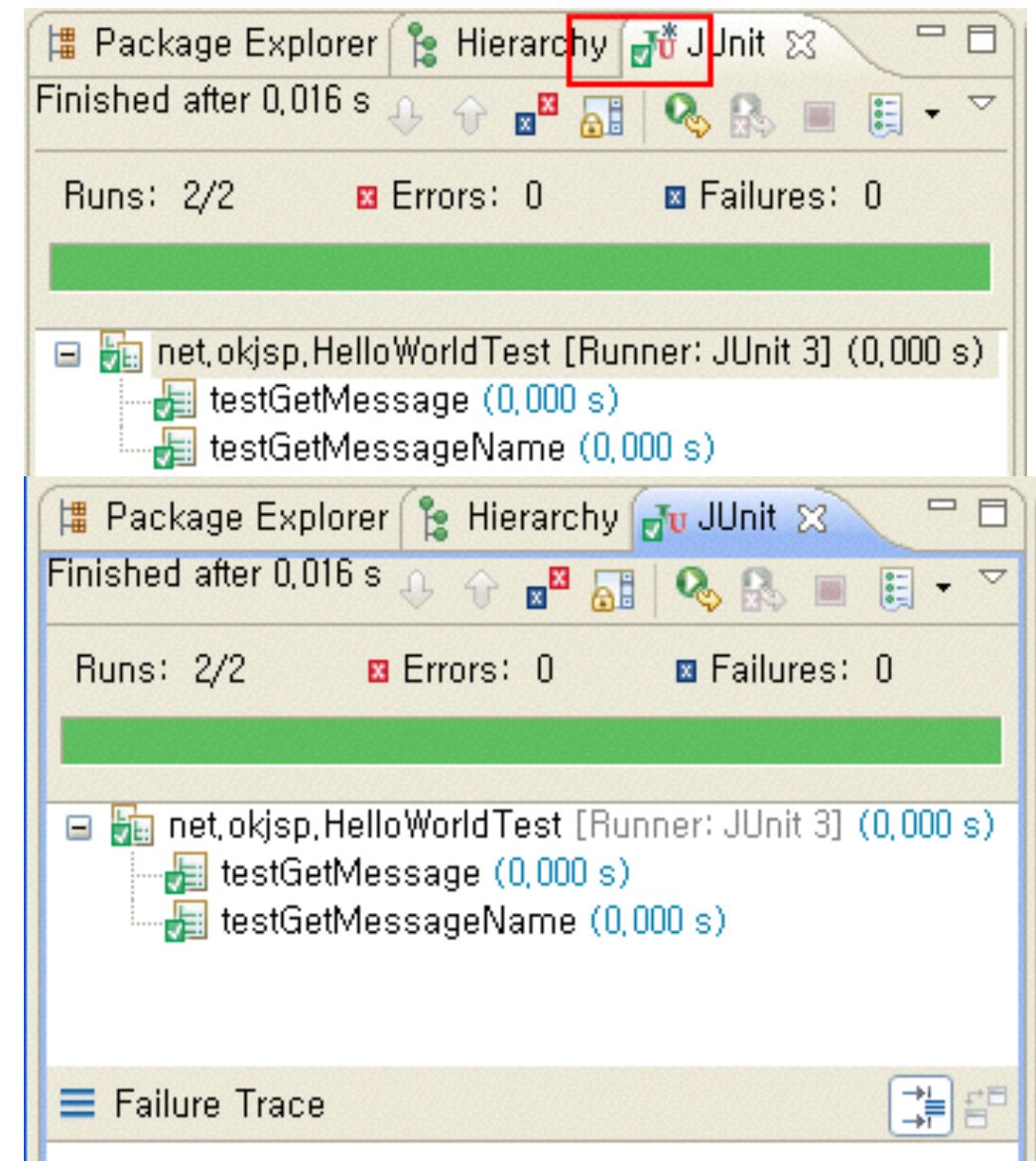
-중복제거 재활용

```
HelloWorldTest.java HelloWorld.java x
package net.okjsp;

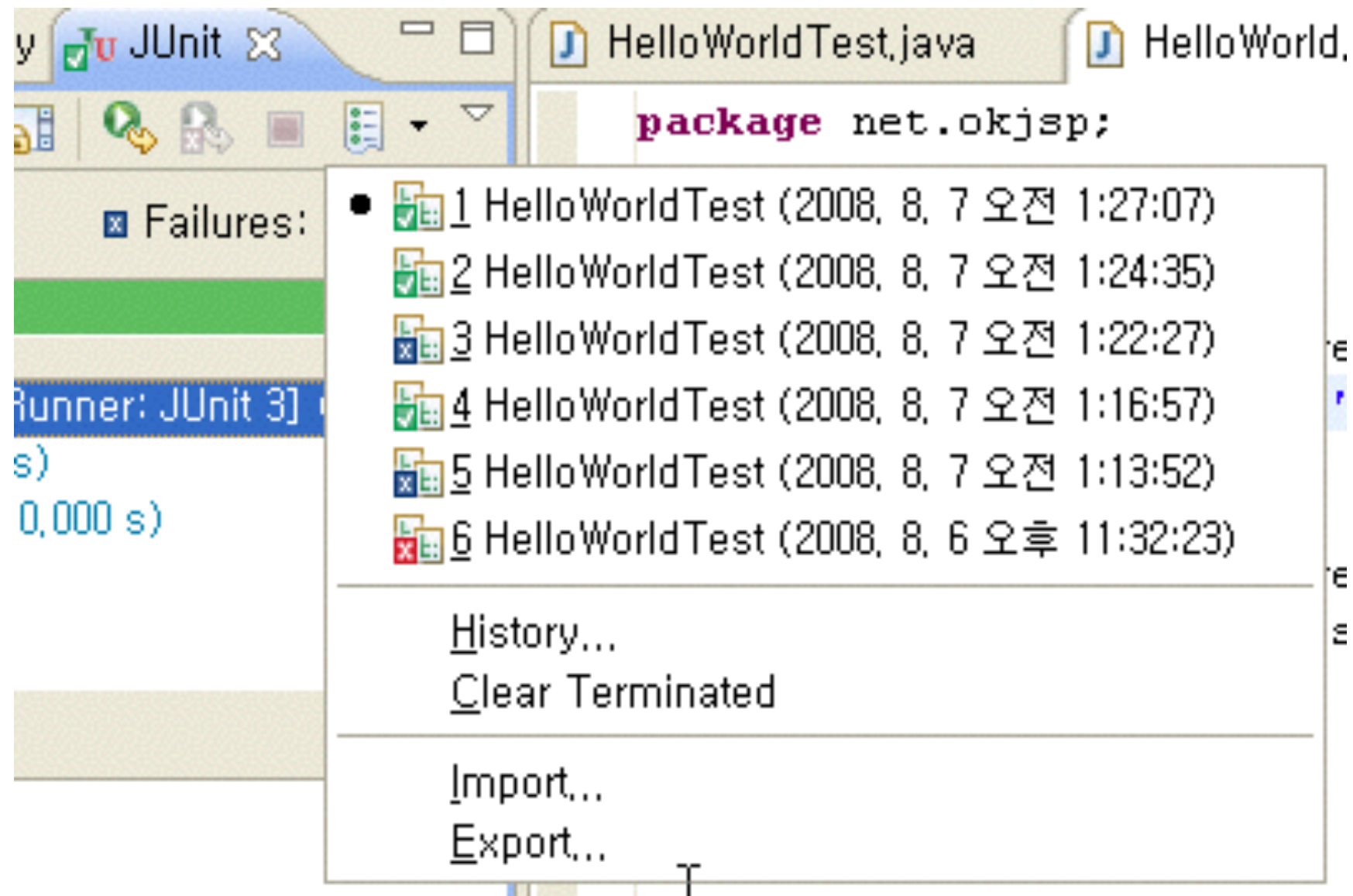
public class HelloWorld {

    public static String getMessage() {
        return getMessage("World");
    }

    public static String getMessage(String string)
        return "Hello " + string;
    }
}
```



테스트 히스토리



테스트 조건 강화하기



```
package net.okjsp;

import junit.framework.TestCase;

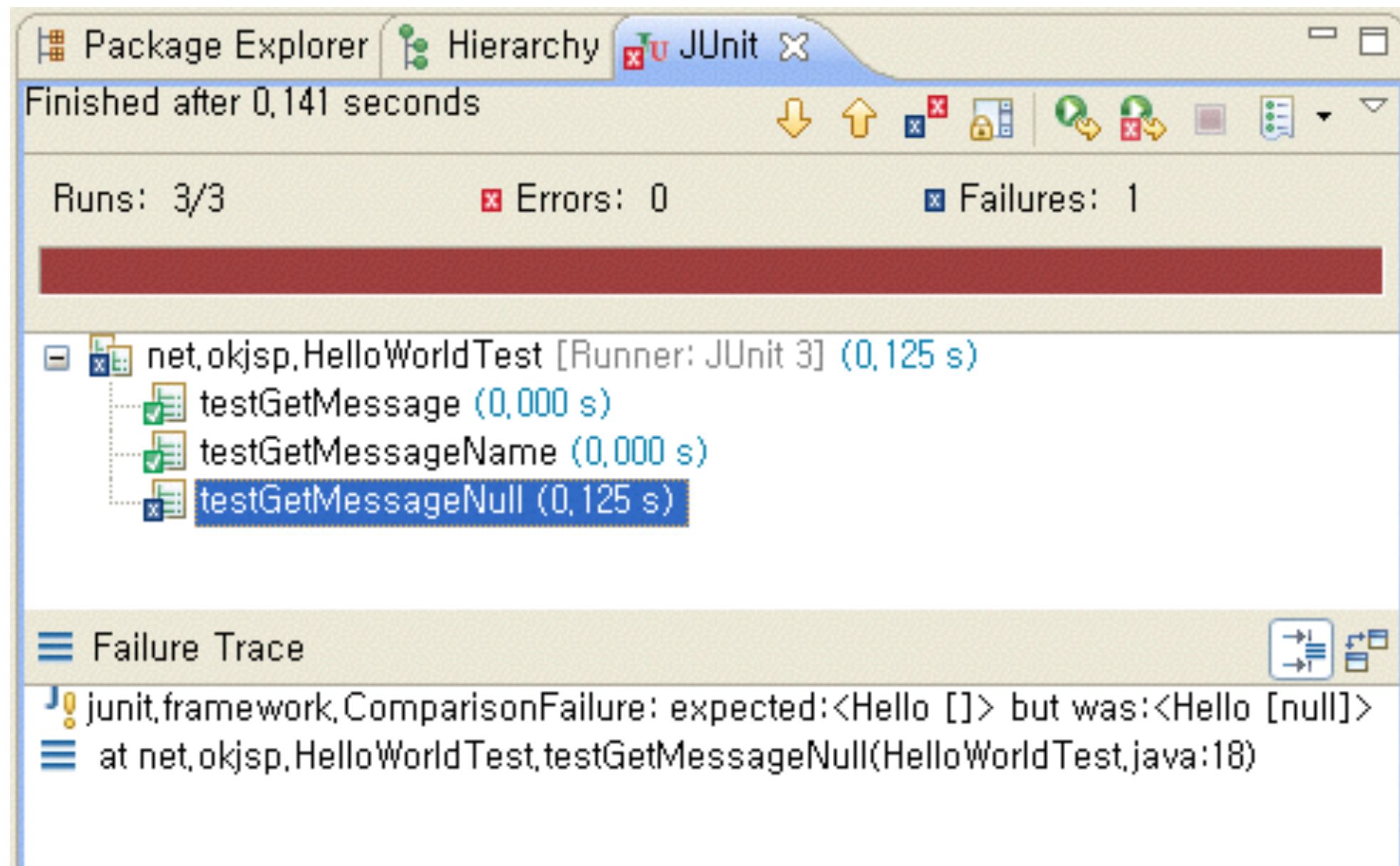
public class HelloWorldTest extends TestCase {

    public void testGetMessage() {
        String message = HelloWorld.getMessage();
        assertEquals("Hello World", message);
    }

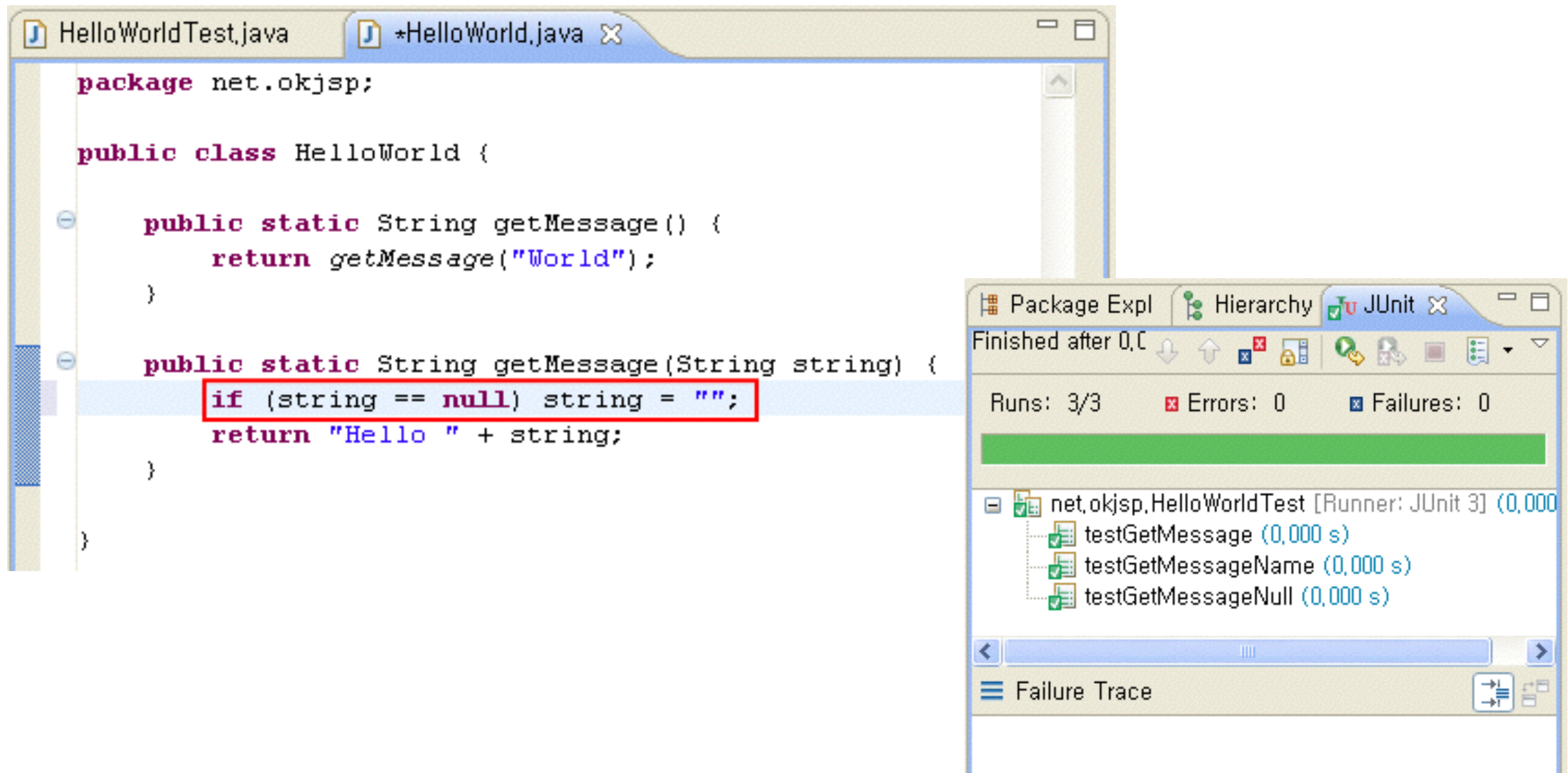
    public void testGetMessageName() {
        String message = HelloWorld.getMessage("kenu");
        assertEquals("Hello kenu", message);
    }

    public void testGetMessageNull() {
        String message = HelloWorld.getMessage(null);
        assertEquals("Hello ", message);
    }
}
```

다시 테스트 고



테스트 통과하게 하기



The image shows a screenshot of an IDE with two windows. The left window displays the source code for `HelloWorldTest.java` and `HelloWorld.java`. The right window shows the JUnit test results.

Source Code:

```
package net.okjsp;

public class HelloWorld {

    public static String getMessage() {
        return getMessage("World");
    }

    public static String getMessage(String string) {
        if (string == null) string = "";
        return "Hello " + string;
    }
}
```

JUnit Test Results:

Finished after 0.0s

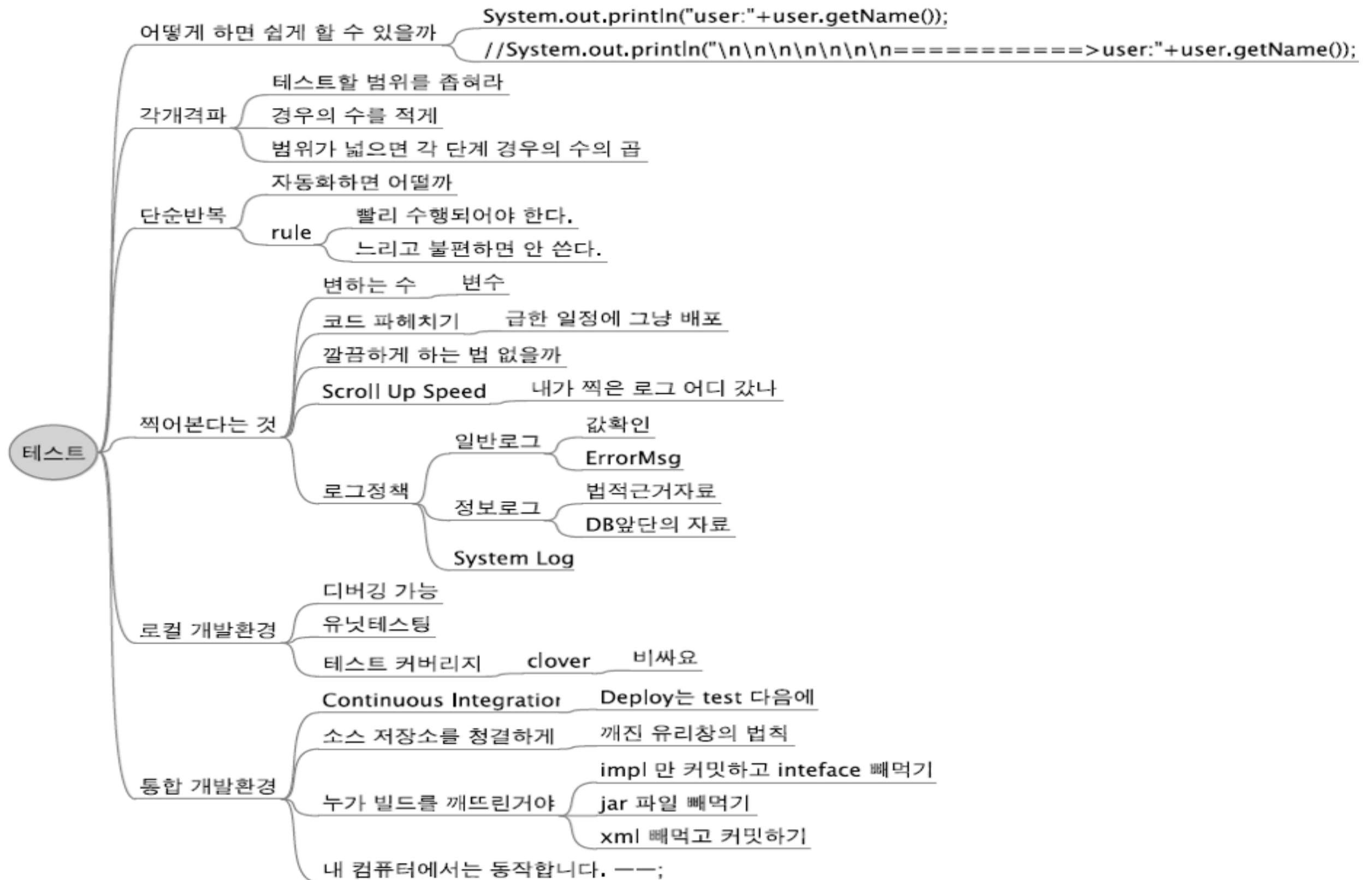
Runs: 3/3 Errors: 0 Failures: 0

net.okjsp.HelloWorldTest [Runner: JUnit 3] (0.000s)

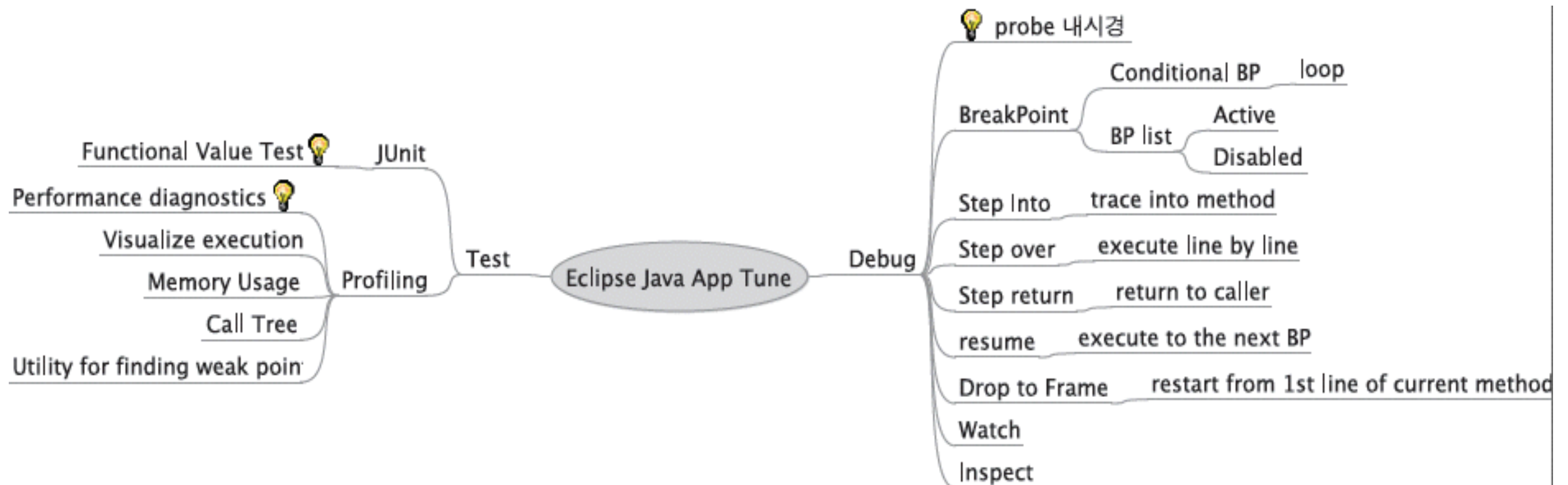
- testGetMessage (0.000 s)
- testGetMessageName (0.000 s)
- testGetMessageNull (0.000 s)

Failure Trace

테스트에 관하여



Test and Debug



참고

- <http://junit.org>
- <http://xper.org>
- <http://okjsp.tistory.com/tag/test>

연습문제

- 테스트케이스란?

2. JUnit 이해하기

- java version of xUnit family.
 - CPPUNIT, JUnit, ASUnit, httpUnit...
- Kent Beck, Erich Gamma
- assert
- TestCase, TestSuite
- Matter of S/W quality.

sf.net/projects/junit

News **Details** Public Related Activity Freshmeat.net

Project Admins : [dsaff](#), [egamma](#), [emeade](#), [kbeck](#)

Developers : [Z](#)

Development Status : [5 - Production/Stable](#)

Intended Audience : [Developers](#)

License : [Common Public License 1.0](#)

Operating System : [OS Independent \(Written in an interpreted language\)](#)

Programming Language : [Java](#)

Topic : [Software Development](#)

Translations : [English](#)

User Interface : [Win32 \(MS Windows\)](#), [X Window System \(X11\)](#)

Project UNIX name : junit

Registered : 2000-11-25 00:05

TestCase

- Ver3.8.x
- Extends TestCase
- assertTrue
- assertEquals
- Ver4.x
- Annotation

TestSuite

- AllTests
- Group of TestCases
- Regression Test Automation

연습문제

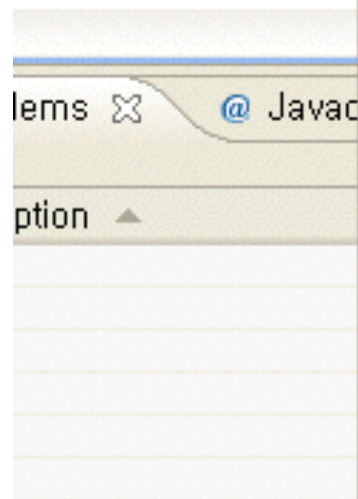
- JUnit 테스트케이스로 디버깅하기
- TestCase와 TestSuite 구분하기

3. 테스트케이스와 리팩토링

리팩토링 하기

```
public void testGetMessageName() {
    assertEquals("Hello kenu", HelloWorld.getMessage("kenu"));
}
```

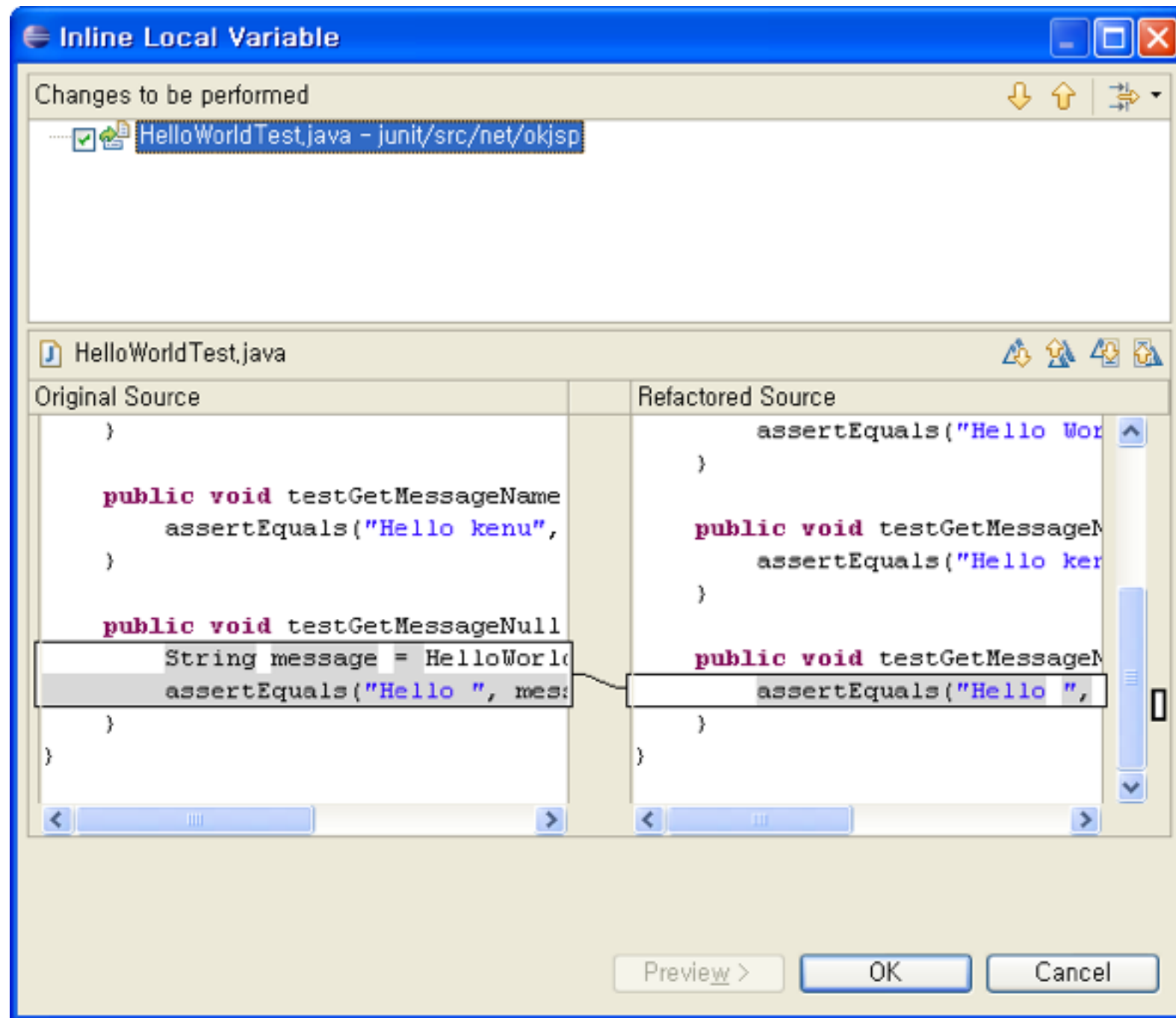
```
public void testGetMessageNull() {
    String mes
    assertEquals
}
```



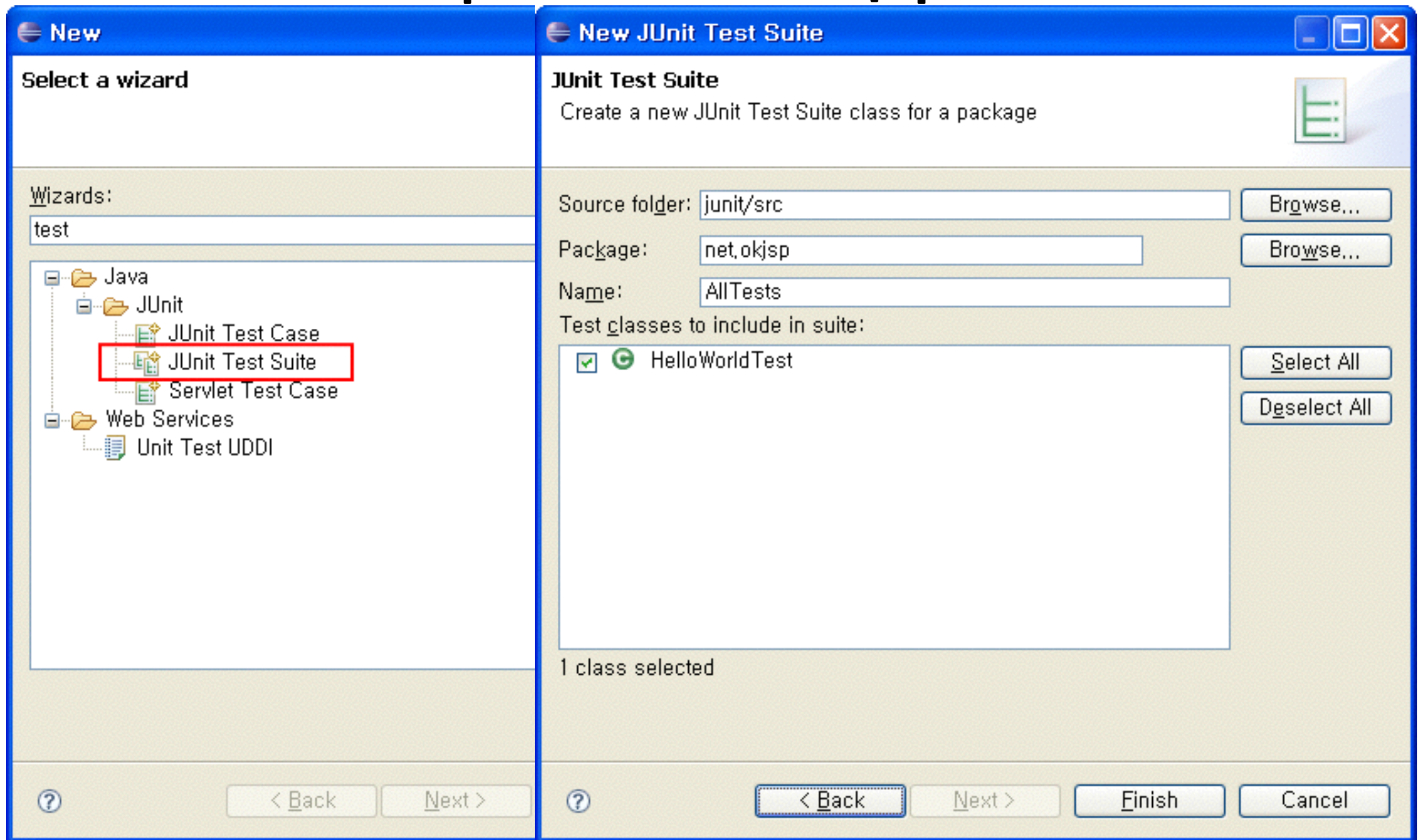
Undo Inline local variable	Ctrl+Z
Revert File	
Save	Ctrl+S
Open Declaration	F3
Open Type Hierarchy	F4
Open Call Hierarchy	Ctrl+Alt+H
Show in Breadcrumb	Alt+Shift+B
Quick Outline	Ctrl+O
Quick Type Hierarchy	Ctrl+T
Show In	Alt+Shift+W
Cut	Ctrl+X
Copy	Ctrl+C
Copy Qualified Name	
Paste	Ctrl+V
Quick Fix	Ctrl+I
Source	Alt+Shift+S
Refactor	Alt+Shift+T
Local History	
References	

Rename...	Alt+Shift+R
Move...	Alt+Shift+V
Change Method Signature...	Alt+Shift+C
Extract Local Variable...	Alt+Shift+L
Extract Constant...	
Inline...	Alt+Shift+I
Convert Local Variable to Field...	
Extract Interface...	
Extract Superclass...	
Use Supertype Where Possible...	
Pull Up...	
Push Down...	
Extract Class...	
Introduce Parameter Object...	
Introduce Parameter...	
Generalize Declared Type...	
Infer Generic Type Arguments...	

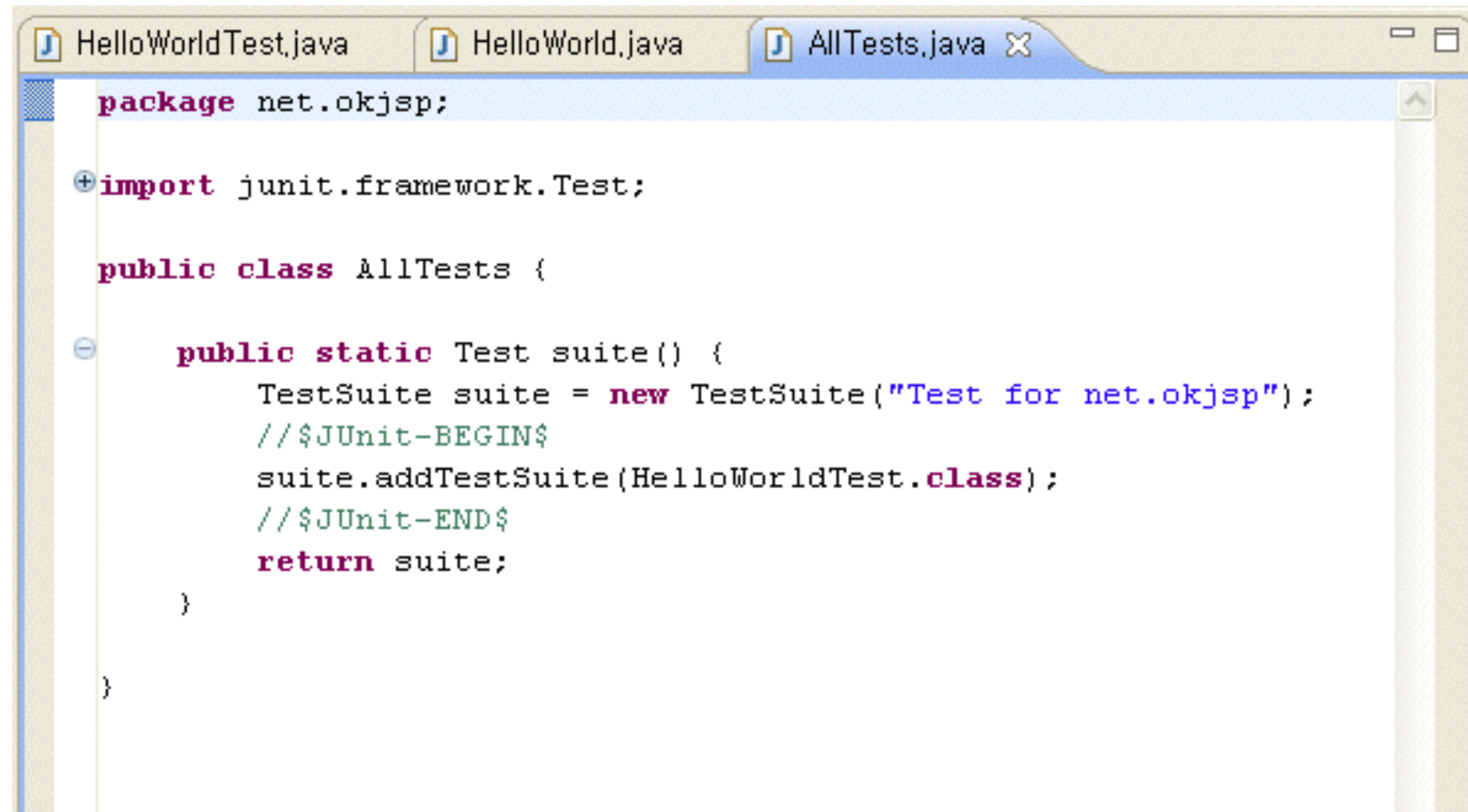
리팩토링 하기-inline



테스트 스위트



테스트 스위트



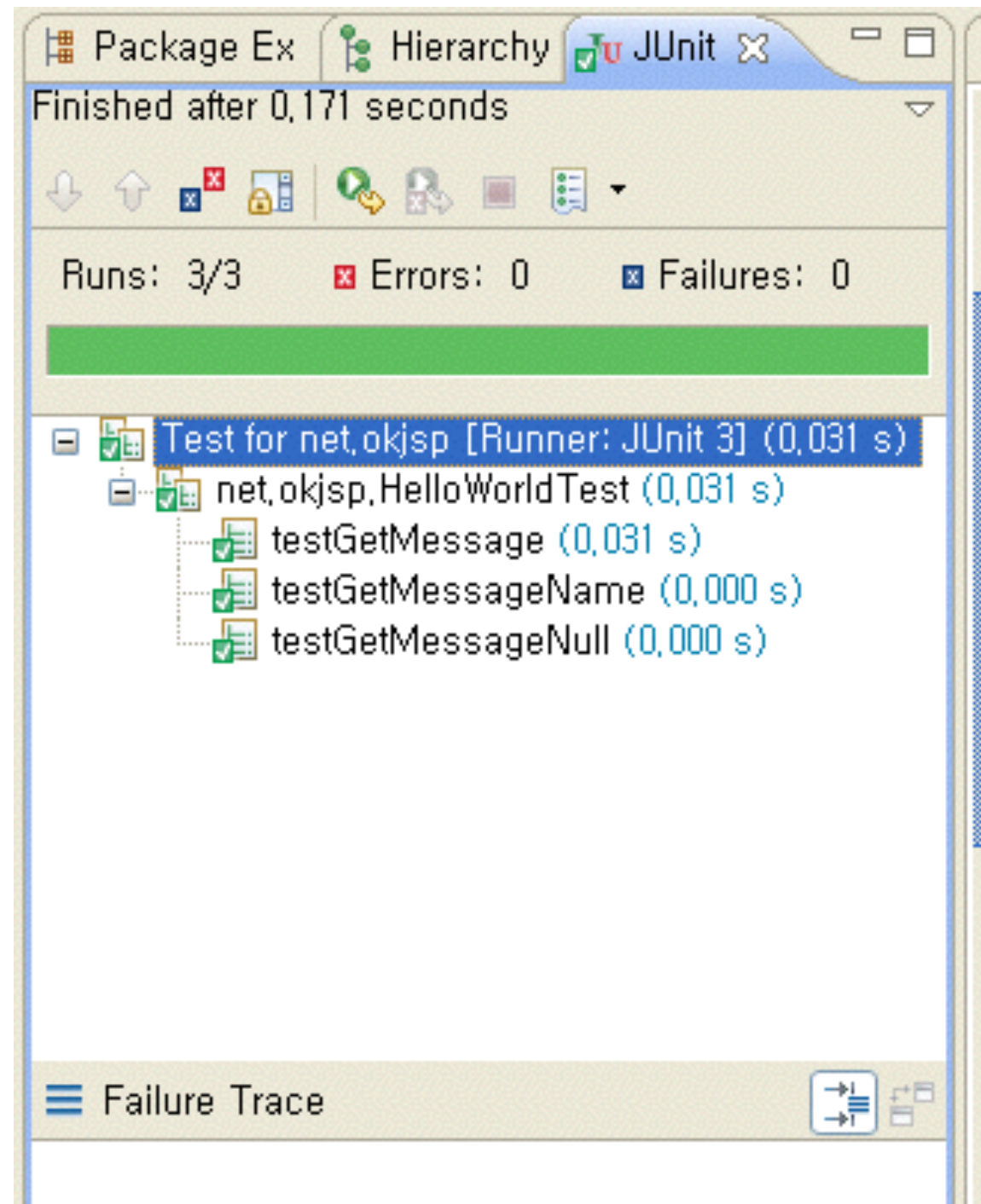
```
package net.okjsp;

import junit.framework.Test;

public class AllTests {

    public static Test suite() {
        TestSuite suite = new TestSuite("Test for net.okjsp");
        //$JUnit-BEGIN$
        suite.addTestSuite(HelloWorldTest.class);
        //$JUnit-END$
        return suite;
    }
}
```

전체 테스트하기

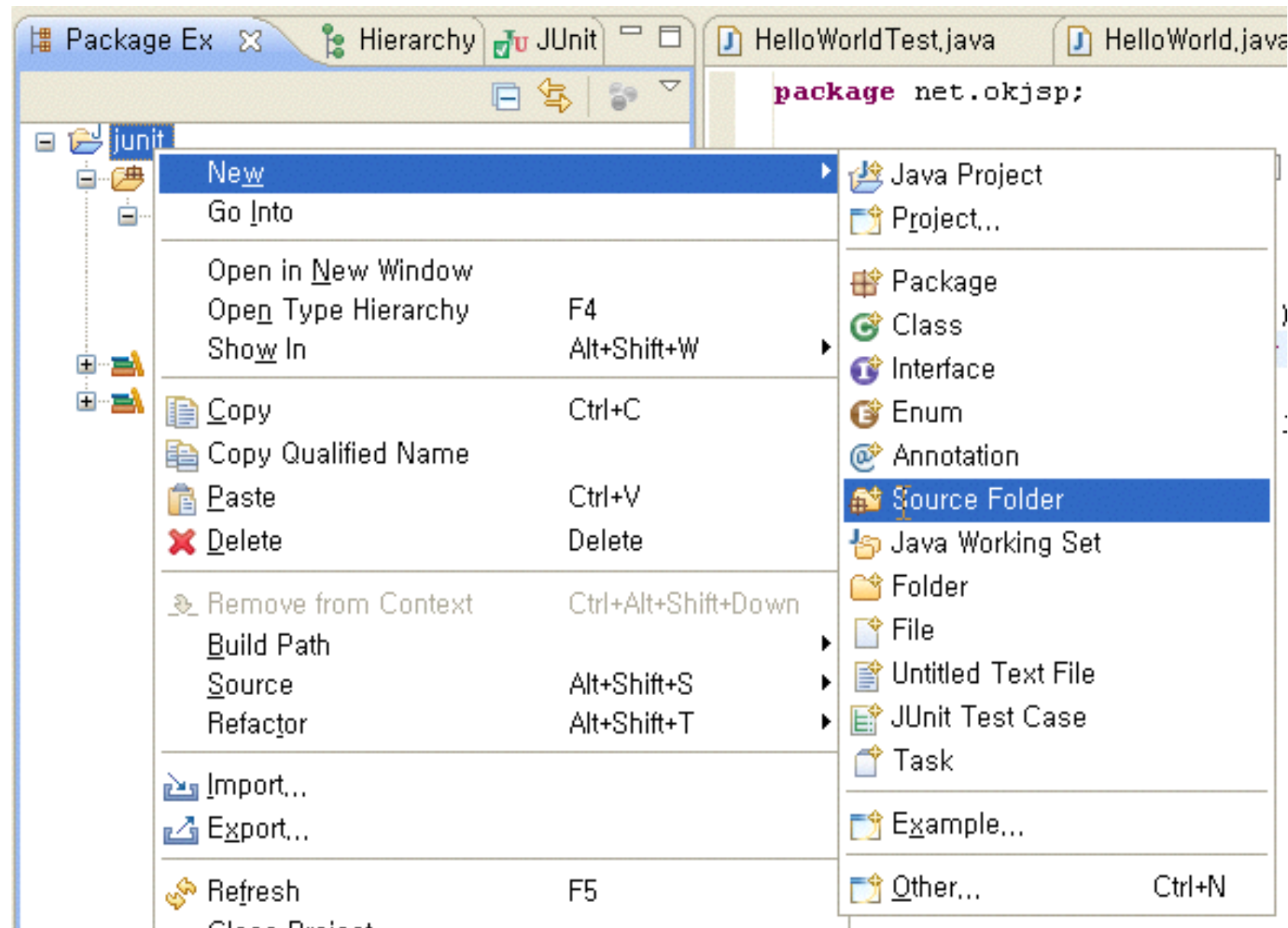


정리-테스트 리듬

- Rhythm
 - TestCase
 - Change Small Code
 - All Test
 - Refactoring
 - All Test
- 돌다리도 두드려 가라.
- 모래 코드 위에 코드를 추가하지 말라.

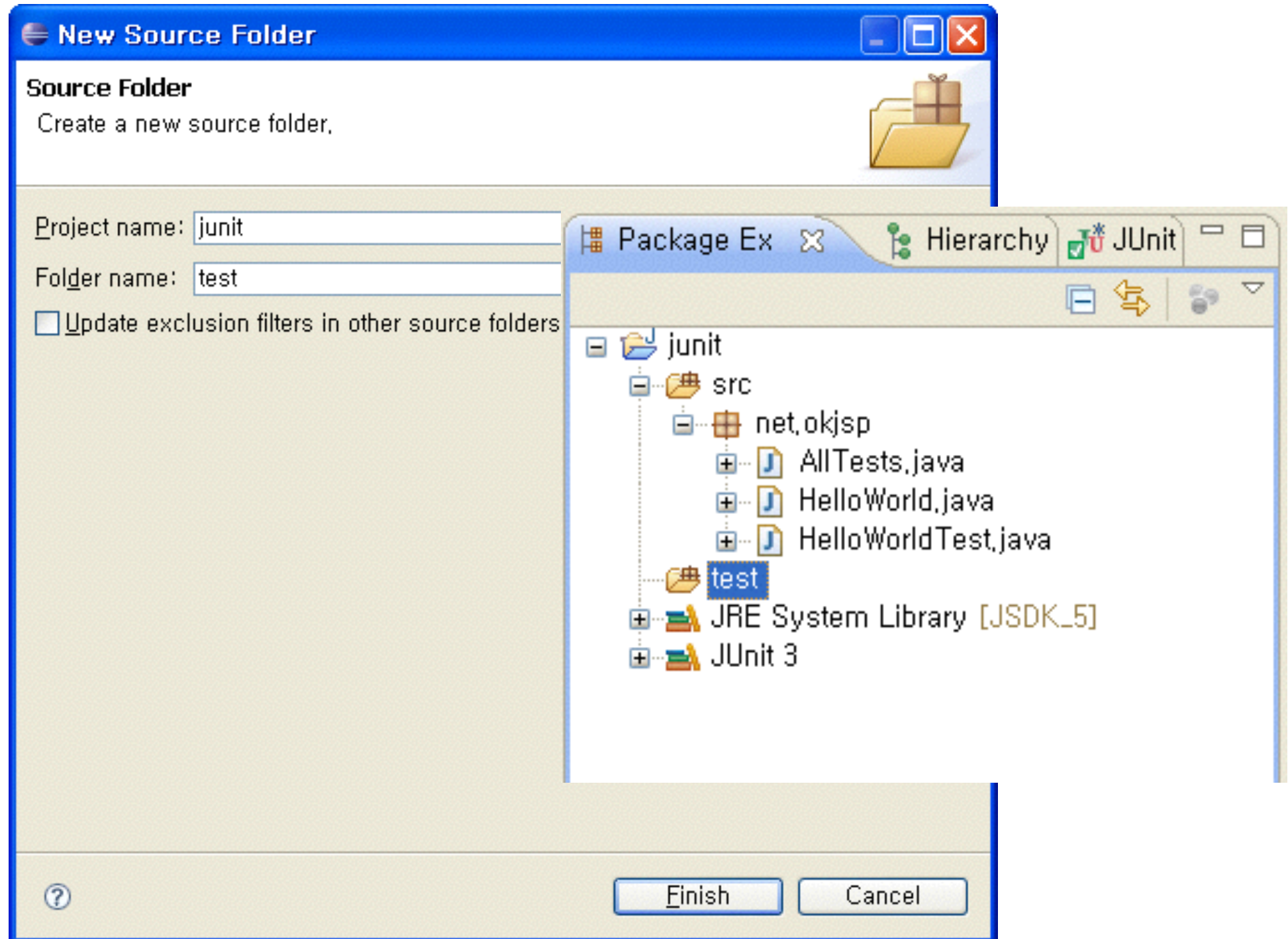
테스트 코드 분리하기

-소스폴더

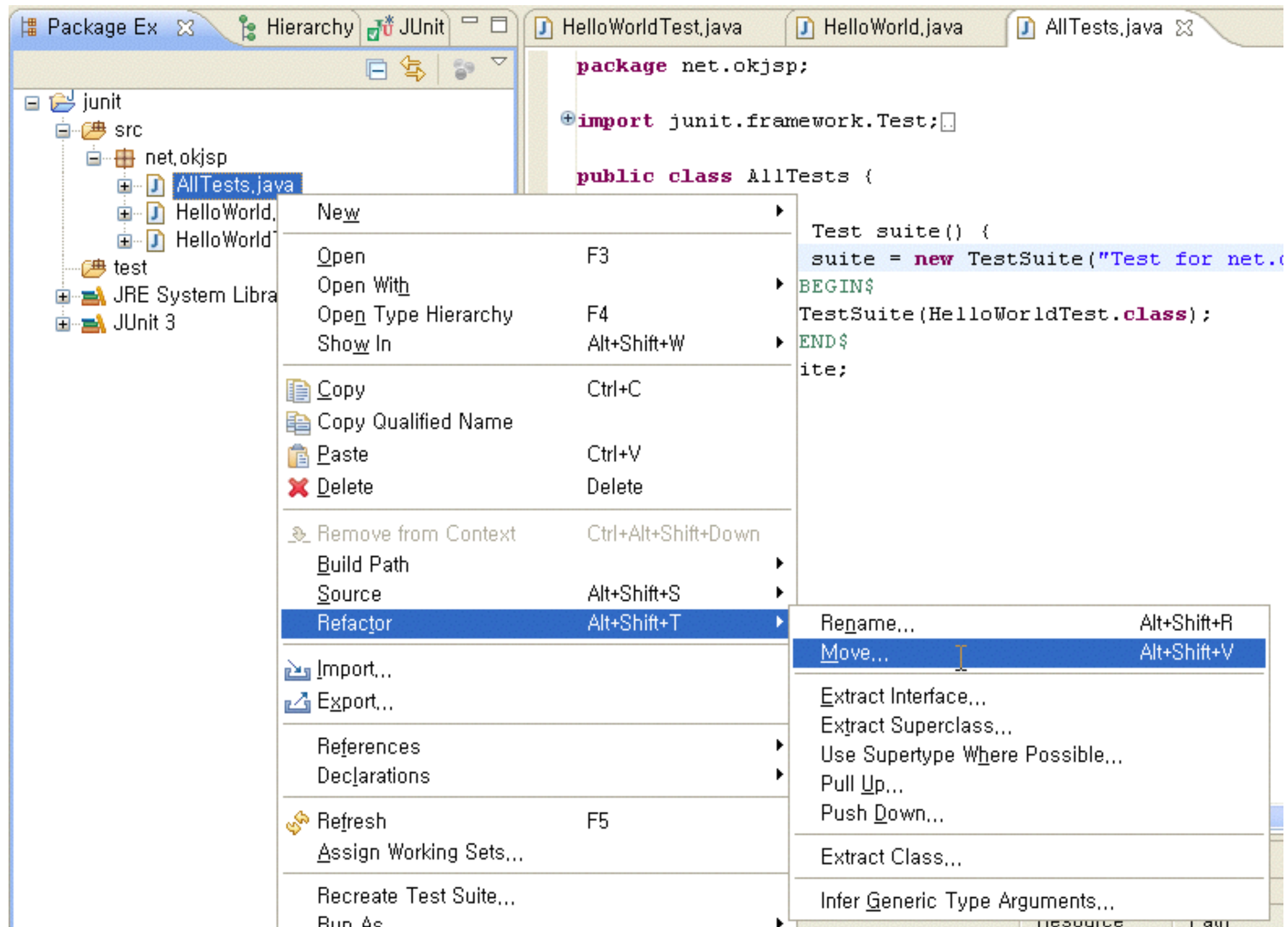


테스트 코드 분리하기

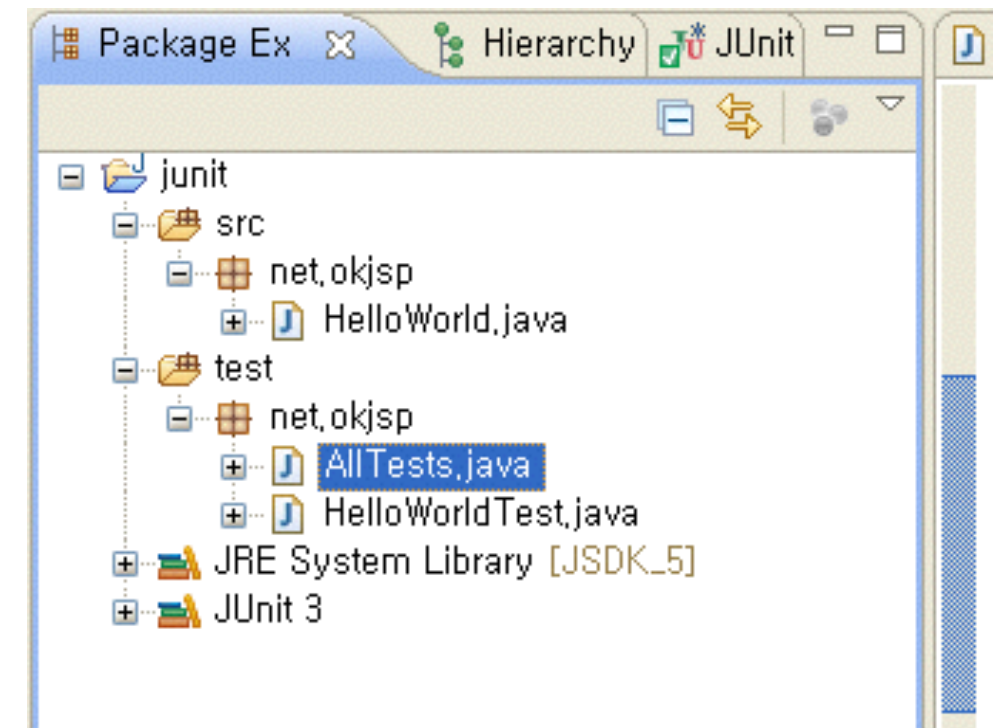
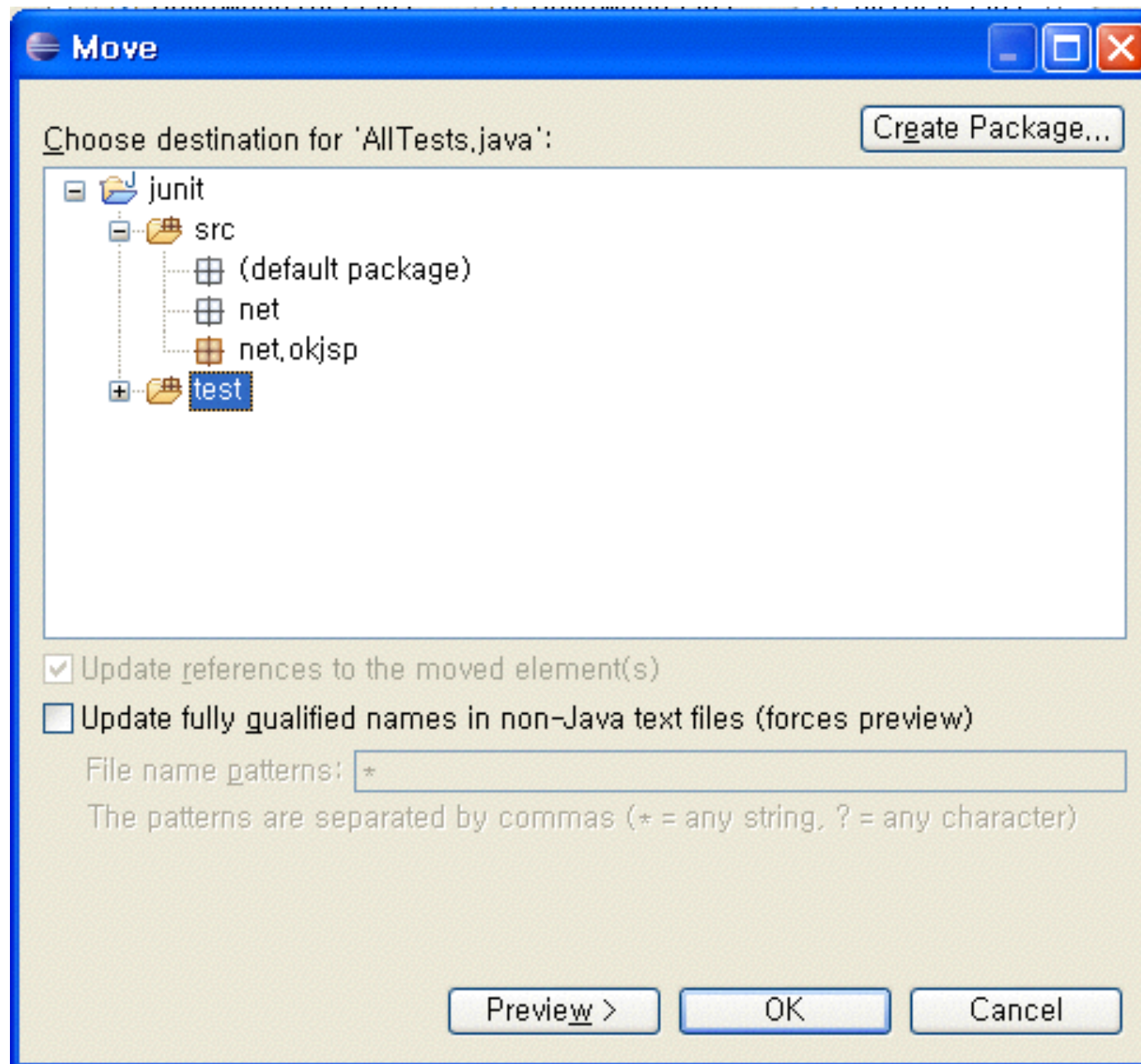
-소스폴더



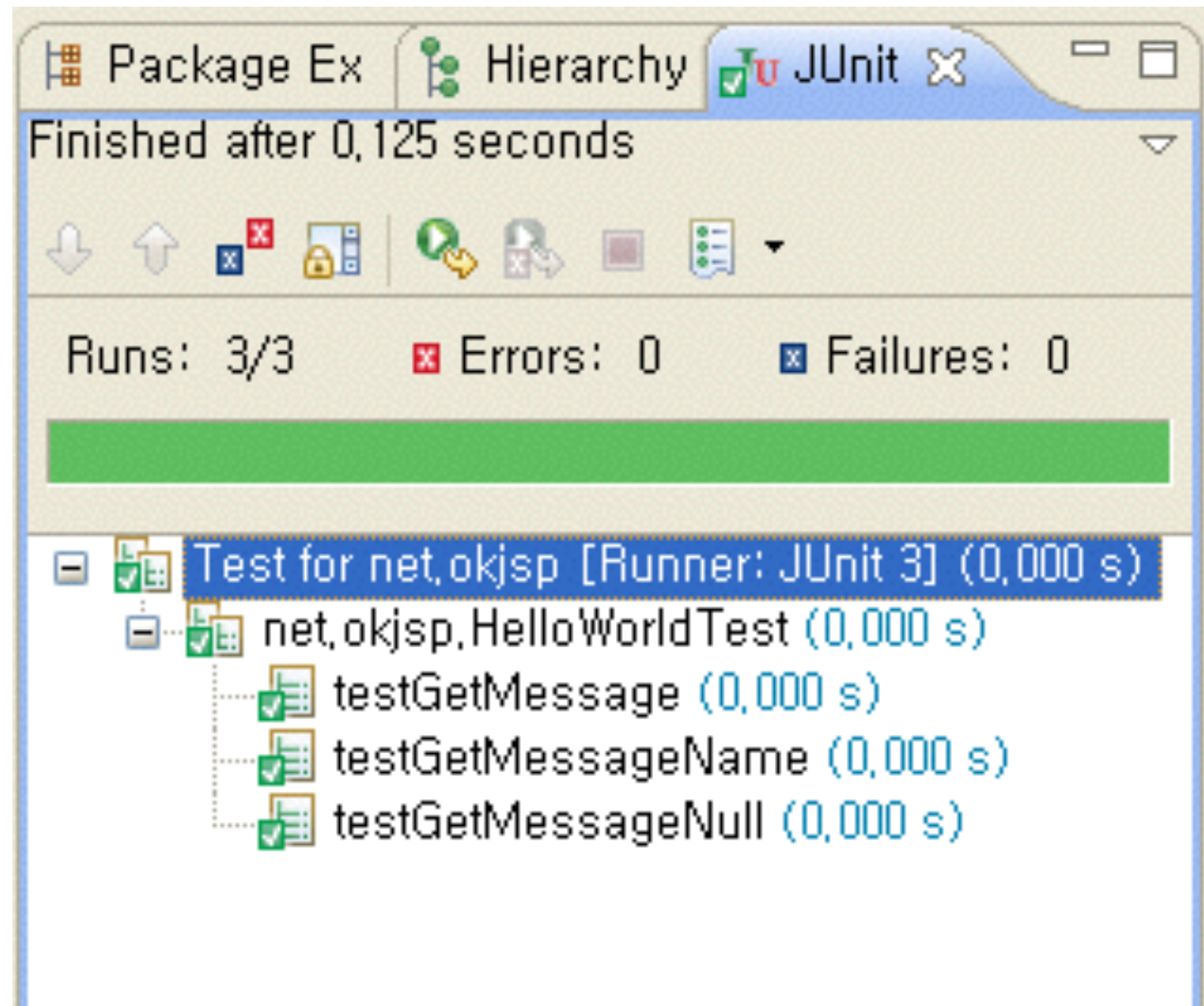
테스트 코드 분리-Move



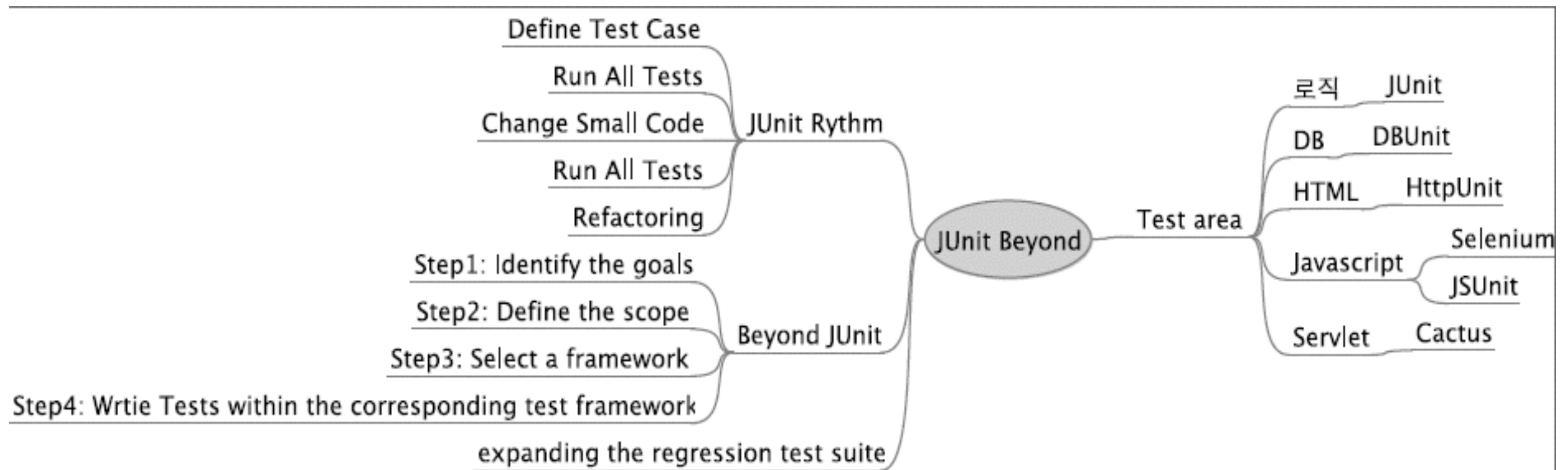
테스트 코드 분리-Move



다시 테스트 고



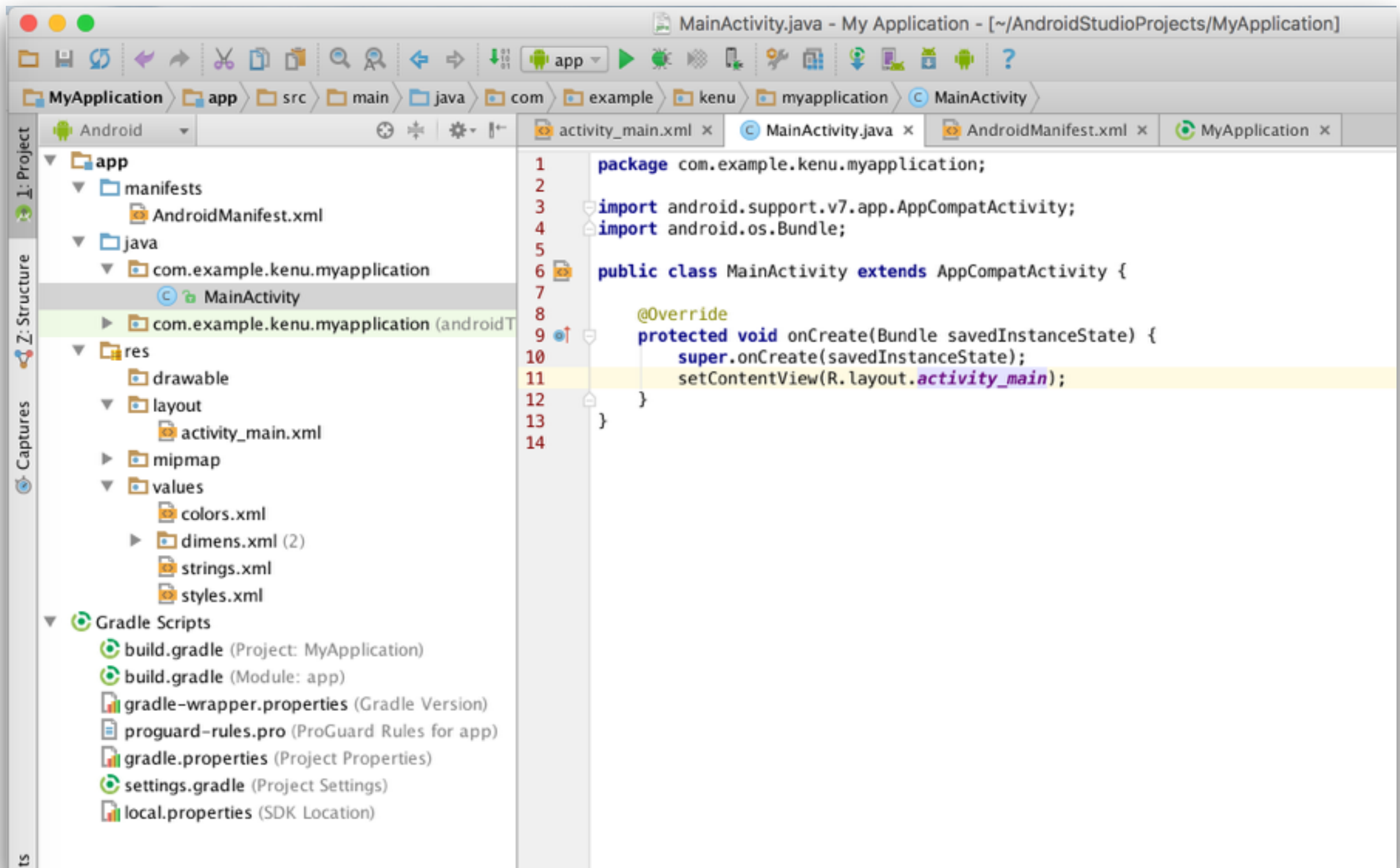
JUnit Beyond

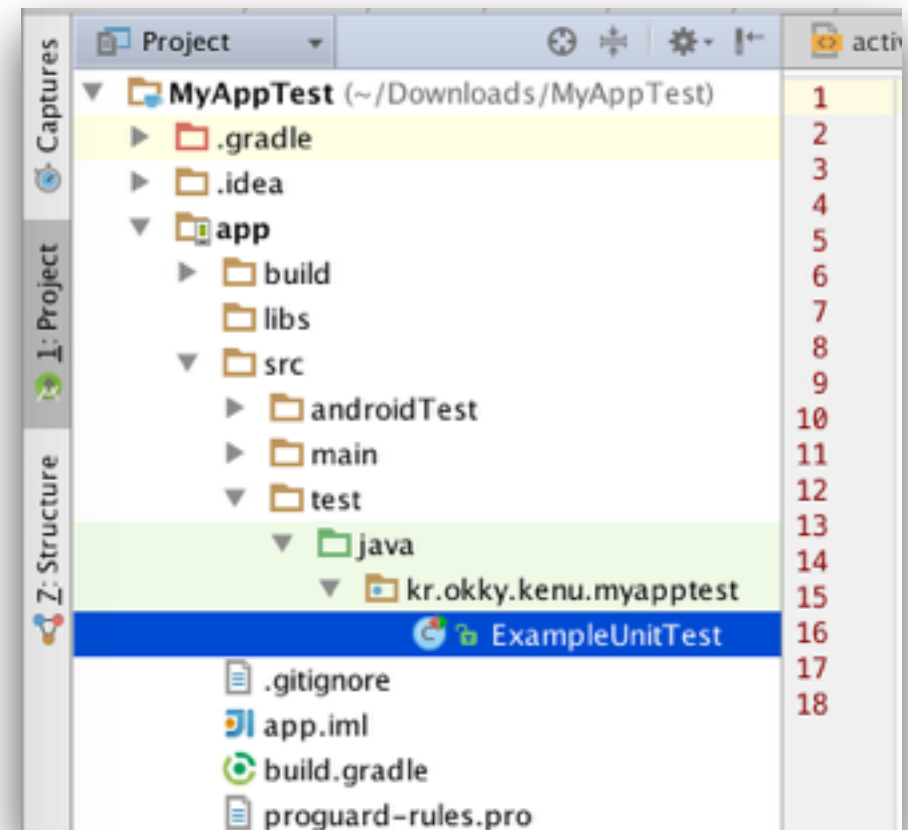


연습문제

- 리팩토링이란?
- 리팩토링과 테스트케이스의 관계?

안드로이드 스튜디오 JUnit





- 1. Project > Android 탭을 Project로 변경
- app > src > test 폴더의 JUnit 테스트 코드 활용
- 왼쪽 사이드바 Build Variants 클릭

