

Introduction to JUnit

Dependable Software Laboratory

Contents

- Introduction to Junit
- Assert
- Annotation
- Installation
- MOK

JUnit

- JUnit is a simple framework to write repeatable tests. It is an instance of the xUnit architecture for unit testing frameworks.
 - Unit test framework
 - Assertion
 - Annotation

- <http://junit.org/junit4/>

Assertions

- <http://junit.sourceforge.net/javadoc/org/junit/Assert.html>
 - Assert API

Asserts	
assertArrayEquals	Asserts that two object arrays are equal. If they are not, an AssertionError is thrown with the given message. If expected and actuals are null, they are considered equal.
AssertEquals	Asserts that two objects are equal. If they are not, an AssertionError without a message is thrown. If expected and actual are null, they are considered equal.
assertFalse	Asserts that a condition is false. If it isn't it throws an AssertionError without a message.
assertNotNull	Asserts that an object isn't null. If it is an AssertionError is thrown.
assertNotSame	Asserts that two objects do not refer to the same object. If they do refer to the same object, an AssertionError without a message is thrown.
assertNull	Asserts that an object is null. If it isn't an AssertionError is thrown.
assertSame	Asserts that two objects refer to the same object. If they are not the same, an AssertionError without a message is thrown.
assertThat	
assertTrue	Asserts that a condition is true. If it isn't it throws an AssertionError without a message.
fail	Fails a test with the given message.

Assertions

assertThat

```
public static <T> void assertThat(T actual,
                                org.hamcrest.Matcher<T> matcher)
```

Asserts that `actual` satisfies the condition specified by `matcher`. If not, an `AssertionError` is thrown with information about the matcher and failing value. Example:

```
assertThat(0, is(1)); // fails:
// failure message:
// expected: is <1>
// got value: <0>
assertThat(0, is(not(1))) // passes
```

Type Parameters:

`T` - the static type accepted by the matcher (this can flag obvious compile-time problems such as `assertThat(1, is("a"))`)

Parameters:

`actual` - the computed value being compared

`matcher` - an expression, built of `Matchers`, specifying allowed values

See Also:

`CoreMatchers`, [JUnitMatchers](#)

- 각 assert 별로 다양한 객체 지원

static void	<code>assertEquals(double expected, double actual)</code> Deprecated. Use <code>assertEquals(double expected, double actual, double epsilon)</code> instead
static void	<code>assertEquals(double expected, double actual, double delta)</code> Asserts that two doubles or floats are equal to within a positive delta.
static void	<code>assertEquals(long expected, long actual)</code> Asserts that two longs are equal.
static void	<code>assertEquals(java.lang.Object[] expecteds, java.lang.Object[] actuals)</code> Deprecated. use <code>assertArrayEquals</code>
static void	<code>assertEquals(java.lang.Object expected, java.lang.Object actual)</code> Asserts that two objects are equal.
static void	<code>assertEquals(java.lang.String message, double expected, double actual)</code> Deprecated. Use <code>assertEquals(String message, double expected, double actual, double epsilon)</code> instead
static void	<code>assertEquals(java.lang.String message, double expected, double actual, double delta)</code> Asserts that two doubles or floats are equal to within a positive delta.
static void	<code>assertEquals(java.lang.String message, long expected, long actual)</code> Asserts that two longs are equal.
static void	<code>assertEquals(java.lang.String message, java.lang.Object[] expecteds, java.lang.Object[] actuals)</code> Deprecated. use <code>assertArrayEquals</code>
static void	<code>assertEquals(java.lang.String message, java.lang.Object expected, java.lang.Object actual)</code> Asserts that two objects are equal.

Annotation

Annotation	설명
@Test	Unit Test를 수행하는 대상 method
@Before	각 Unit test의 method 실행 전에 실행되는 method
@After	각 Unit test의 실행 후에 실행되는 method
@BeforeClass	Class안에 정의된 모든 method에 대해서 Test 전, 후에 한번만 호출된다. 객체 생성 등에 사용.
@AfterClass	
@Ignore	테스트를 수행하지 않을 method
@RunWith(value=class)	Unit Test 클래스를 실행하기 위한 러너(Runner)를 명시적으로 지정할 수 있다.
@SuiteClasses (value=class)	보통 여러 개의 Test Class를 수행하기 위해 쓰인다. @RunWith를 이용해 Suite Class를 러너로 사용한다.
@Parameter	하나의 method에 대해 다양한 테스트 값을 한꺼번에 실행시키고자 할 때 사용한다.

Annotation

- The Test annotation tells JUnit that the public void method to which it is attached can be run as a test case.

```
@Test
public void testSum() {

}
```

```
@Test(timeout=5000)
public void testSum() {

}
```


Annotation

- If you allocate expensive external resources in a [BeforeClass](#) method you need to release them after all the tests in the class have run.
- Sometimes several tests need to share computationally expensive setup (like logging into a database).

```

@BeforeClass
public static void setUpBeforeClass() throws Exception {

}

@AfterClass
public static void tearDownAfterClass() throws Exception {

}

```

Annotation

- If you allocate external resources in a [Before](#) method you need to release them after the test runs.
- When writing tests, it is common to find that several tests need similar objects created before they can run.

```

@Before
public void setUp() throws Exception {

}

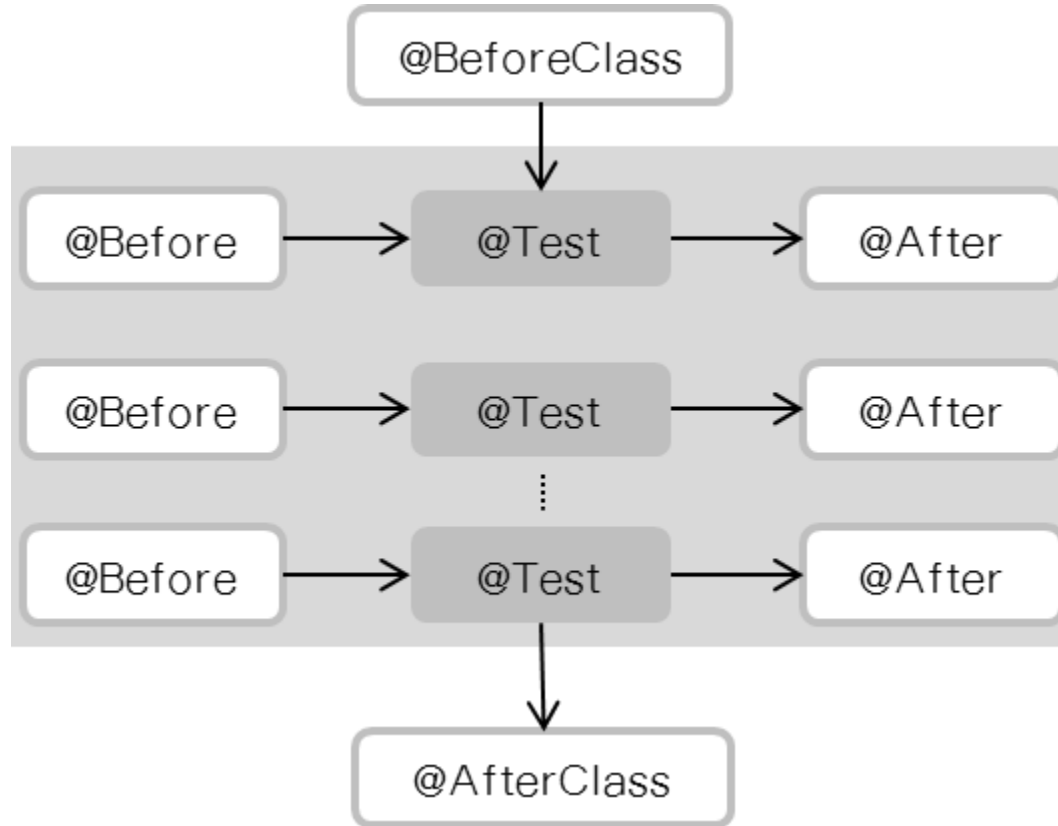
@After
public void tearDown() throws Exception {

}

```

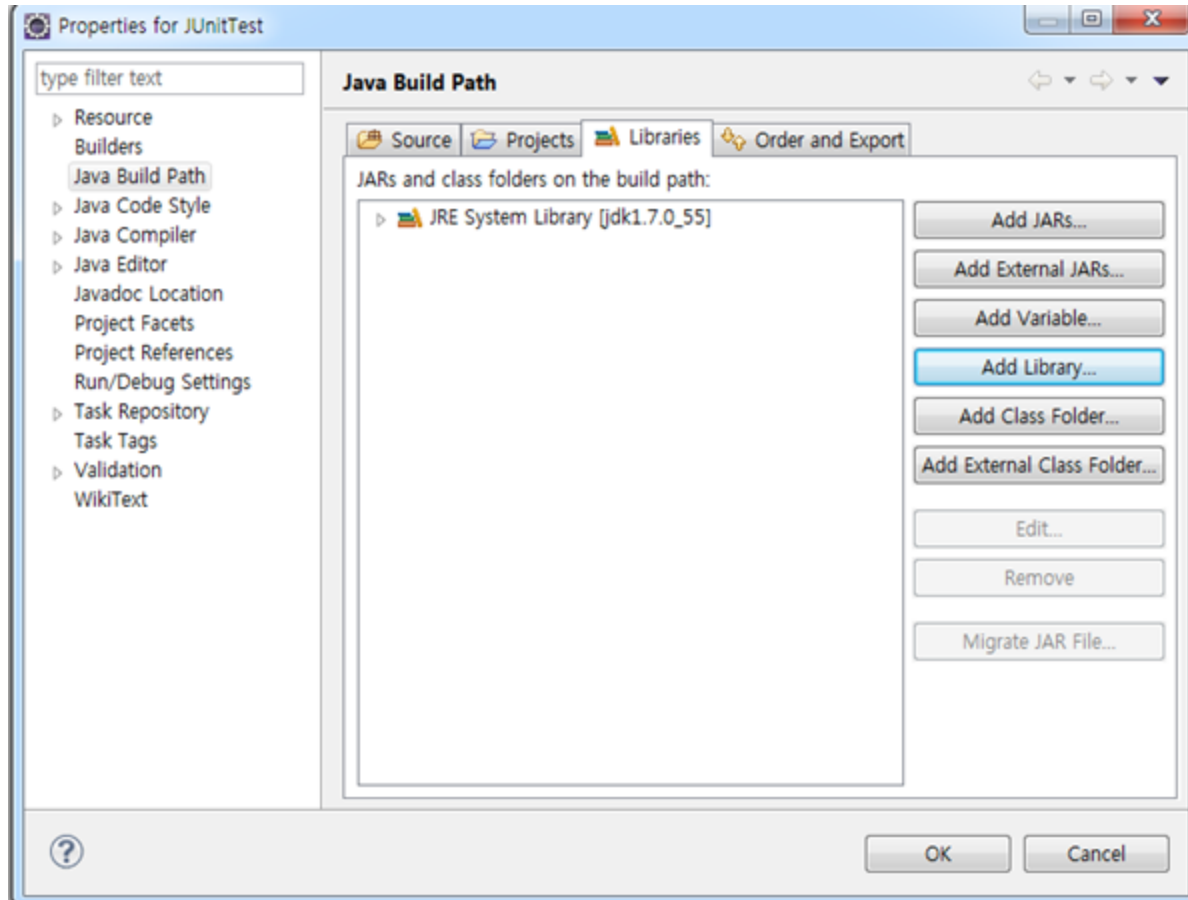
Annotation

- Annotation 흐름

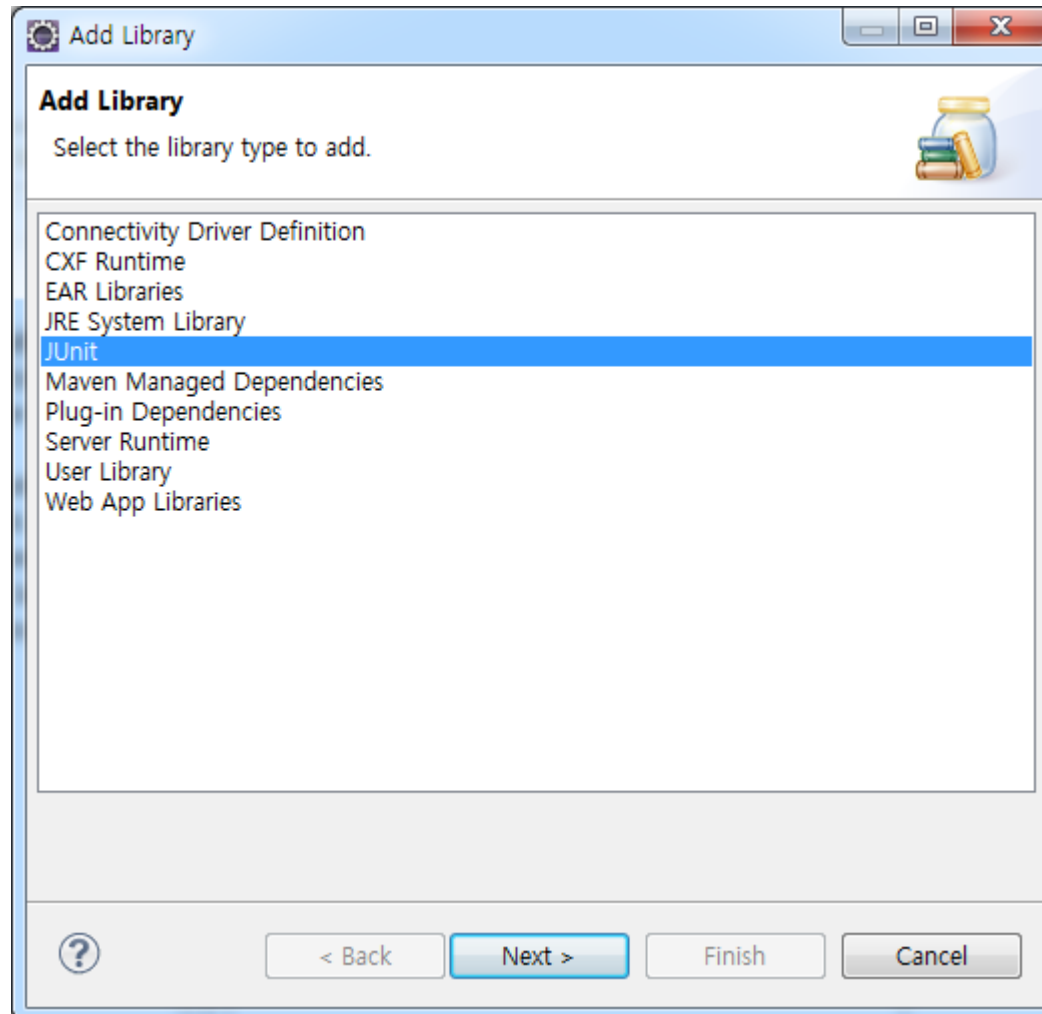


Installation

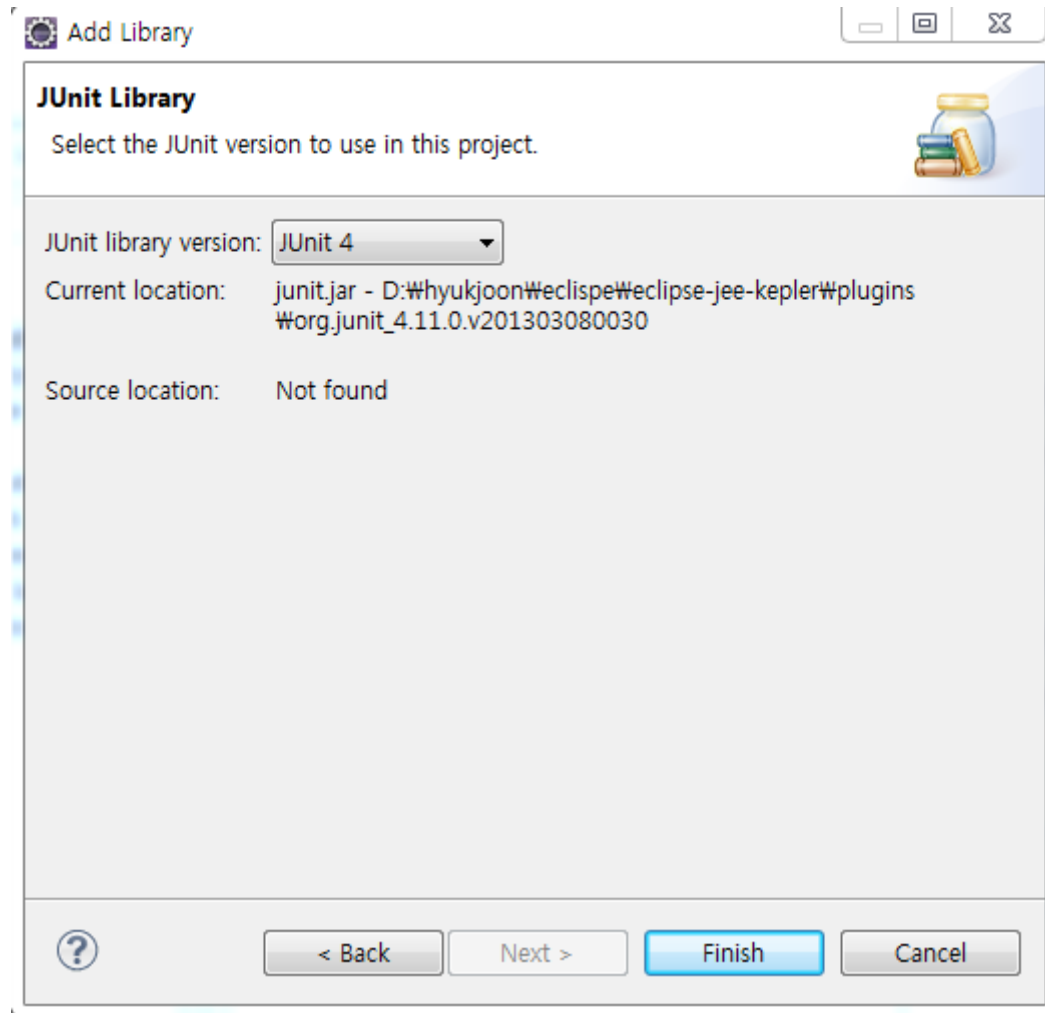
- Eclipse를 사용하면 JUnit library가 기본적으로 내장



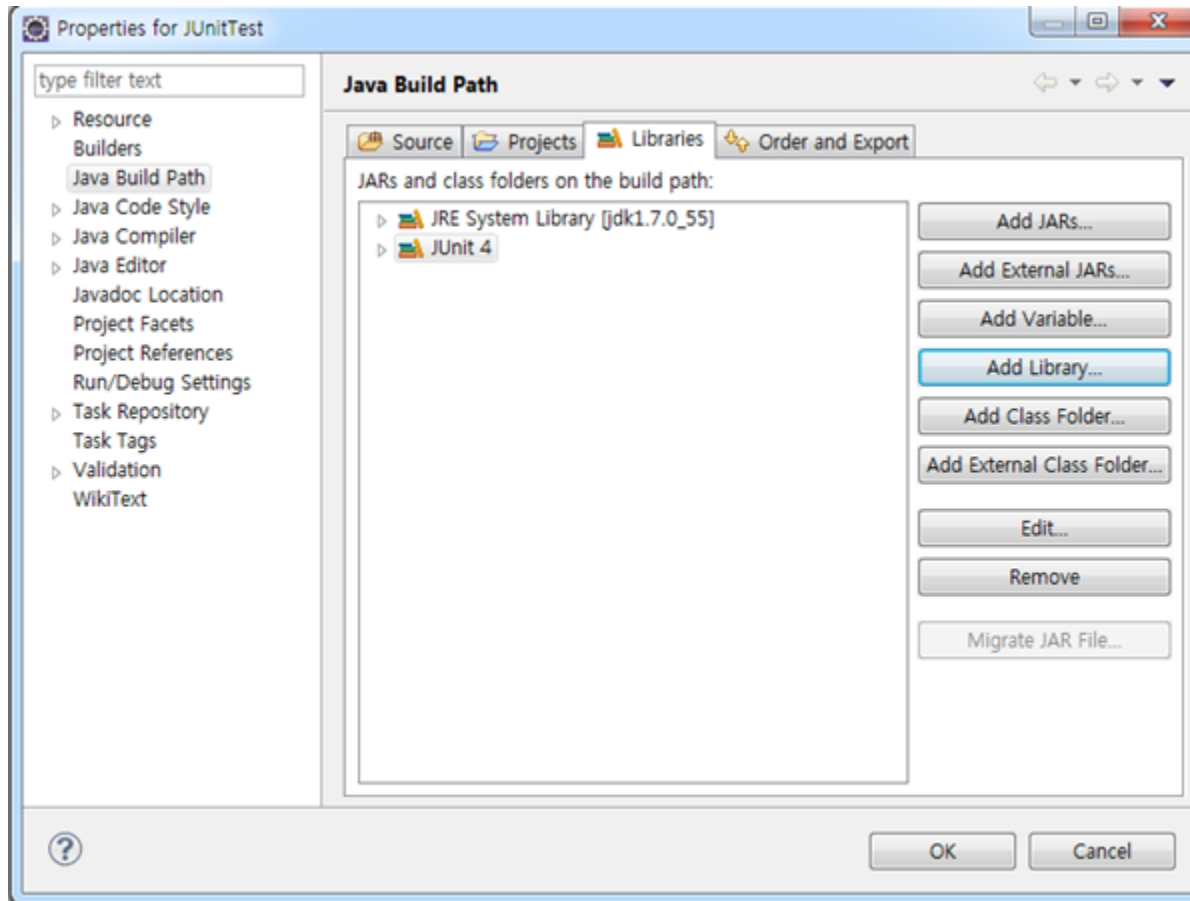
Installation



Installation

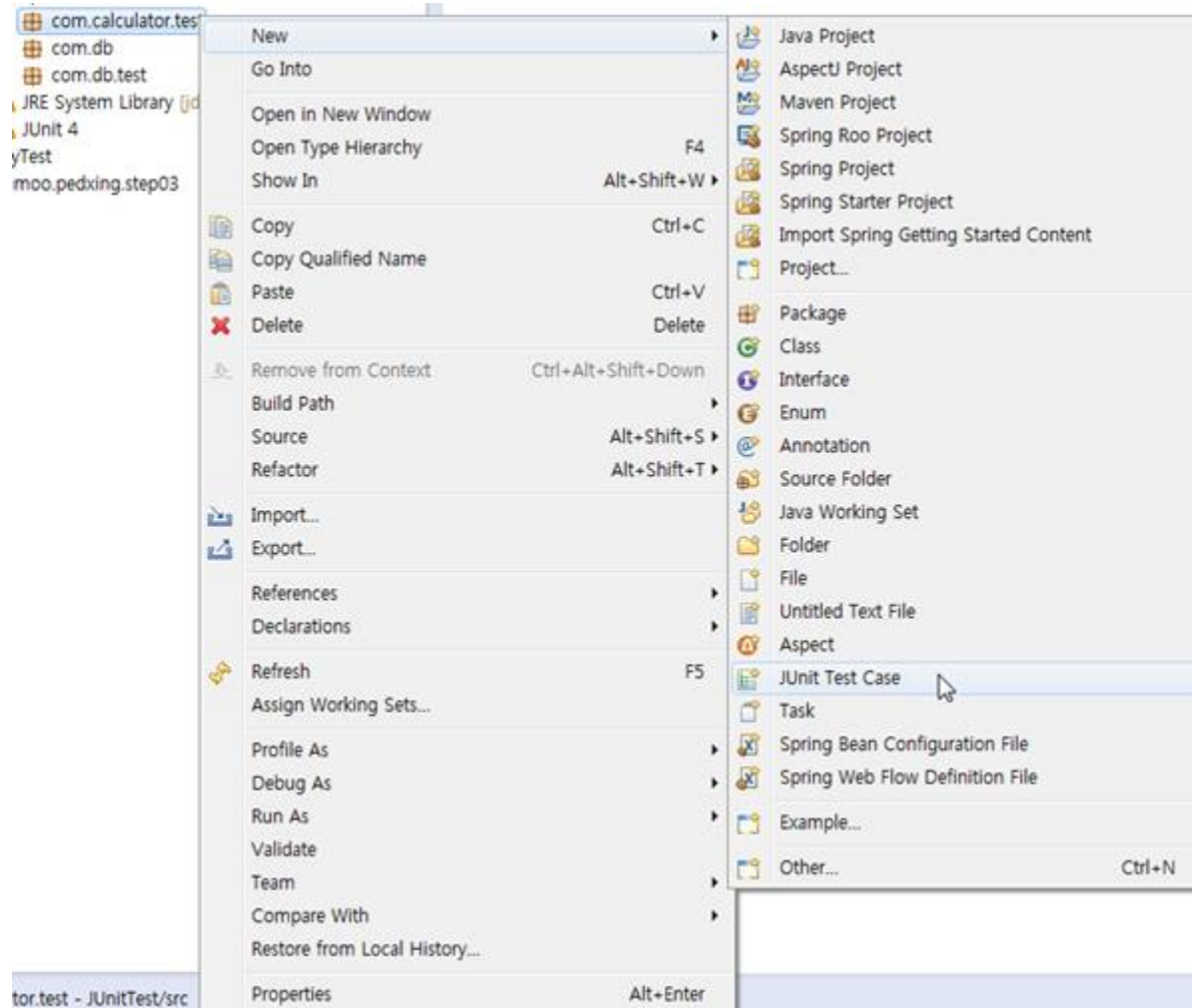


Installation

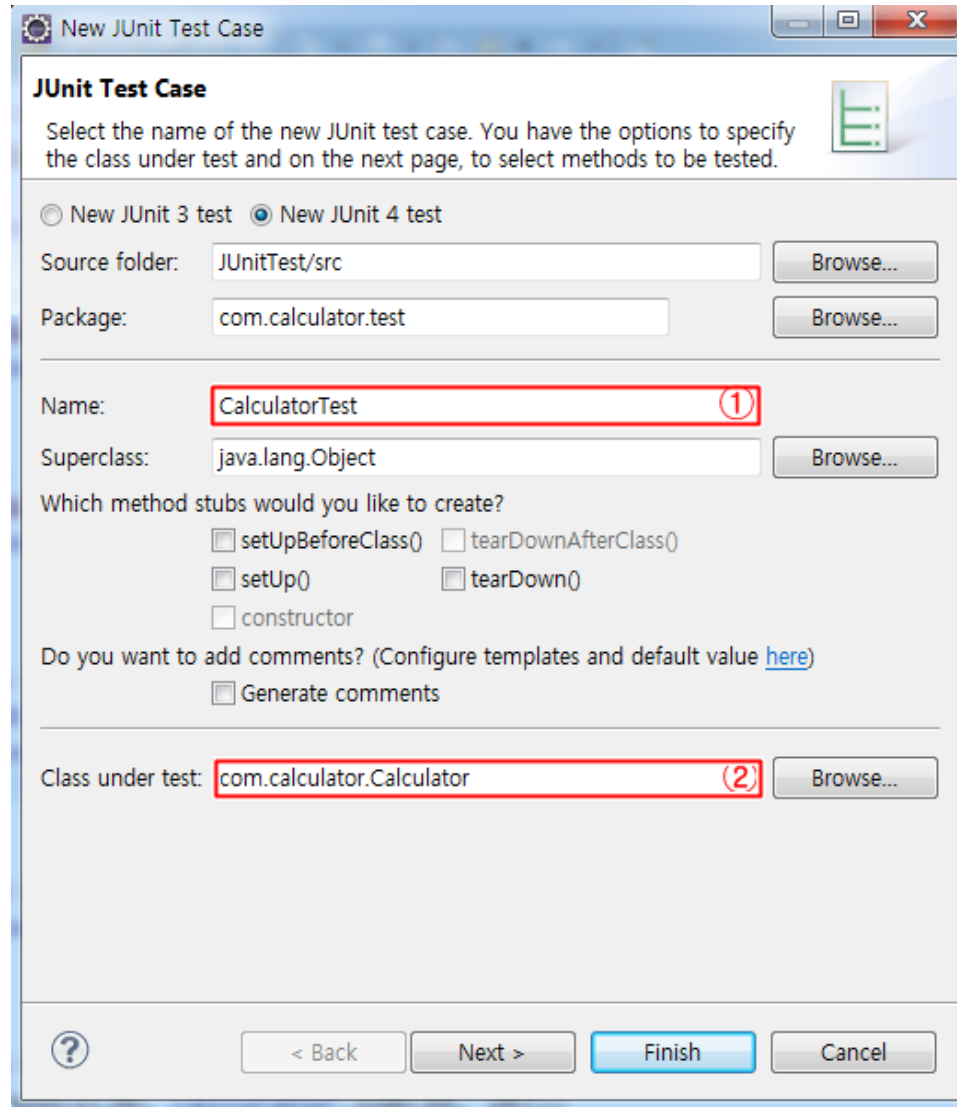


Installation

- JUnit을 사용하기 위한 test class 생성



Installation



JUnit Test Case

Select the name of the new JUnit test case. You have the options to specify the class under test and on the next page, to select methods to be tested.

New JUnit 3 test
 New JUnit 4 test

Source folder:

Package:

Name: (1)

Superclass:

Which method stubs would you like to create?

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

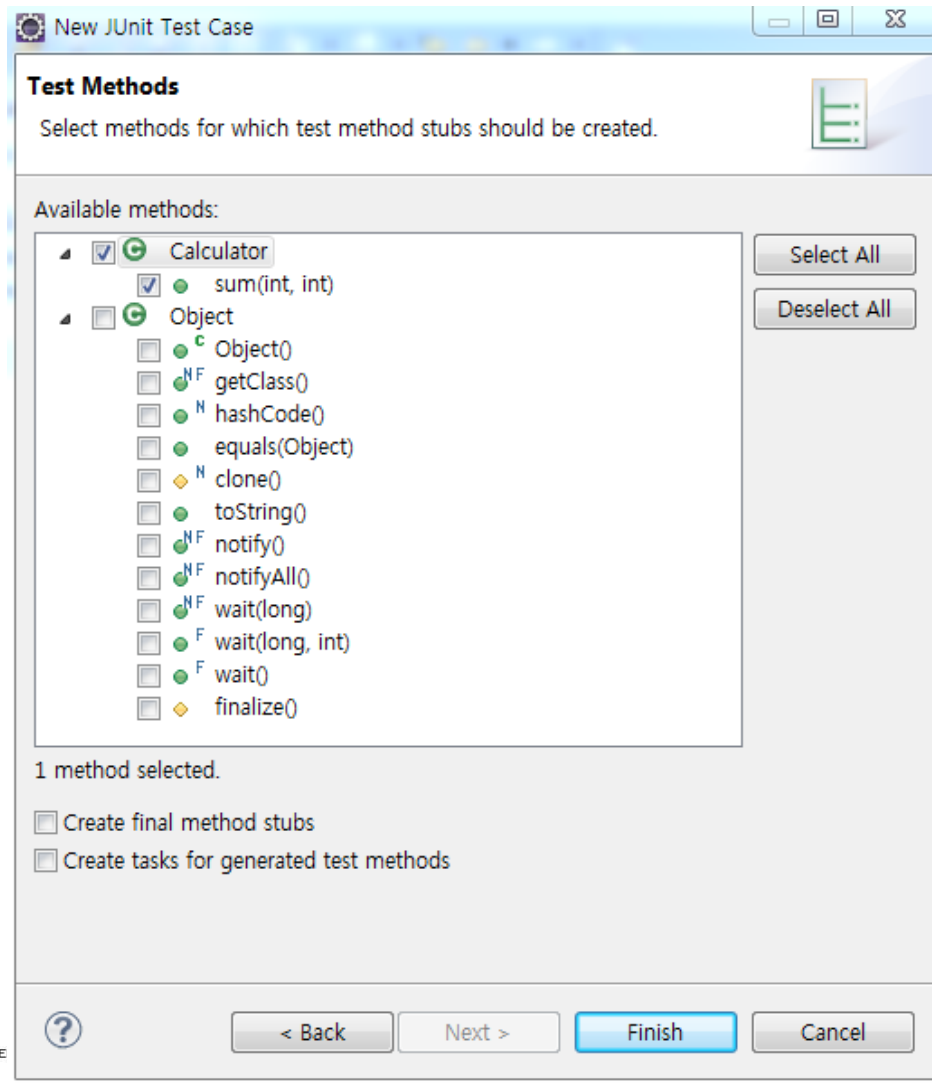
Do you want to add comments? (Configure templates and default value [here](#))

Generate comments

Class under test: (2)

Installation

- 생성 결과



```
import static org.junit.Assert.*;

public class calculatorTest {

    @Test
    public void testSum() {

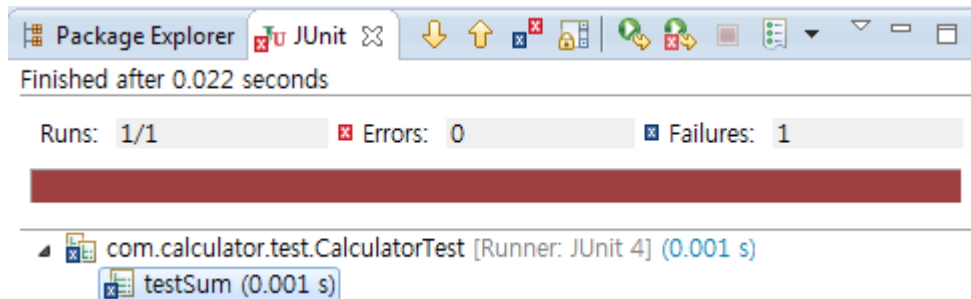
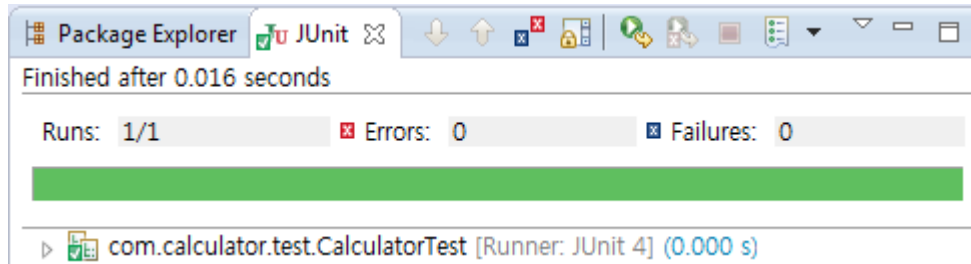
    }

}
```



Installation

- 실행 예제



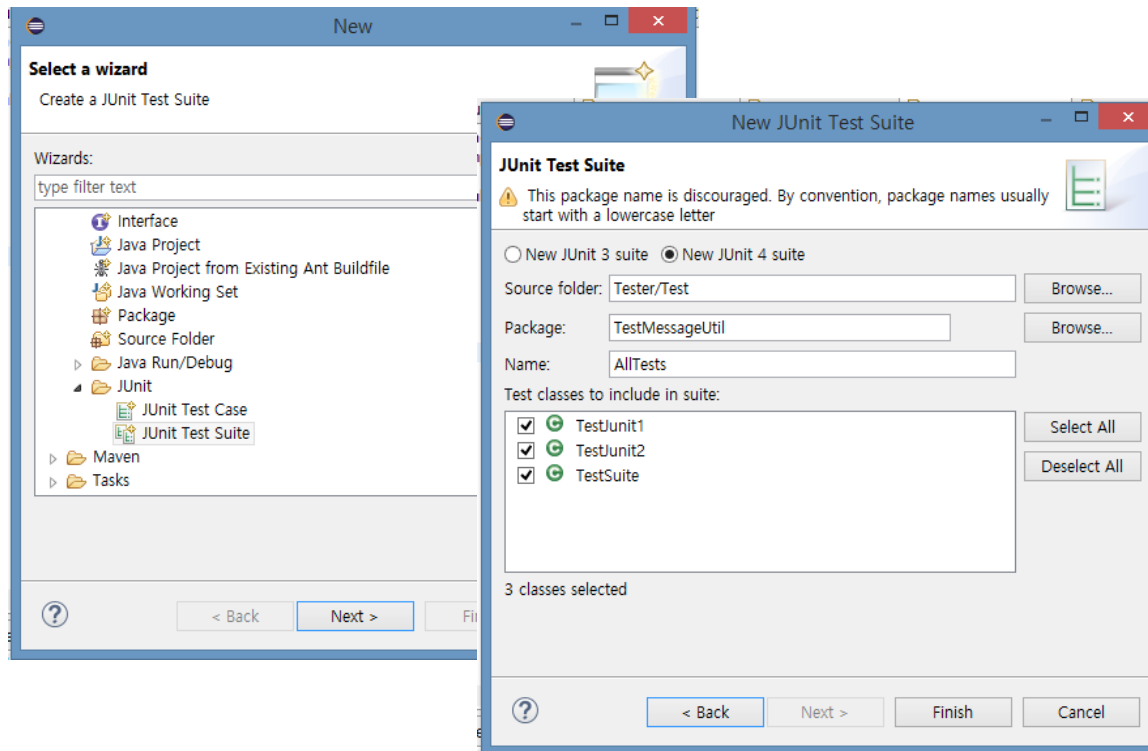
```
import static org.junit.Assert.*;

public class calculatorTest {

    @Test
    public void testSum() {
        Calculation cal = new Calculation();
        assertEquals(30, cal.sum(10,20));
        //fail("Not yet implemented");
    }
}
```

Installation

- Test Suite
 - 여러 단위의 test method들의 집합을 실행



```

1 import org.junit.runner.RunWith;
2 import org.junit.runners.Suite;
3 @RunWith(Suite.class)
4 @Suite.SuiteClasses({
5     TestJUnit1.class,
6     TestJUnit2.class
7 })
8 public class TestSuite {
9 }
    
```