


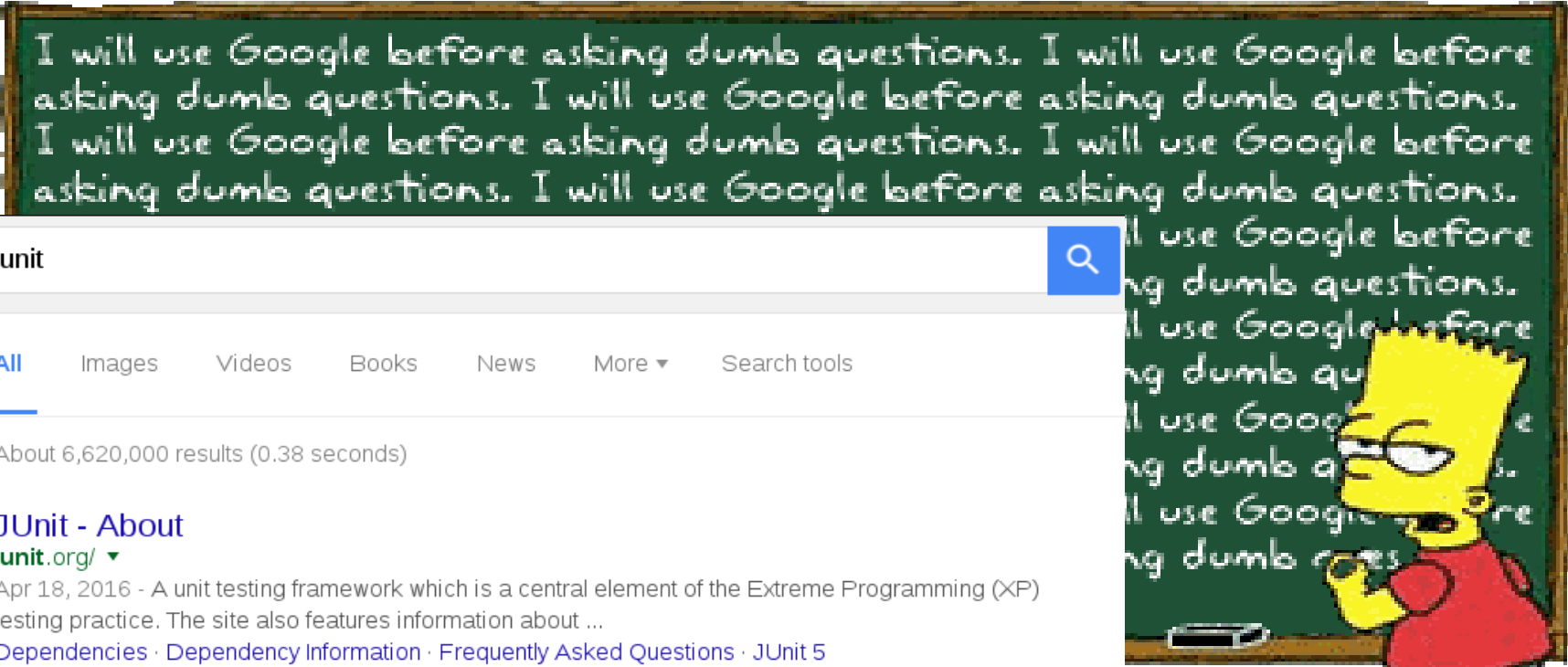
Junit

Presentation & Tools
(Eclipse, Maven, Mockito, Spring)

arnaud.nauwynck@gmail.com

This document:
<http://arnaud.nauwynck.free.fr/>

What is JUnit ?



Google

junit

All Images Videos Books News More ▾ Search tools

About 6,620,000 results (0.38 seconds)

JUnit - About
junit.org/ ▾
Apr 18, 2016 - A unit testing framework which is a central element of the Extreme Programming (XP) testing practice. The site also features information about ...
[Dependencies](#) · [Dependency Information](#) · [Frequently Asked Questions](#) · [JUnit 5](#)

JUnit - Wikipedia
<https://en.wikipedia.org/wiki/JUnit> ▾
JUnit is a unit testing framework for the Java programming language. **JUnit** has been important in the development of test-driven development, and is one of a ...
[Example of JUnit test fixture](#) · [Ports](#) · [See also](#) · [References](#)

Maven Repository: junit » junit
<https://mvnrepository.com/artifact/junit/junit> ▾
... Mapping · PDF Libraries · Top Categories · Home » **junit** » **junit**. **JUnit**. **JUnit** is a unit testing framework for Java, created by Erich Gamma and Kent Beck.

Used By: 49,142 **Categories:** [Testing Frameworks](#)
Tags: [testing](#)

Wikipedia JUnit



WIKIPEDIA
The Free Encyclopedia

[Main page](#)
[Contents](#)
[Featured content](#)
[Current events](#)
[Random article](#)
[Donate to Wikipedia](#)
[Wikipedia store](#)

Interaction
[Help](#)
[About Wikipedia](#)
[Community portal](#)
[Recent changes](#)
[Contact page](#)

Tools
[What links here](#)
[Related changes](#)
[Upload file](#)
[Special pages](#)
[Permanent link](#)
[Page information](#)
[Wikidata item](#)
[Cite this page](#)

Print/export
[Create a book](#)

Not logged in [Talk](#) [Contributions](#) [Create account](#) [Log in](#)

Article [Talk](#)

Read [Edit](#) [View history](#)



JUnit

From Wikipedia, the free encyclopedia

Not to be confused with [G-Unit](#).

"[Junit](#)" redirects here. For the Egyptian goddess, see [Junit \(goddess\)](#).

JUnit is a [unit testing framework](#) for the [Java programming language](#). JUnit has been important in the development of [test-driven development](#), and is one of a family of [unit testing](#) frameworks which is collectively known as [xUnit](#) that originated with [SUnit](#).

JUnit is linked as a [JAR](#) at compile-time; the framework resides under package `junit.framework` for JUnit 3.8 and earlier, and under package `org.junit` for JUnit 4 and later.

A research survey performed in 2013 across 10,000 Java projects hosted on GitHub found that JUnit, (in a tie with [slf4j-api](#)), was the most commonly included external library. Each library was used by 30.7% of projects. ^[3]

Contents [\[hide\]](#)

- [Example of JUnit test fixture](#)
- [Ports](#)
- [See also](#)
- [References](#)
- [External links](#)

Example of JUnit test fixture [\[edit \]](#)

JUnit

Developer(s)	Kent Beck , Erich Gamma , David Saff , Mike Clark (University of Calgary)
Stable release	4.12 ^[1] / <div>December 4, 2014</div>
Preview release	5 Milestone 2 / <div>July 23, 2016</div>
Repository	github.com / junit-team/junit4
Written in	Java
Operating system	Cross-platform
Type	Unit testing tool
License	Eclipse Public License ^[2] (relicensed from CPL before)
Website	junit.org

Junit birth

Mid-90's

Kent Beck developed xUnit test tool for Smalltalk



1997

Beck and Gamma (authors of design patterns “Gof”) **developed JUnit**

... on a flight from Zurich to Washington, D.C.



Written in few hours => The code is small, simple

How Small is Junit ?

~200 java source files

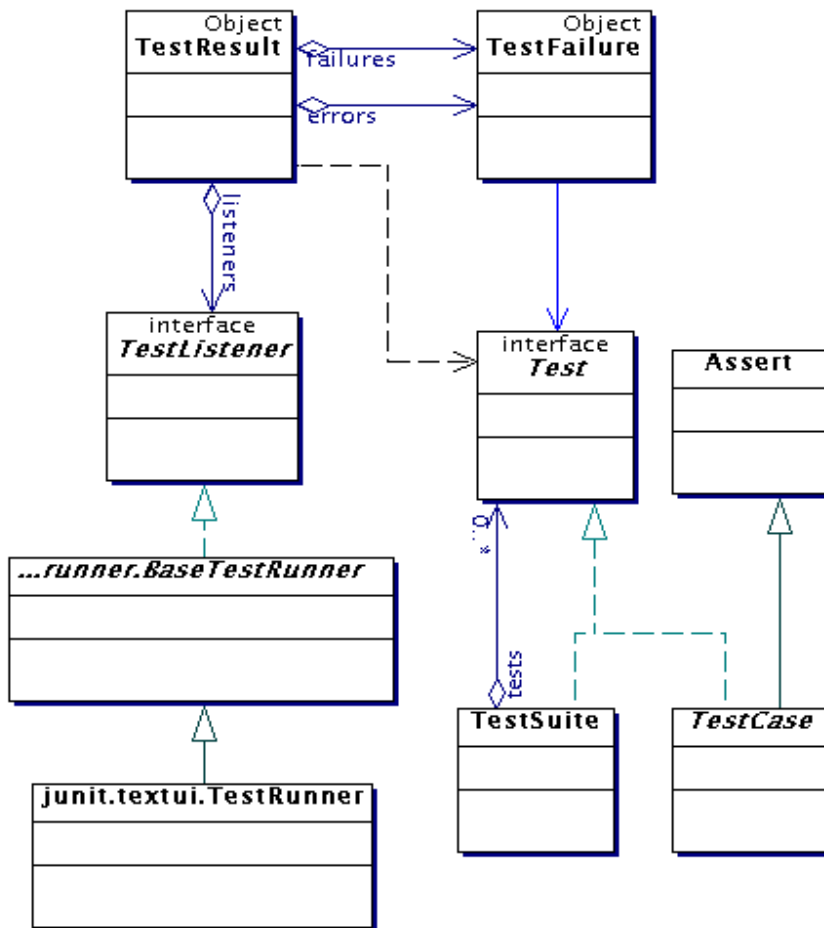
```
$ git clone https://github.com/junit-team/junit4.git  
$ cd junit4
```

```
$ find src/main/java -name '*.java' | wc -l  
204
```

```
$ find src/main/java -name '*.java' | xargs cat | wc -l  
18136
```

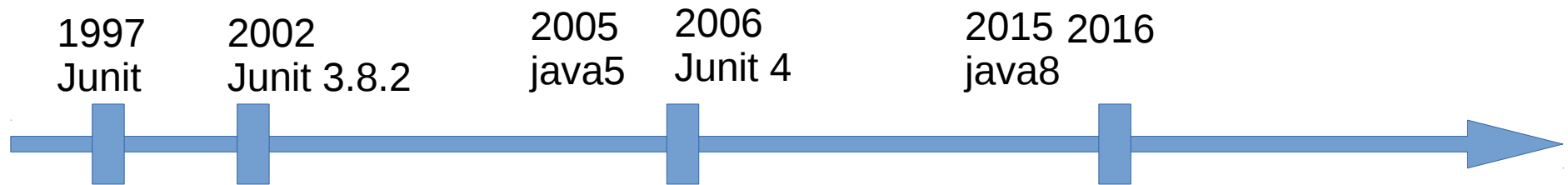
```
$ ... | grep -v '^$' | grep -v '^$' | wc -l  
11958
```

~12 000 lines of code
+ 6 000 lines of javadoc



Junit History : 3 < 4 < 5

The code has not changed for years (annotation with Java 5)



JUnit 3

```
class MyTest extends TestCase {  
  
    public void testFoo() {  
        Assert.assertEquals(...);  
    }  
  
}
```

JUnit 4

supports Java 5
Annotations

```
@Test  
public void foo() {  
}
```

No extends TestCase

JUnit5 in alpha version

Small but Powerfull

“Never in the field of software development was so much owed by so many to so few lines of code.”



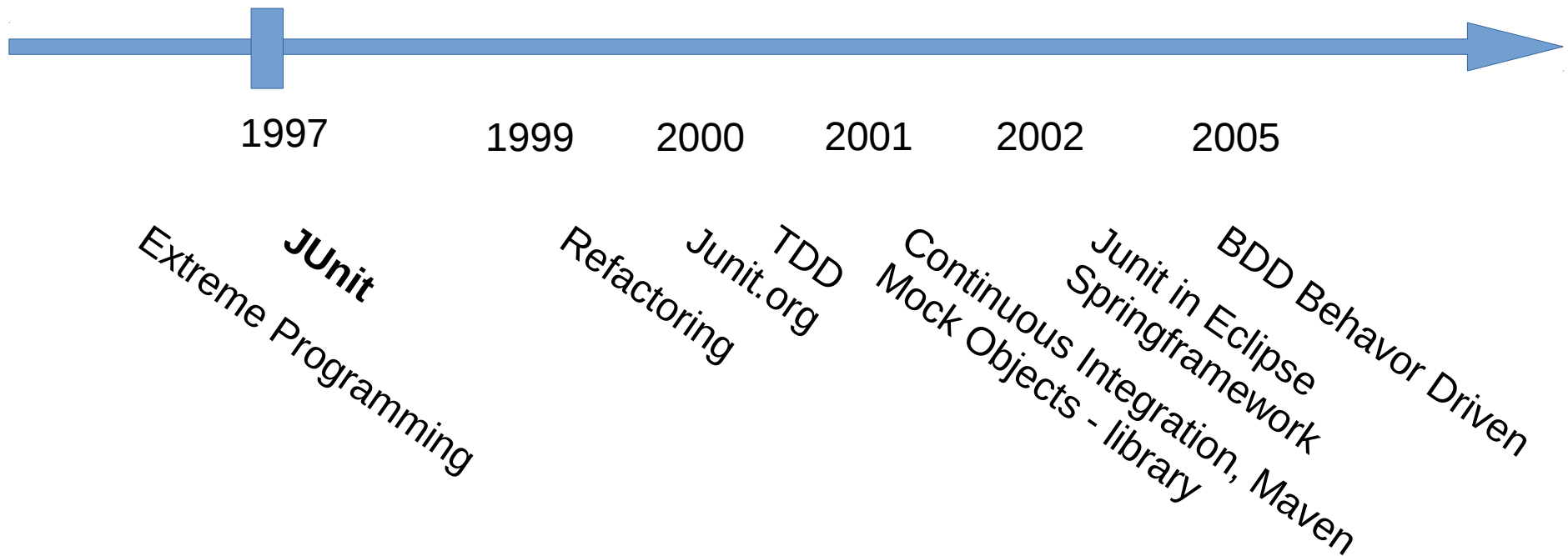
Martin Fowler

De Facto Standard

Used by ~30% major
Github projects

Integrated in ALL Tools

Junit impacts .. eXtreme Programing, TDD, BDD, Refactoring, Clean Code, Mock, IOC Injection ...



<http://www.martinfowler.com/bliki/Xunit.html>

<http://wiki.c2.com/?TenYearsOfTestDrivenDevelopment>

xUnit

JUnit ... **J**=Java

but not only Java ...

xUnit = ports in all languages

Standard in ALL languages

Ports [\[edit \]](#)

JUnit alternatives have been [written in other languages](#) including:

- [Actionscript](#) ([FlexUnit](#))
- [Ada](#) ([AUnit](#))
- [C](#) ([CUnit](#))
- [C#](#) ([NUnit](#))
- [C++](#) ([CPPUnit](#), [CxxTest](#))
- [Coldfusion](#) ([MXUnit](#))
- [Erlang](#) ([EUnit](#))
- [Eiffel](#) ([Auto-Test](#)) - JUnit inspired getest (from Gobosoft), which
- [Fortran](#) ([fUnit](#), [pFUnit](#))
- [Delphi](#) ([DUnit](#))
- [Free Pascal](#) ([FPCUnit](#))
- [Haskell](#) ([HUnit](#))
- [JavaScript](#) ([JSUnit](#))
- [Microsoft .NET](#) ([NUnit](#))
- [Objective-C](#) ([OCUnit](#))
- [OCaml](#) ([OUnit](#))
- [Perl](#) ([Test::Class](#) and [Test::Unit](#))
- [PHP](#) ([PHPUnit](#))
- [Python](#) ([PyUnit](#))
- [Qt](#) ([QTestLib](#))
- [R](#) ([RUnit](#))
- [Ruby](#) ([Test::Unit](#))

http://junit.org/junit4/

JUnit 4 ▾ Project Documentation ▾

Fork me on GitHub



[JUnit 4](#) / [About](#)

Version: 4.12 | Last Published: 2016-04-18

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

```
@Test
public void newArrayListsHaveNoElements() {
    assertThat(new ArrayList().size(), is(0));
}
```

```
@Test
public void sizeReturnsNumberOfElements() {
    List instance = new ArrayList();
    instance.add(new Object());
    instance.add(new Object());
    assertThat(instance.size(), is(2));
}
```

Annotations

Start by marking your tests with `@Test`.

Let's take a tour »

Welcome

- [Download and install](#)
- [Getting started](#)
- [Release Notes](#)
 - [4.12](#)
 - [4.11](#)
 - [4.10](#)
 - [4.9.1](#)

Usage and Idioms

- [Assertions](#)
- [Test Runners](#)
- [Aggregating tests in Suites](#)
- [Test Execution Order](#)
- [Exception Testing](#)
- [Matchers and assertThat](#)
- [Ignoring Tests](#)

Third-party extensions

- [Custom Runners](#)
- [net.trajano.commons:commons-testing](#) for [UtilityClassTestUtil](#) per #646
- [System Rules](#) – A collection of JUnit rules for testing code that uses `java.lang.System`.
- [JUnit Toolbox](#) - Provides runners for

https://github.com/junit-team/junit4

 junit-team / junit4

603 5,402 2,107

133 24 0

A programmer-oriented testing framework for Java. <http://junit.org/junit4/>

 2,127 commits

 6 branches





 20 releases

 129 contributors

 EPL-1.0

Branch: master ▾

 PeterWippermann committed with kcooney Test for #1320 - Description produced by Request.classes() shouldn't ... Latest commit 23b0bb0 17 hours ago

 .settings	Revert on request of kcooney:	3 years ago
 doc	Fix dead link to blog post by Joe Walnes about assertThat (#1210)	7 months ago
 lib	Add Hamcrest source JAR for easy reference	4 years ago
 src	Test for #1320 - Description produced by Request.classes() shouldn't ...	17 hours ago

http://junit.org/junit5/



[JUnit 4](#)

The new major version of the programmer-friendly testing framework for Java 8



User Guide



Javadoc



Code & Issues



Q & A

About

JUnit 5 is the next generation of JUnit. The goal is to create an up-to-date foundation for developer-side testing on the JVM. This includes focusing on Java 8 and above, as well as enabling many different styles of testing.

JUnit 5 is the result of [JUnit Lambda](#) and its [crowdfunding campaign on Indiegogo](#).

The JUnit 5 team released [Milestone 2](#) on July 23, 2016, and is currently working on additional milestones and ultimately a GA release (due late 2016).

Resources

You're invited to follow our ongoing work, review it and give feedback. This short list of links will get you started:

- [User Guide](#)
- [Javadoc](#)
- [GitHub Repository](#) with all the code and issues

Upcoming Events

2016-11-09

Deep Dive into JUnit 5 - Devvxx Belgium 2016 in Antwerp, Belgium

Sam Brannen

https://github.com/junit-team/junit5

[Personal](#)[Open source](#)[Business](#)[Explore](#)[Pricing](#)[Blog](#)[Support](#)[This repository](#)[Sign in](#)[Sign up](#)[junit-team](#) / [junit5](#)[Watch](#)

130

[★ Star](#)

687

[Fork](#)

151

[Code](#)[Issues](#) 111[Pull requests](#) 10[Projects](#) 0[Wiki](#)[Pulse](#)[Graphs](#)

The next generation of JUnit. <http://junit.org/junit5/>

[2,661](#) commits[11](#) branches[5](#) releases[30](#) contributors[EPL-1.0](#)Branch: **master** ▾[New pull request](#)[Find file](#)[Clone or download ▾](#)**marcphilipp** Disable default JAR task to prevent it overwriting the shadowJar ...

Latest commit 0818031 9 days ago

.github	Fixed spotless issue	8 months ago
buildSrc	Inline jopt-simple into junit-platform-console	9 days ago
documentation	Document inlining of jopt-simple into junit-platform-console	9 days ago
gradle	Make DeGraph check faster and more user friendly (#532)	29 days ago
junit-jupiter-api	Introduce @Testable to facilitate test discovery in IDEs	25 days ago
junit-jupiter-engine	Inline jopt-simple into junit-platform-console	9 days ago

Getting Started : New Project Eclipse + Maven

New Maven Project

New Maven Project

New Maven project

Enter a location for the project.

☒ Create a simple project (skip archetype selection)

☐ Use default Workspace location

Location:

Browse...

☒ Add project(s) to working set

Working set:

junit

More...

Advanced

?

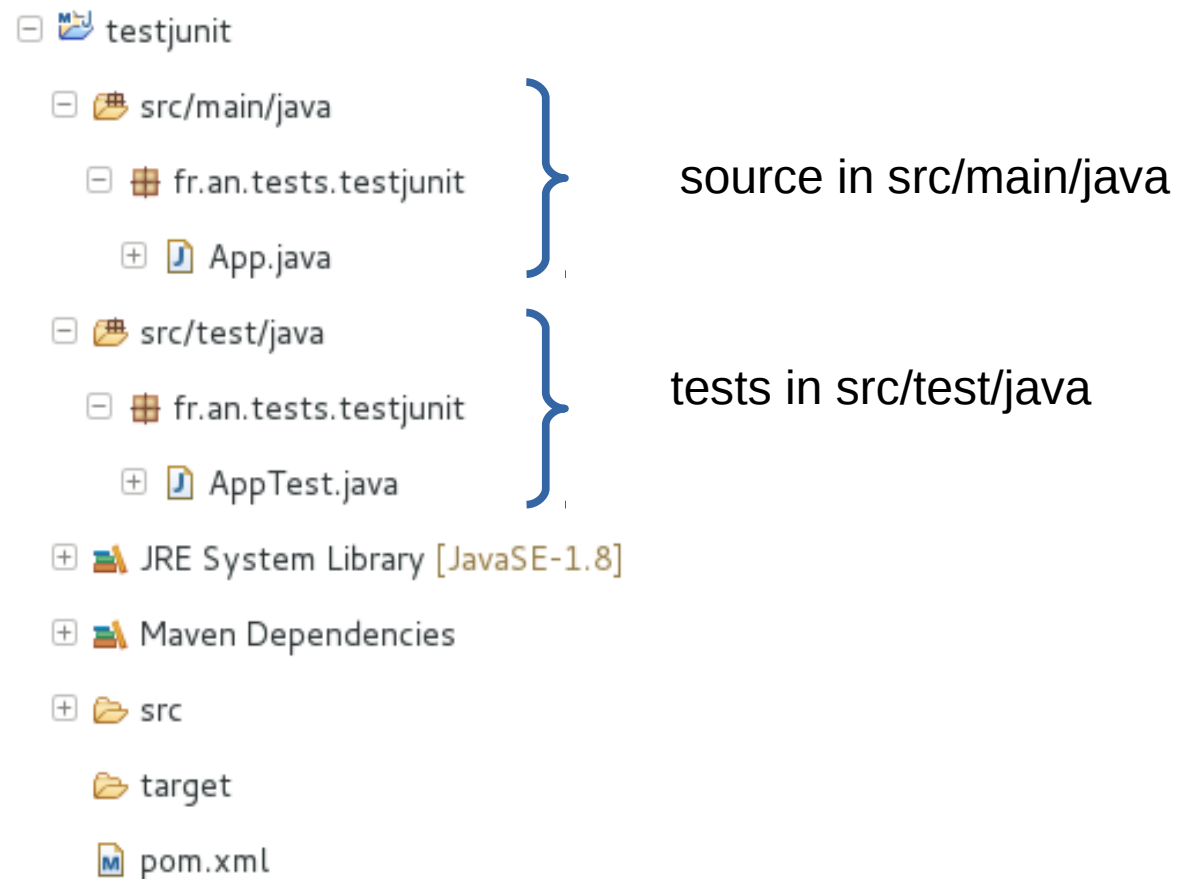
< Back

Next >

Cancel

Finish

Blank Junit Project



JUnit = Unitary Class Test

1 class => 1 Test Class

1 method => 1 Test method

SUT = System Under Test

Example of System Under Test

App.java

```
package fr.an.tests.testjunit;

import javax.script.ScriptEngine;
import javax.script.ScriptEngineManager;
import javax.script.ScriptException;

public class App {

    private static final ScriptEngine engine =
        new ScriptEngineManager().getEngineByName("nashorn");

    public static Object evalJS(String script) {
        try {
            Object res = engine.eval(script);
            return res;
        } catch (ScriptException ex) {
            throw new RuntimeException("Failed to eval script '" + script + "'", ex);
        }
    }
}
```

This method is supposed to eval any JavaScript from JRE
Does it work ??

Let's Test It – Start Small, Trivial

```
App.java  AppTest.java

package fr.an.tests.testjunit;

import org.junit.Assert;
import org.junit.Test;

/**
 * a Unit test for class App
 */
public class AppTest {

    @Test
    public void evalJS_1_plus_1() throws Exception {
        Assert.assertEquals(2, App.evalJS("1+1"));
    }

    @Test
    public void evalJS_invoke_immediate_func() throws Exception {
        Assert.assertEquals(2, App.evalJS("(function() { return 1+1; })()"));
    }

}
```

At least, it should not break
(but it could be anything else than JavaScript)

It looks Ugly
It looks real JavaScript

Junit in Eclipse

Let's Run It

The image shows a screenshot of an IDE interface. On the left, a code editor displays Java code for a unit test. A context menu is open in the center, listing various actions and their keyboard shortcuts. On the right, a test runner interface shows the execution of a JUnit test, including a progress bar and a failure trace.

```
/**
 * a Unit test for
 */
public class AppTest {

    @Test
    public void testAssert() {
        Assert.assertEquals(1, 1);
    }

    @Test
    public void testAssertException() {
        Assert.assertEquals(1, 1);
    }
}
```

Context Menu:

- Show in Breadcrumb (Shift+Alt+B)
- Quick Outline (Ctrl+O)
- Quick Type Hierarchy (Ctrl+T)
- Open With >
- Show In (Shift+Alt+W) >
- Cut (Ctrl+X)
- Copy (Ctrl+C)
- Copy Qualified Name
- Paste (Ctrl+V)
- Quick Fix (Ctrl+I)
- Source (Shift+Alt+S) >
- Refactor (Shift+Alt+T) >
- Local History >
- References >
- Declarations >
- Add to Snippets...
- Run As >
- Debug As >**
- Profile As >
- Validate
- GitHub >
- MoreUnit >
- Team >
- Compare With >
- Replace With >

Test Runner Interface:

JUnit Test (Shift+Alt+D T)

Debug Configurations...

Failure Trace

JUnit Result View

The screenshot displays the Eclipse IDE interface during a JUnit test run. The top toolbar shows icons for Debug, Breakpoints, Expressions, and Registers. The Debug console at the top left shows the test runner's execution path and exit status.

The Java editor in the center shows the source code for `AppTest.java`:

```
import org.junit.Assert;

/**
 *
 */
public class AppTest {

    @Test
    public void evalJS_1_plus_1() throws Exception {
        Assert.assertEquals(2, App.evalJS("1+1"));
    }
}
```

The bottom panel, titled "JUnit", shows the test results. It indicates that the test finished after 0.381 seconds, with 2/2 runs, 0 errors, and 0 failures. A green progress bar is visible. The test results list shows two tests: `evalJS_1_plus_1` (0.356 s) and `evalJS_invoke_immediate_func` (0.003 s).

JUnit Results Summary:

Test Name	Duration (s)
evalJS_1_plus_1	0.356
evalJS_invoke_immediate_func	0.003

Launch Shortcuts in Eclipse

F11 = execute

(On test, on main, on launch target..)

Shift+Alt+D T (=debug test)

CTRL + R (see next: MoreUnit)

Junit Results – OK / Assert / Failure

```
App.java  AppTest.java  AssertionTest.java ✕  
  
public class AssertionTest {  
    @Test  
    public void testOK() {  
        Assert.assertEquals(2, 1+1);  
    }  
  
    @Test  
    public void testError() {  
        Assert.assertEquals(42, 1+1);  
    }  
  
    @Test  
    public void testFailure() {  
        throw new RuntimeException();  
    }  
}
```

Console Display JUnit Search Call Hierarchy Progress

Finished after 0.019 seconds

Runs: 3/3 ✕ Errors: 1 ✕ Failures: 1

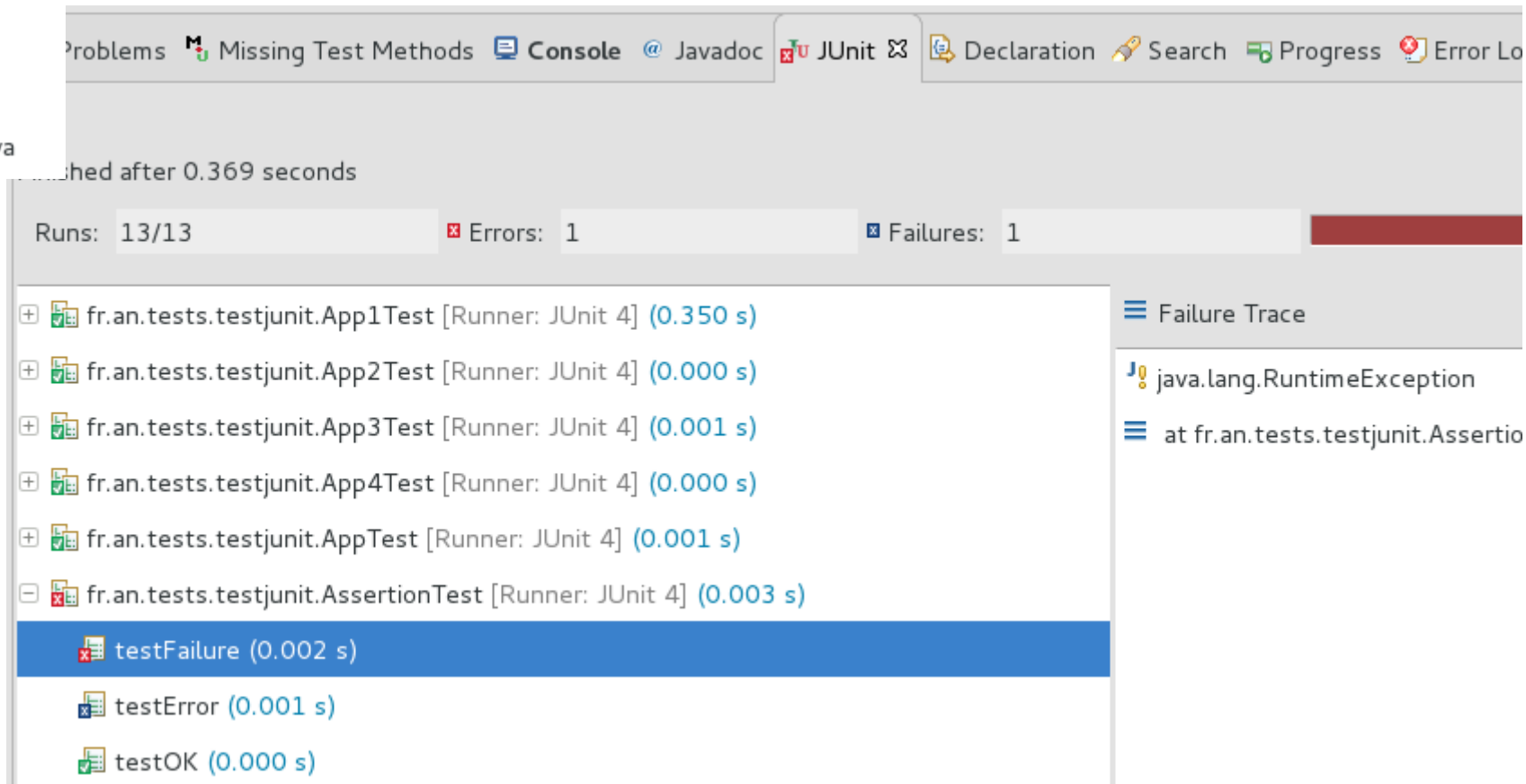
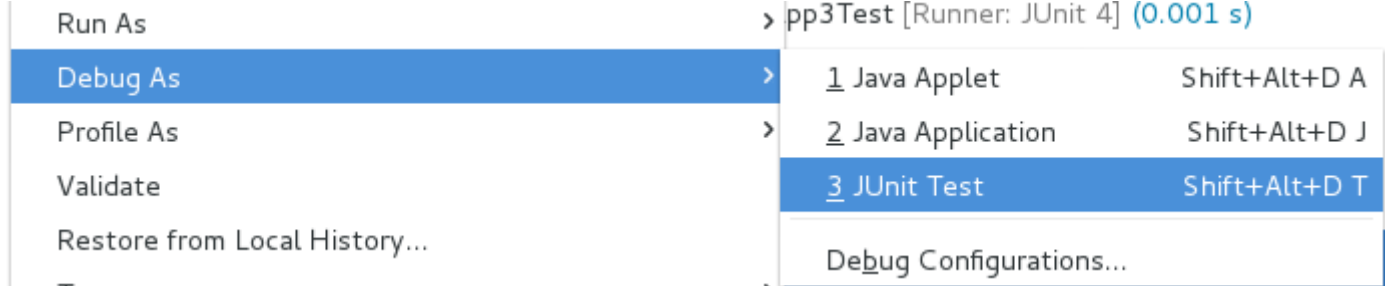
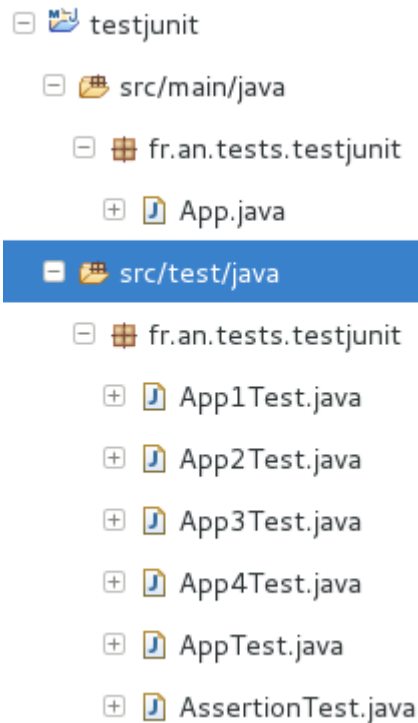
fr.an.tests.testjunit.AssertionTest [Runner: JUnit 4] (0.005 s)

- ✕ testFailure (0.002 s)
- ✕ testError (0.002 s)
- ✓ testOK (0.001 s)

Failure Trace

- java.lang.RuntimeException
- at fr.an.tests.testjunit.AssertionTest.testFailure(AssertionTest.java:20)

Run All package Tests



Junit in Maven

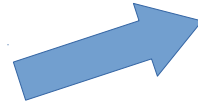
Junit dependency in Maven pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http
  <modelVersion>4.0.0</modelVersion>

  <groupId>fr.an.tests</groupId>
  <artifactId>testjunit</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.12</version>
      <scope>test</scope>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-compiler-plugin</artifactId>
        <version>3.2</version>
        <configuration>
          <source>1.8</source>
          <target>1.8</target>
        </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```



“jar” project => **built-in supports** for
compile +
compile-test + test +
jar ...



Junit.jar test compile dependency

Junit = built-in in Maven

\$ mvn install

test only:

\$ mvn test

Skip Tests in Maven

all but test:

\$ mvn install -DskipTests

In ~/.m2/
settings.xml

```
<profile>
  <id>noTest</id>
  <properties>
    <skipTests>true</skipTests>
  </properties>
</profile>

</profiles>

<activeProfiles>
  <activeProfile>noTest</activeProfile>
</activeProfiles>
```

Test Phase in Maven

Phases:

- resources
- compile
- **compile -test**
- **test**
- jar
- install

```
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ testjunit ---
[INFO] Surefire report directory: /mnt/a_1tera2/homeData/arnaud/perso/devPerso/tests/test-junit/target/surefire-reports

-----
T E S T S
-----

Running fr.an.tests.testjunit.AppTest
Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.39 sec

Results :

Tests run: 2, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:2.4:jar (default-jar) @ testjunit ---
[INFO] Building jar: /mnt/a_1tera2/homeData/arnaud/perso/devPerso/tests/test-junit/testjunit/target/testjunit-0.0.1-SNAPSHOT.jar
[INFO] META-INF/maven/fr.an.tests/testjunit/pom.xml already added, skipping
[INFO] META-INF/maven/fr.an.tests/testjunit/pom.properties already added, skipping
[INFO]
[INFO] --- maven-install-plugin:2.4:install (default-install) @ testjunit ---
[INFO] Installing /mnt/a_1tera2/homeData/arnaud/perso/devPerso/tests/test-junit/testjunit/target/testjunit-0.0.1-SNAPSHOT.jar to /home/arnaud/.m2/repository/fr/an/tests/testjunit/0.0.1-SNAPSHOT/testjunit-0.0.1-SNAPSHOT.jar
[INFO] Installing /mnt/a_1tera2/homeData/arnaud/perso/devPerso/tests/test-junit/testjunit/pom.xml to /home/arnaud/.m2/repository/fr/an/tests/testjunit/0.0.1-SNAPSHOT/testjunit-0.0.1-SNAPSHOT.pom
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
```

Junit error => Build Failure

```
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ testjunit ---
[INFO] Surefire report directory: /mnt/a_1tera2/homeData/arnaud/perso/devPerso/tests/t
ports
```

```
-----
T E S T S
-----
```

```
Running fr.an.tests.testjunit.AppTest
```

```
Tests run: 3, Failures: 1, Errors: 0, Skipped: 0, Time elapsed: 0.361 sec <<< FAILURE!
```

```
test_failing(fr.an.tests.testjunit.AppTest) Time elapsed: 0.002 sec <<< FAILURE!
```

```
java.lang.AssertionError: expected:<2> but was:<3>
```

```
    at org.junit.Assert.fail(Assert.java:88)
```

```
    at org.junit.Assert.failNotEquals(Assert.java:834)
```

```
    at org.junit.Assert.assertEquals(Assert.java:645)
```

```
    at org.junit.Assert.assertEquals(Assert.java:631)
```

```
    at fr.an.tests.testjunit.AppTest.test_failing(AppTest.java:23)
```

```
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
```

```
    ... (truncated)
```

```
    at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:75)
```

Results :

```
Failed tests:  test_failing(fr.an.tests.testjunit.AppTest): expected:<2> but was:<3>
```

```
Tests run: 3, Failures: 1, Errors: 0, Skipped: 0
```

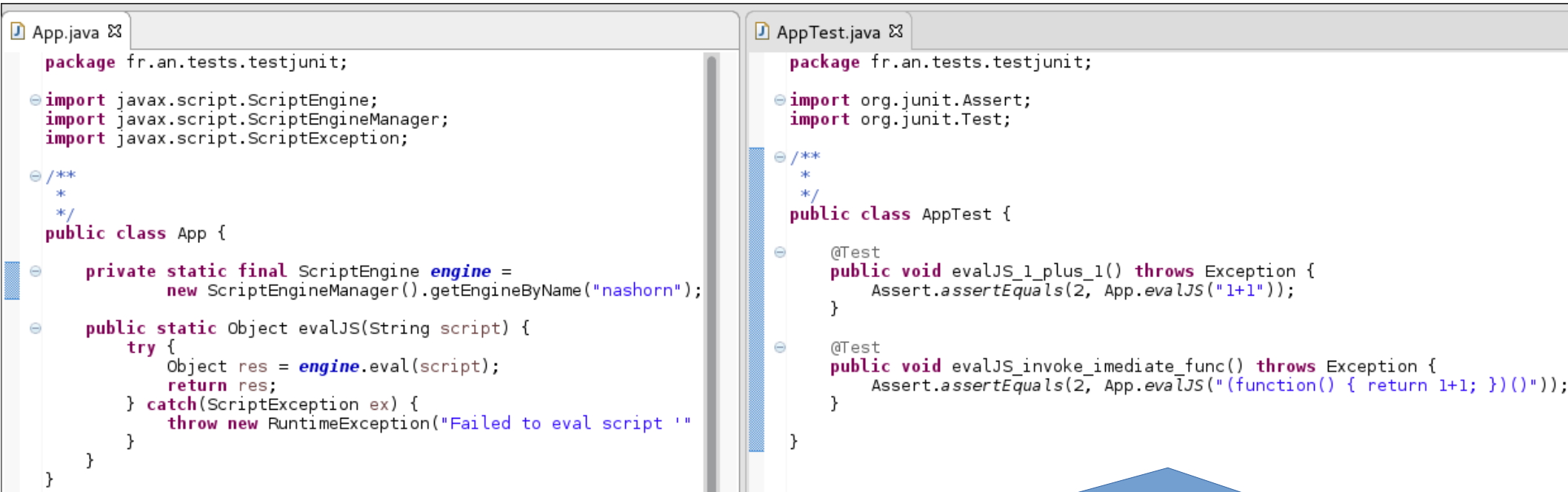
```
[INFO] -----
```

```
[INFO] BUILD FAILURE
```

```
[INFO] -----
```


MoreUnit Eclipse Plugin

Junit Naming Conventions



```
App.java
package fr.an.tests.testjunit;

import javax.script.ScriptEngine;
import javax.script.ScriptEngineManager;
import javax.script.ScriptException;

/**
 *
 */
public class App {

    private static final ScriptEngine engine =
        new ScriptEngineManager().getEngineByName("nashorn");

    public static Object evalJS(String script) {
        try {
            Object res = engine.eval(script);
            return res;
        } catch (ScriptException ex) {
            throw new RuntimeException("Failed to eval script '"
        }
    }
}

AppTest.java
package fr.an.tests.testjunit;

import org.junit.Assert;
import org.junit.Test;

/**
 *
 */
public class AppTest {

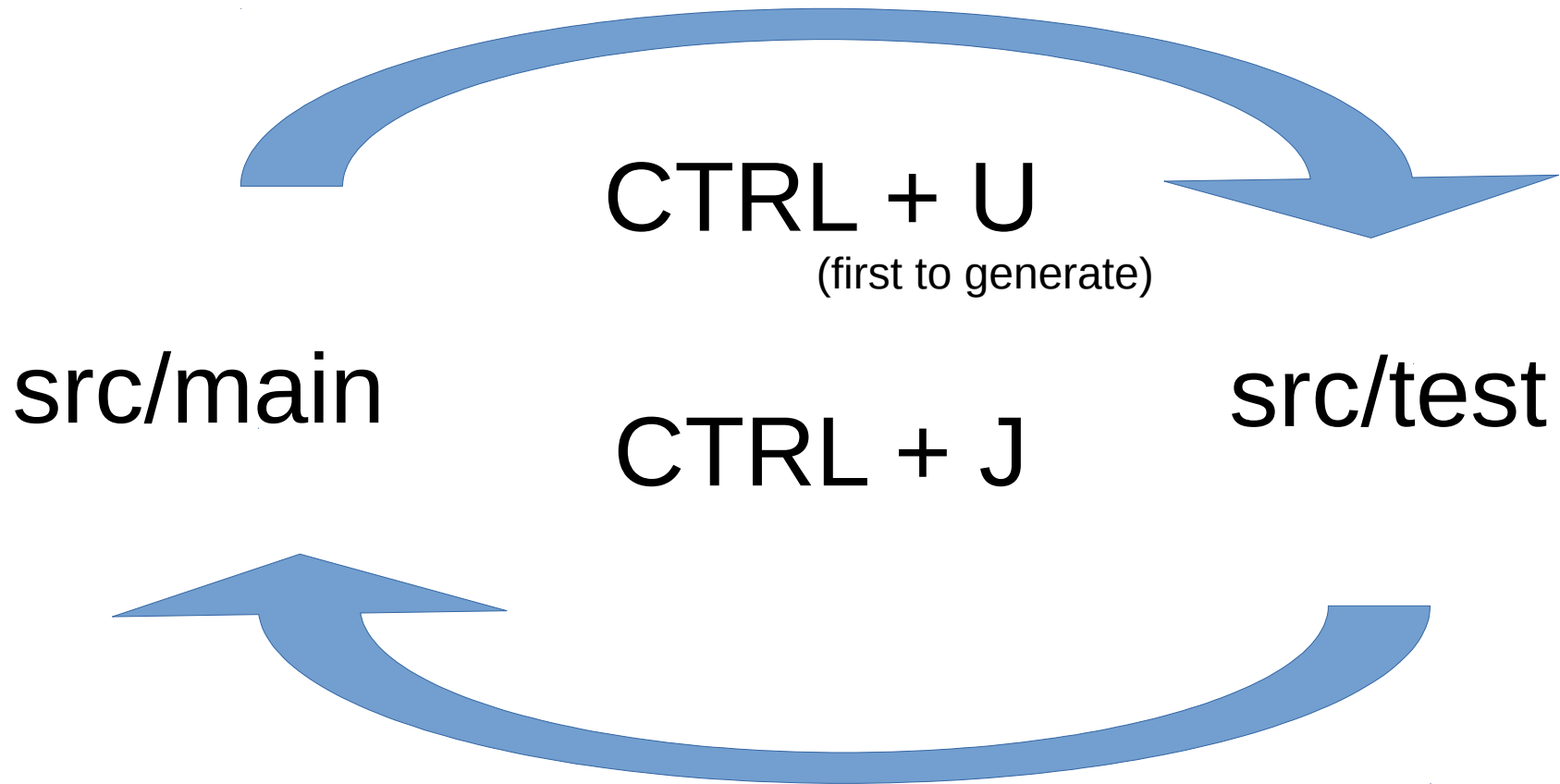
    @Test
    public void evalJS_1_plus_1() throws Exception {
        Assert.assertEquals(2, App.evalJS("1+1"));
    }

    @Test
    public void evalJS_invoke_imediate_func() throws Exception {
        Assert.assertEquals(2, App.evalJS("(function() { return 1+1; })()"));
    }
}
```

A class “App” in package “a.b.c”
in src/main/java

“AppTest” in same package “a.b.c”
in src/test/java

Eclipse .. Switching from src to test




Install Eclipse MoreUnit

Eclipse Marketplace


Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.




SearchRecentPopularFavoritesInstalledOctober Newsletter (Runtimes)

Find:




MoreUnit 3.1.1

MoreUnit is an Eclipse plugin that should assist you in writing more unit tests. It supports all programming languages (switching between tests and classes under... [more info](#))


by  EPL


[test](#) [Favorite](#) [junit](#) [testng](#) [mock ...](#)

★ 152

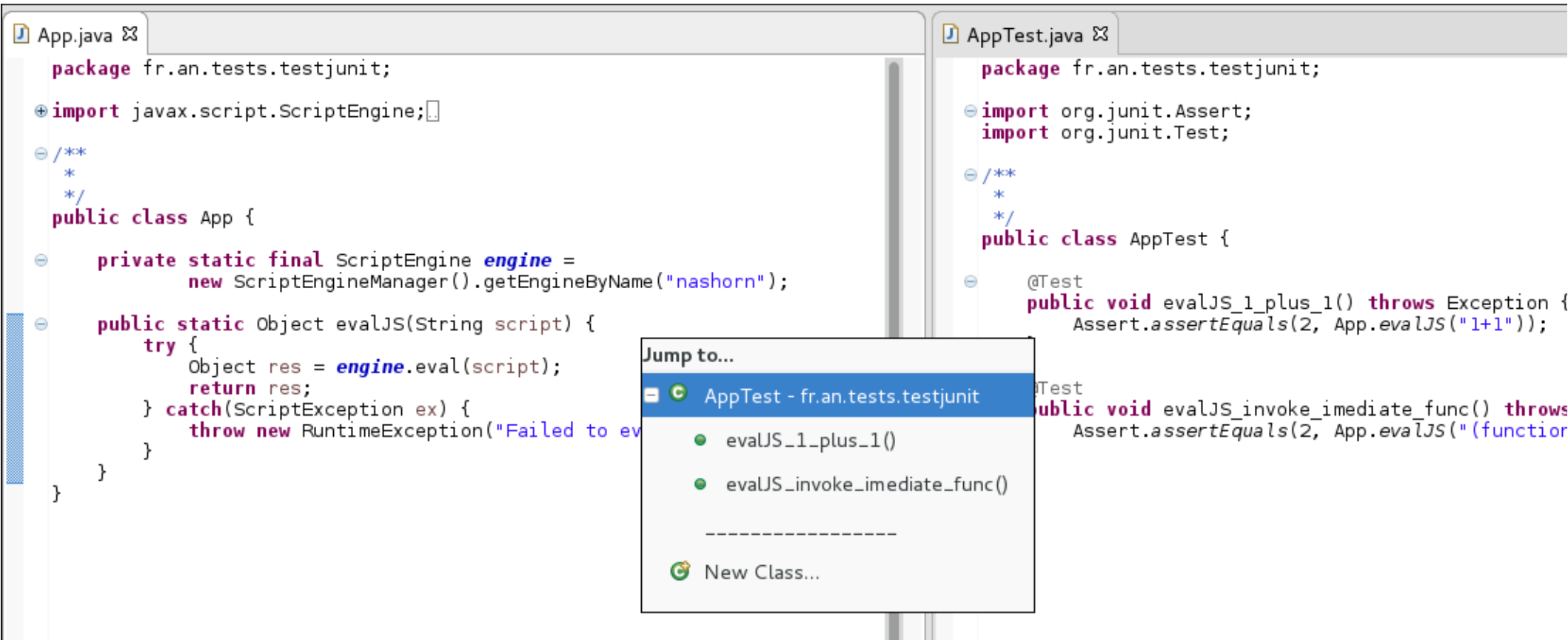
 Installs: **60.4K** (1,419 last month)

Marketplaces



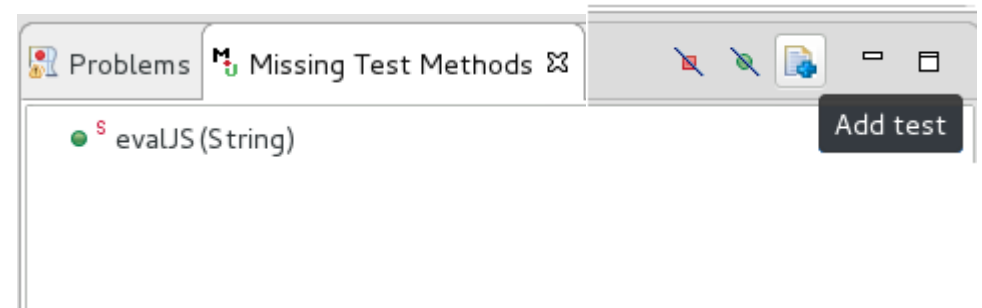
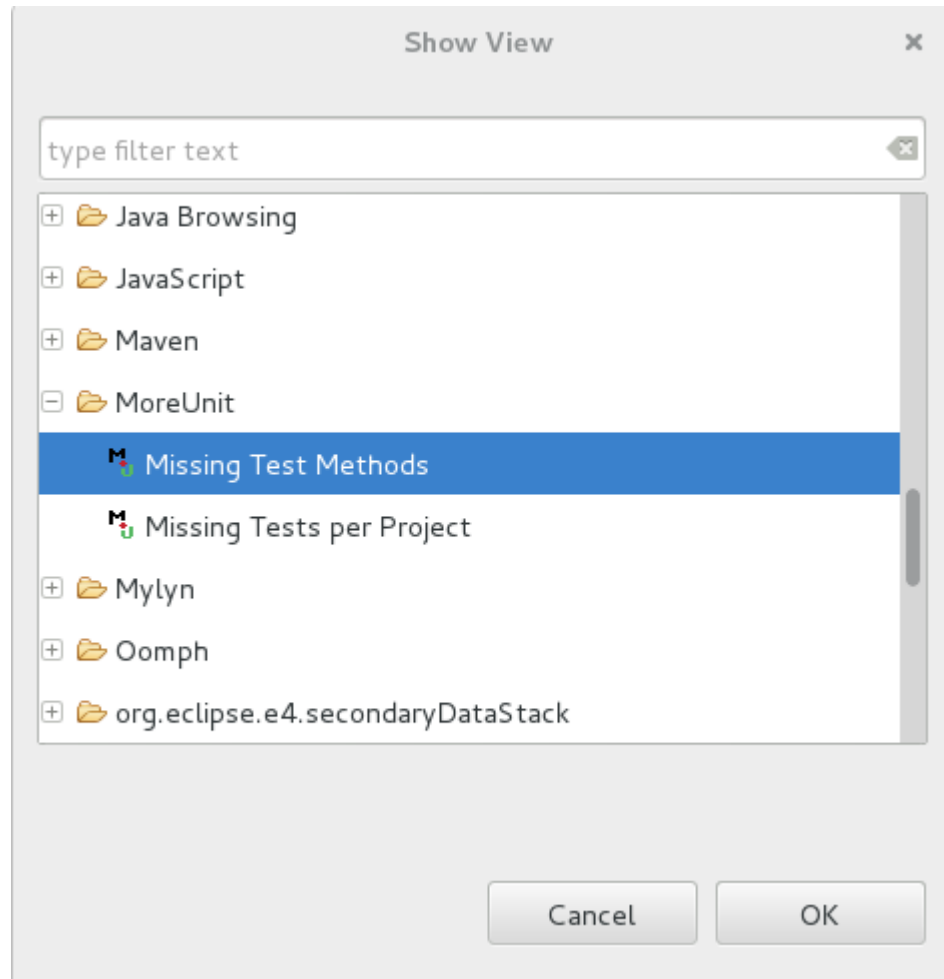


Eclipse MoreUnit plugin

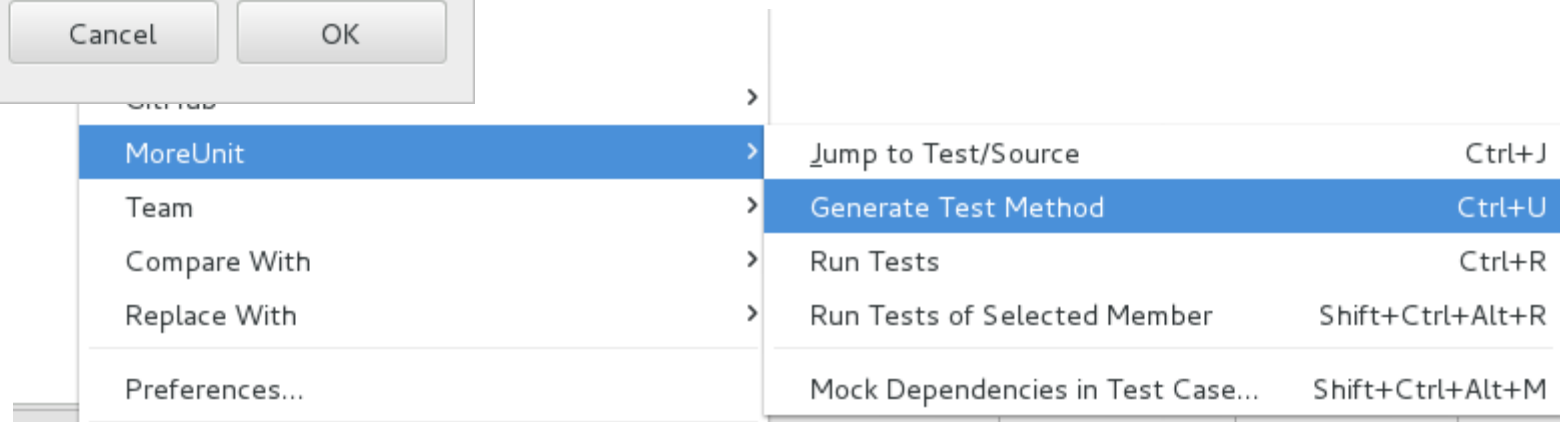


CTRL + J

MoreUnit – Missing Tests Methods



CTRL + U



InfiniTest Eclipse Plugin

InfiniTest Installation

Eclipse Marketplace

Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Search

Recent

Popular


Favorites

Installed

October Newsletter (Runtimes)

Find:

Infinittest 5.1.116




Infinittest is a continuous test runner for Java, and is valuable to developers using a unit testing tool called JUnit. Continuous testing is the practice of... [more info](#)


by [Open Source](#), MIT


[junit](#) [continuous testing](#) [unit tests](#)

★ 40

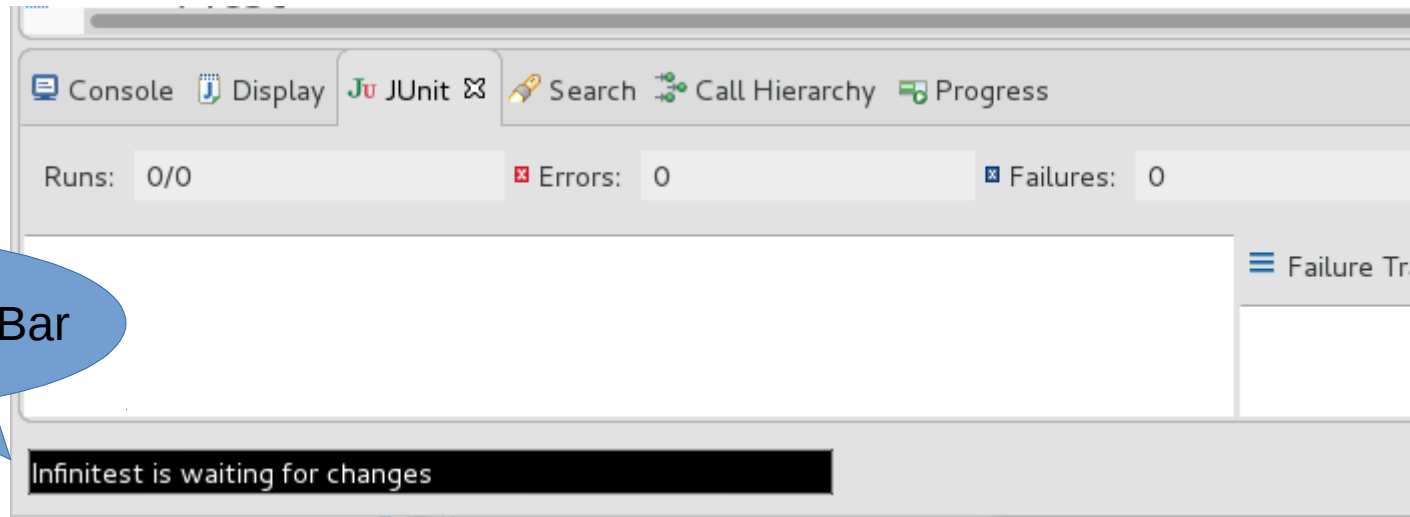
 Installs: **20.4K** (427 last month)

Marketplaces





InfiniTest Status Bar



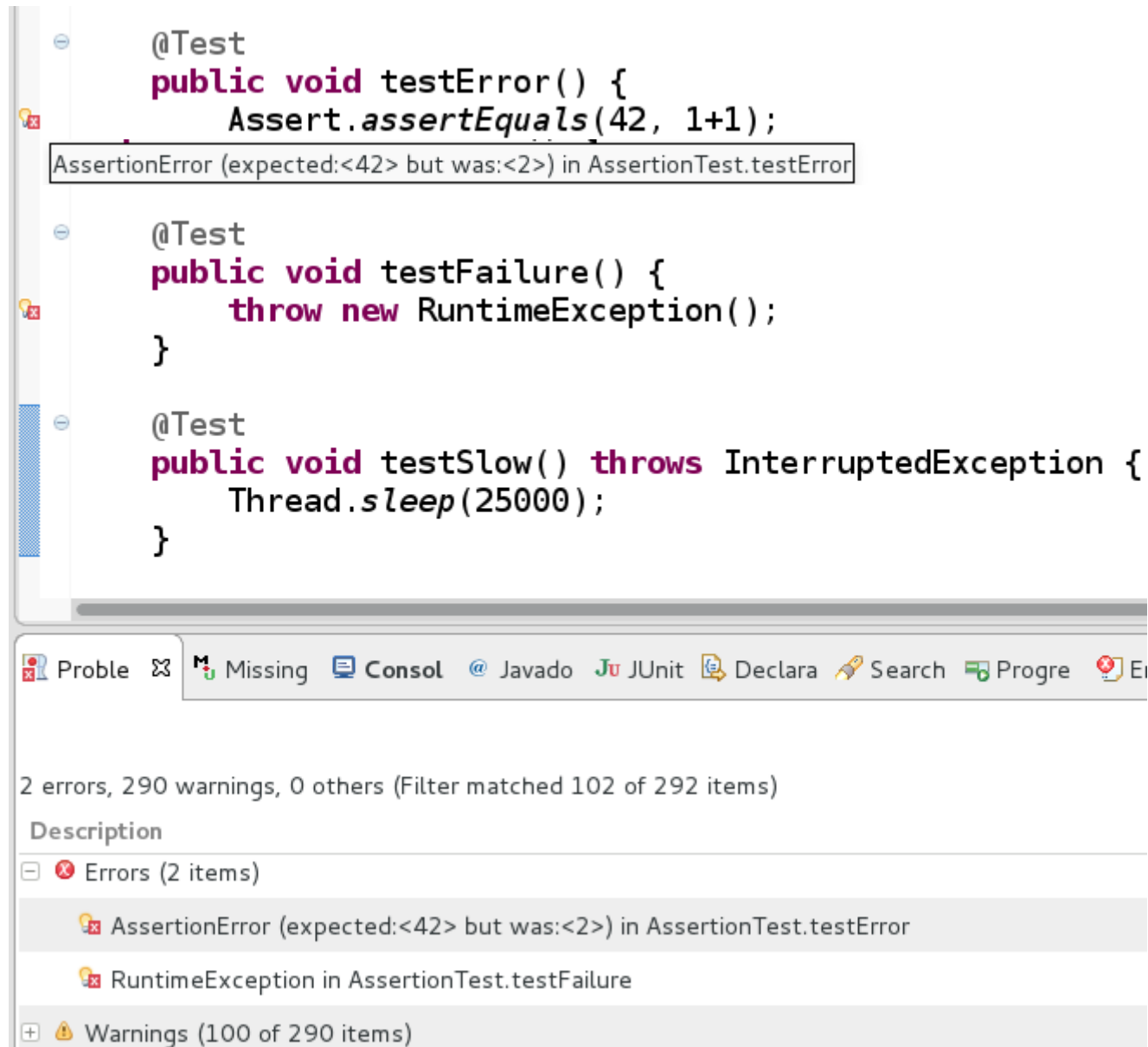
No related tests found for last change.

Running AssertionTest (0 remaining)

1 test cases ran at 9:41:43 PM

Save => InfiniTest runs Tests

Test Fails = Compile Error



The screenshot shows an IDE window with three Java test methods. The first method, `testError()`, contains `Assert.assertEquals(42, 1+1);`. A tooltip points to this line, displaying the error: `AssertionError (expected:<42> but was:<2>) in AssertionTest.testError`. The second method, `testFailure()`, throws a `RuntimeException`. The third method, `testSlow()`, calls `Thread.sleep(25000)`. Below the code editor, the 'Problems' tab is active, showing a summary: '2 errors, 290 warnings, 0 others (Filter matched 102 of 292 items)'. Under the 'Errors (2 items)' section, the first error is the same `AssertionError` from `testError()`, and the second is `RuntimeException in AssertionTest.testFailure`. The 'Warnings (100 of 290 items)' section is partially visible at the bottom.

```
@Test
public void testError() {
    Assert.assertEquals(42, 1+1);
}

@Test
public void testFailure() {
    throw new RuntimeException();
}

@Test
public void testSlow() throws InterruptedException {
    Thread.sleep(25000);
}
```

AssertionError (expected:<42> but was:<2>) in AssertionTest.testError

Problems: 2 errors, 290 warnings, 0 others (Filter matched 102 of 292 items)

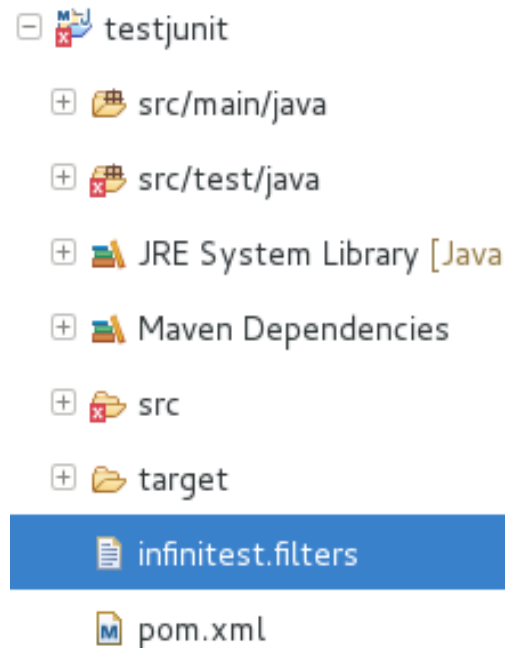
Description

Errors (2 items)

- AssertionError (expected:<42> but was:<2>) in AssertionTest.testError
- RuntimeException in AssertionTest.testFailure

Warnings (100 of 290 items)

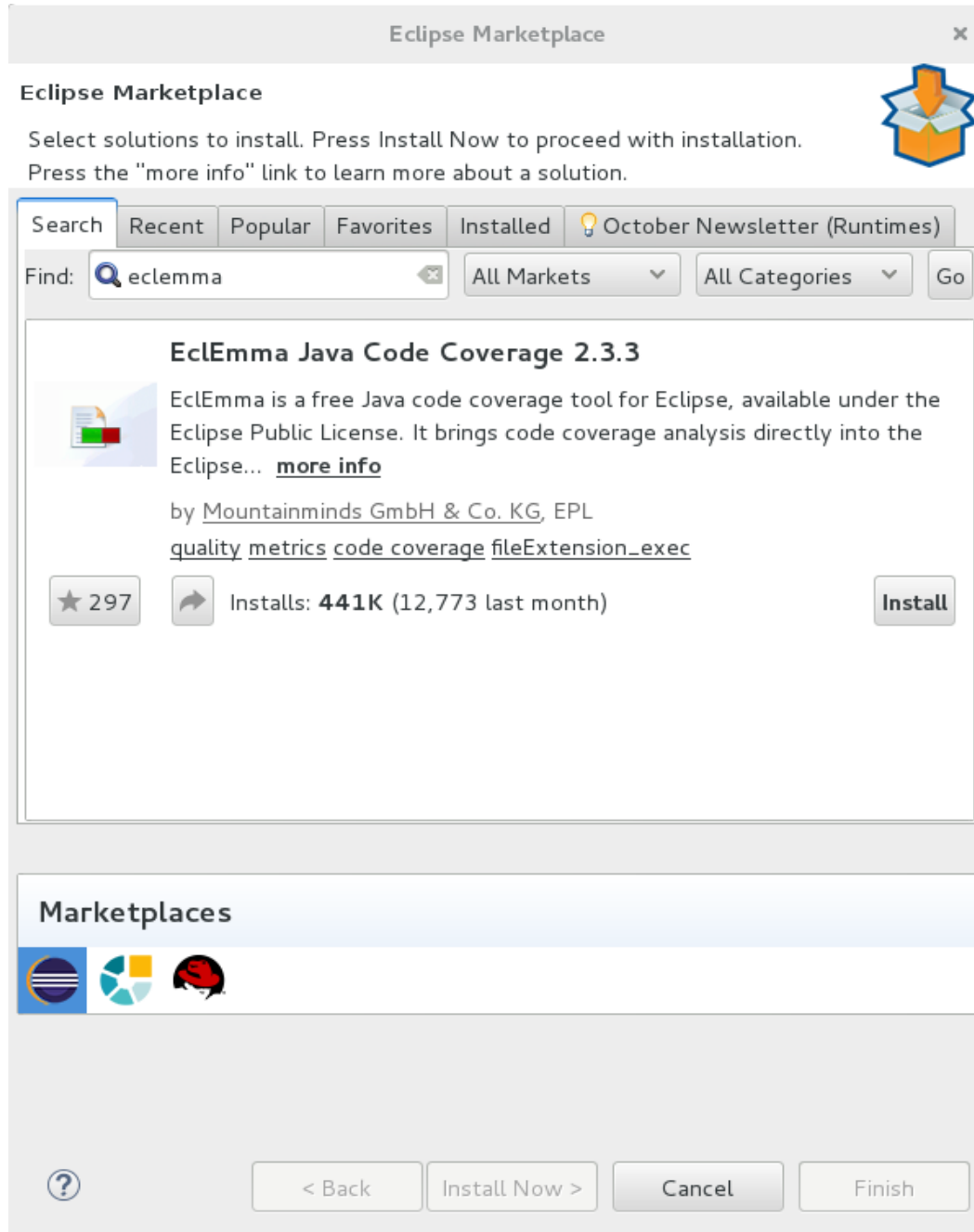
Slow Test => infinitest.filters



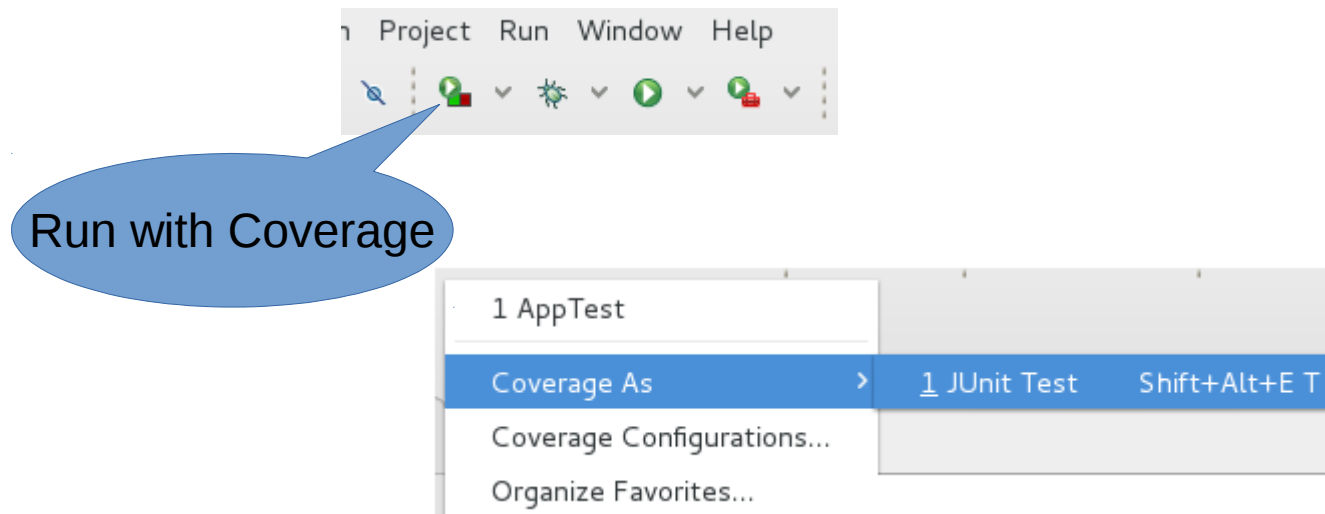
```
infinitest.filters ⌵  
  
# Tests that end in ITest:  
.*ITest  
  
# Inner Classes:  
.*\$.*  
  
# Tests in a certain package:  
com\\.mycompany\\.mypackage\\..*  
  
# All tests in this project:  
# .*
```

EclEma Eclipse Plugin

EclEmma Installation



Ecl Emma = Code Coverage



Shift+Alt+E T

Code Coverage %

The screenshot shows an IDE with two tabs: `AppTest.java` and `App.java`. The `App.java` tab is active, displaying the following Java code:

```
package fr.an.tests.testjunit;

import javax.script.ScriptEngine;

public class App {

    private static final ScriptEngine engine =
        new ScriptEngineManager().getEngineByName("nashorn");

    public static Object evalJS(String script) {
        try {
            Object res = engine.eval(script);
            return res;
        } catch (ScriptException ex) {
            throw new RuntimeException("Failed to eval script '" + script + '");
        }
    }
}
```

Below the code editor, a toolbar contains icons for `Proble`, `Missin`, `Conso`, `Javad`, `JUnit`, `Declar`, `Search`, `Progr`, `Error`, `Call Hi`, `Histor`, and `Cover`. The `Cover` icon is selected, displaying a table of code coverage data.

Element	Coverage	Covered Instruct	Missed Instructi
testjunit	22.6 %	28	96
src/test/java	16.1 %	15	78
src/main/java	41.9 %	13	18
fr.an.tests.testjunit	41.9 %	13	18
App.java	41.9 %	13	18

Testability Principles

How to Unit Test a System with Dependencies ?

```
package fr.an.tests.testjunit;

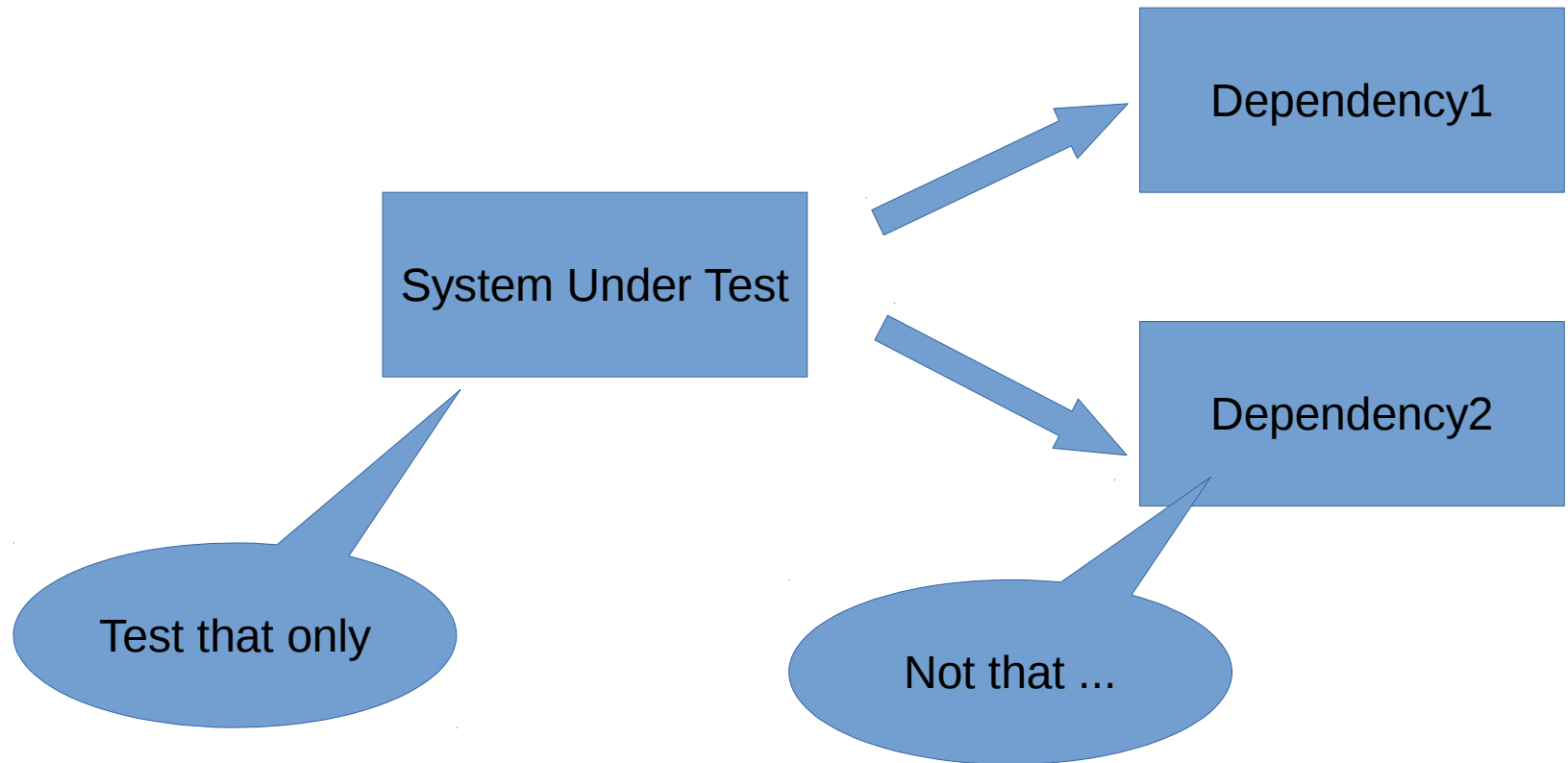
public class MyService {

    private MyDAO dao = new MyDAO();

    private MyDTOConverter converter = new MyDTOConverter();

    public MyDTO methodWithDependencies() {
        MyEntity e = dao.findMyEntity(123);
        if (e == null) {
            e = new MyEntity();
            dao.insertMyEntity(e);
        }
        return converter.entity2dto(e);
    }
}
```

“Unit” Test = Isolated Unitary Test



... Difficult to Test ...

use @Override if not final

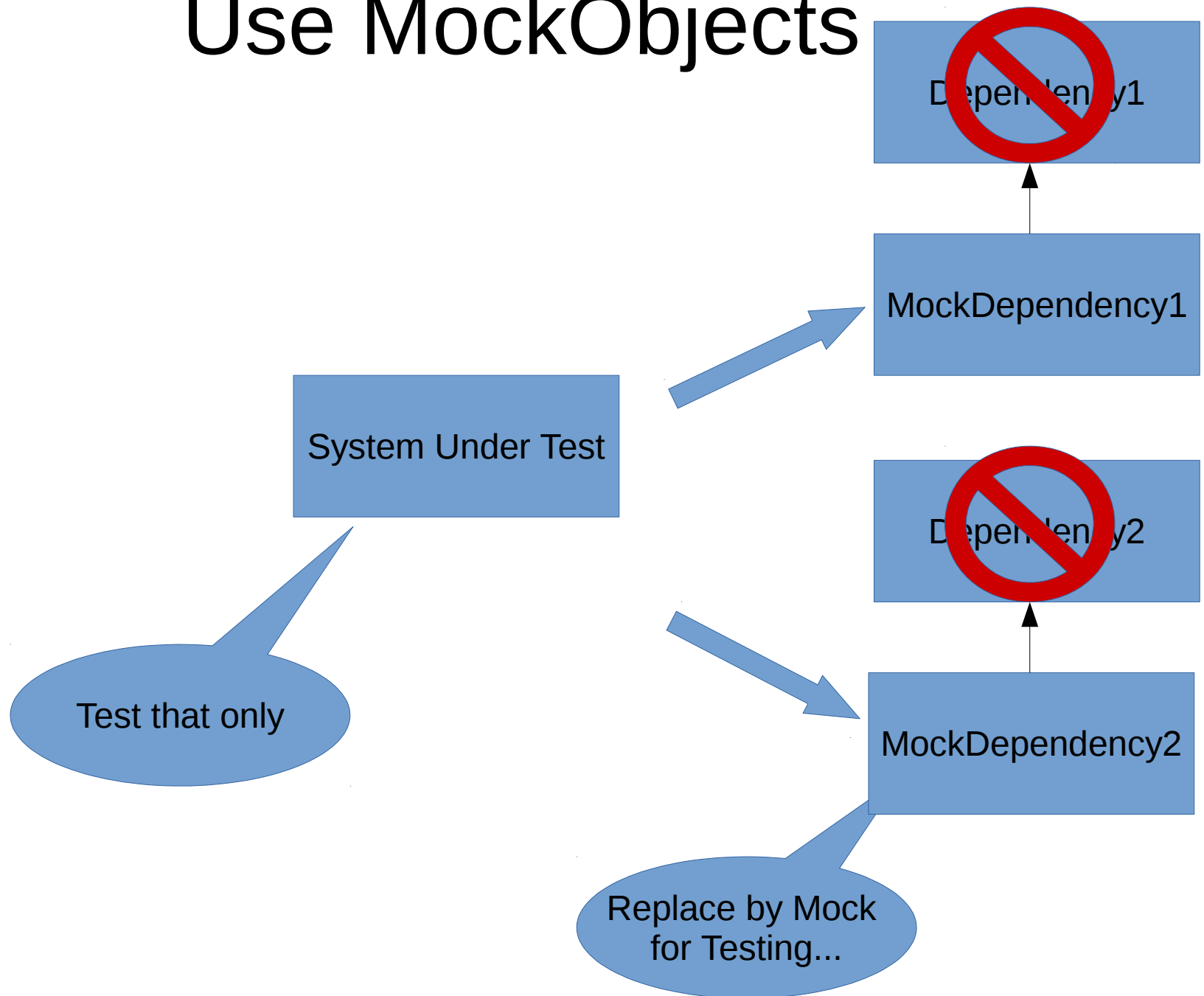
```
public class MyServiceTest {

    private MyService sut = new MyService() {
        /** override to replace real dependencies by mock methods */
        @Override
        public MyDTO methodWithDependencies() {
            MyEntity e = mock_dao_findMyEntity(123); // dao.findMyEntity(123);
            if (e == null) {
                e = new MyEntity();
                mock_dao_insertMyEntity(e); // dao.insertMyEntity(e);
            }
            return mock_converter_entity2dto(e); // converter.entity2dto(e);
        }
    };

    @Test
    public void testMethodWithDependencies() {
        MyDTO res = sut.methodWithDependencies();
        Assert.assertNotNull(res);
    }

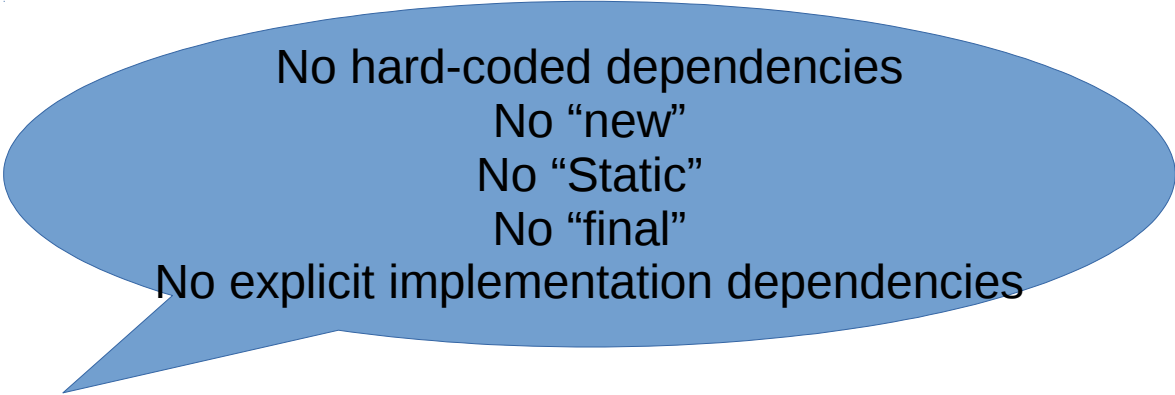
    protected MyEntity mock_dao_findMyEntity(int id) {
        return new MyEntity();
    }
}
```

Use MockObjects

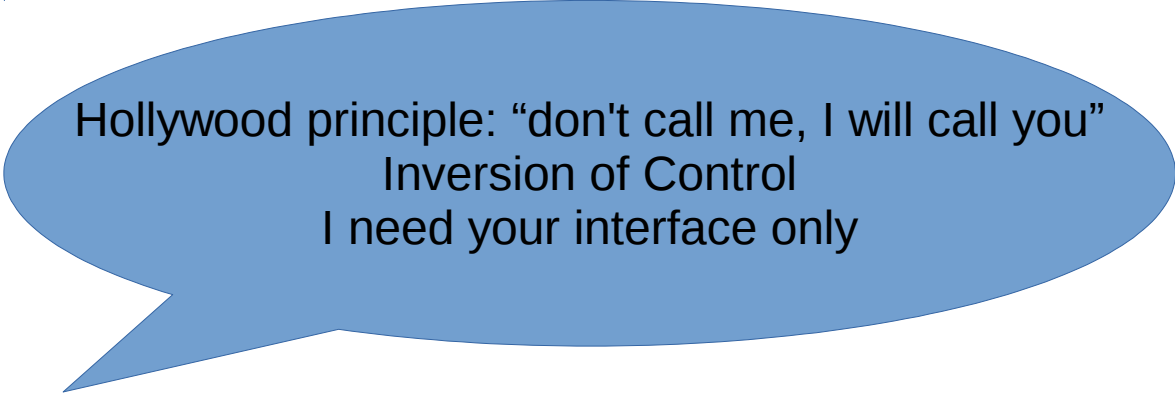


Testability Principles

IOC = “Inversion Of Control”
/ DI = Dependency Injection



No hard-coded dependencies
No “new”
No “Static”
No “final”
No explicit implementation dependencies



Hollywood principle: “don't call me, I will call you”
Inversion of Control
I need your interface only

Use Dependency Injection by Constructor

```
package fr.an.tests.testjunit;

public class MyService {

    protected MyDAO dao;

    protected MyDTOConverter converter;

    public MyService(MyDAO dao, MyDTOConverter converter) {
        this.dao = dao;
        this.converter = converter;
    }

    public MyDTO methodWithDependencies() {
        MyEntity e = dao.findMyEntity(123);
        if (e == null) {
            e = new MyEntity();
            dao.insertMyEntity(e);
        }
        return converter.entity2dto(e);
    }
}
```

Or Field Injection by Annotation

```
<dependency>  
  <groupId>javax.inject</groupId>  
  <artifactId>javax.inject</artifactId>  
  <version>1</version>  
</dependency>
```

```
import javax.inject.Inject;  
  
public class MyService {  
  
    @Inject  
    private MyDAO dao;  
  
    @Inject  
    private MyDTOConverter converter;  
  
    public MyDTO methodWithDependencies() {  
        MyEntity e = dao.findMyEntity(123);  
        if (e == null) {  
            e = new MyEntity();  
            dao.insertMyEntity(e);  
        }  
        return converter.entity2dto(e);  
    }  
}
```

Mockito Library

```
<dependency>  
  <groupId>org.mockito</groupId>  
  <artifactId>mockito-all</artifactId>  
  <version>1.10.19</version>  
  <scope>test</scope>  
</dependency>
```


@Mock + @InjectMocks

The SUT is injected its dependencies with @InjectMocks

```
@InjectMocks  
protected MyService sut = new MyService();
```

```
@Mock  
private MyDAO dao;
```

```
@Mock  
private MyDTOConverter converter;
```

Each Mocked dependency is created with @Mock

... equivalent to "Mockito.mock(MyXX.class)"

@RunWith(MockitoJUnitRunner.class)

Where the magic starts ... (from JUnit runner)

```
import org.junit.Assert;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.mockito.InjectMocks;
import org.mockito.Mock;
import org.mockito.Mockito;
import org.mockito.runners.MockitoJUnitRunner;

@RunWith(MockitoJUnitRunner.class)
public class MyServiceTest {
```

JUnit understand to start
a Mockito Test

Otherwise ..

Unless interpreted, annotation are just as javadoc

```
/**
 * a JUnit test for MyService
 *
 * @RunWith(MockitoJUnitRunner.class)
 * .. as a javadoc comment, this would have no effect..
 */
```

Mockito Test Example

```
import org.junit.runner.RunWith;
import org.mockito.InjectMocks;
import org.mockito.Mock;
import org.mockito.Mockito;
import org.mockito.runners.MockitoJUnitRunner;

@RunWith(MockitoJUnitRunner.class)
public class MyServiceTest {

    @InjectMocks
    protected MyService sut = new MyService();

    @Mock
    private MyDAO dao;

    @Mock
    private MyDTOConverter converter;

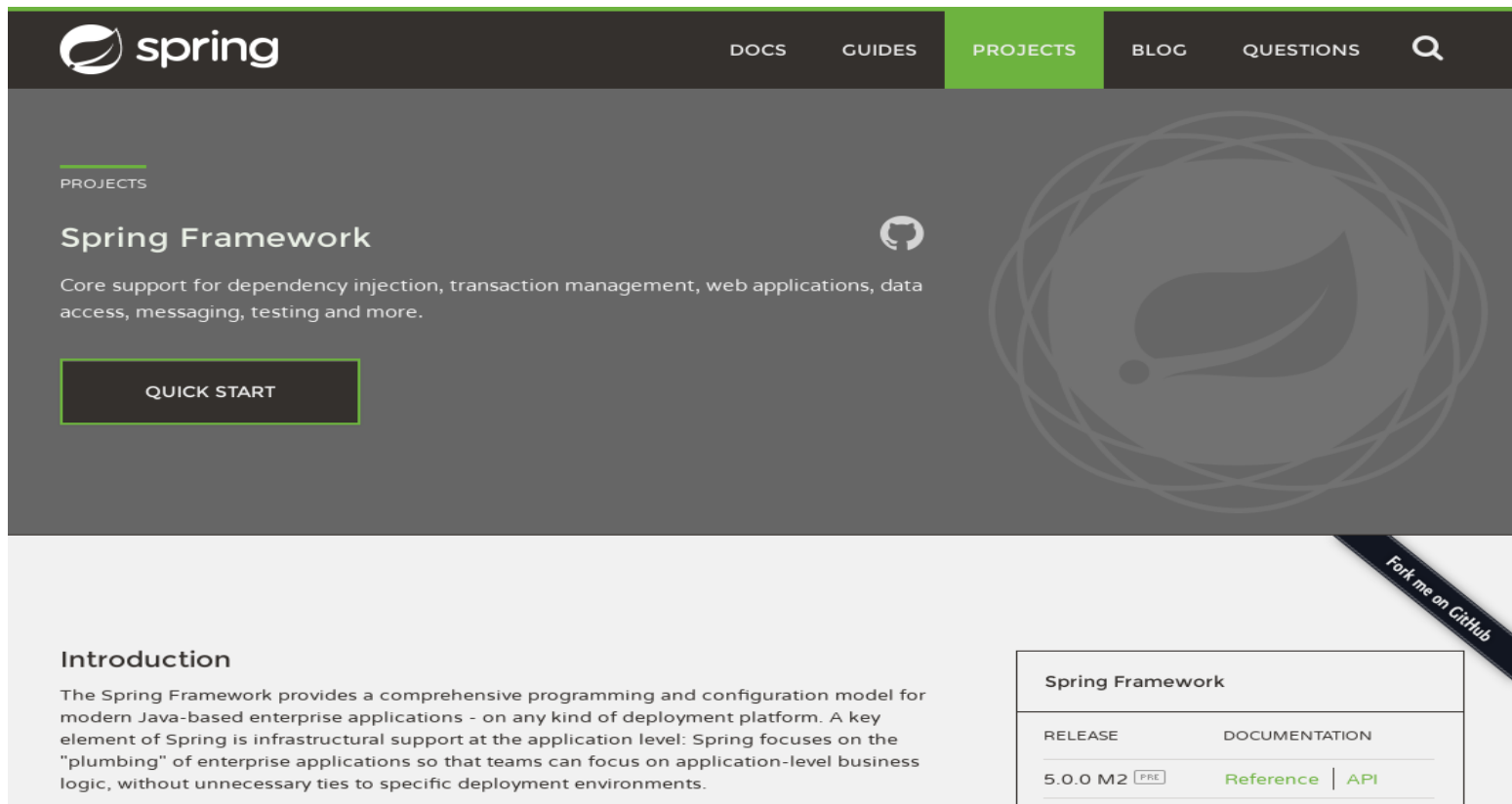
    @Test
    public void testMethodWithDependencies() {
        // Ante
        Mockito.when(converter.entity2dto(Mockito.any())).thenReturn(new MyDTO());
        // Action
        MyDTO res = sut.methodWithDependencies();
        // Assert
        Assert.assertNotNull(res);
    }
}
```

The screenshot shows the bottom portion of an IDE window, specifically the JUnit test runner. The top bar contains tabs for 'Problems', 'Missing Te', 'Console', 'Javadoc', 'JUnit 5', 'Declaratio', 'Search', 'Progress', 'Error Log', 'Call Hierar', and 'History'. The 'JUnit 5' tab is active. Below the tabs, the text 'hed after 0.124 seconds' is visible. At the bottom, a status bar displays 'ns: 1/1', 'Errors: 0', and 'Failures: 0'. A green progress bar is shown on the right side of the status bar.

Springframework Test Library

Spring = THE IOC Library for Java

- DE FACTO Standard for IOC in Java
- Others implementations:
 - Guice, Plexus, EJB (seriously?), ...



The screenshot shows the Spring Framework website. The top navigation bar includes links for DOCS, GUIDES, PROJECTS (highlighted), BLOG, and QUESTIONS, along with a search icon. The main content area features the Spring logo, the text 'Spring Framework', and a description: 'Core support for dependency injection, transaction management, web applications, data access, messaging, testing and more.' A 'QUICK START' button is visible. Below this, there is an 'Introduction' section and a table with release and documentation links.

PROJECTS

Spring Framework

Core support for dependency injection, transaction management, web applications, data access, messaging, testing and more.

[QUICK START](#)

Introduction

The Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications - on any kind of deployment platform. A key element of Spring is infrastructural support at the application level: Spring focuses on the "plumbing" of enterprise applications so that teams can focus on application-level business logic, without unnecessary ties to specific deployment environments.

Spring Framework	
RELEASE	DOCUMENTATION
5.0.0 M2 <small>PRE</small>	Reference API

Fork me on GitHub

Upgrade code to Springframework

1/ Add springboot-parent + spring-* pom.xml

```
<parent>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-parent</artifactId>
  <version>1.4.0.RELEASE</version>
  <relativePath/>
</parent>

<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
</dependencies>
```

2/ Add @Component to managed bean classes

```
import org.springframework.stereotype.Component;

@Component
public class MyDTOConverter {
```

3/ Add @Configuration (with @ComponentScan)

```
@Configuration
@ComponentScan
public class TestConfig {
```

@Inject SUT (with real Dependencies...)

```
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(classes = TestConfig.class)
public class MyServiceIT {

    @Inject
    protected MyService sut;

    @Test
    public void testMethodWithDependencies() {
        // Ante
    }
}
```

The SUT is injected its dependencies
with @Inject

There is NO Mock here !!
This is an Integration Test, not a Unit test

@RunWith(SpringJUnit4Runner.class)

Where the magic starts ... (from Junit runner)

```
import org.junit.runner.RunWith;
import org.springframework.test.context.ContextConfiguration;
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;

/**
 * an Integration Test for MyService (not a Unit test)
 */
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(classes = TestConfig.class) // => @Configuration +
public class MyServiceIT {
```

Junit understand to start
a Spring Test

Springframework understand
its configuration

Otherwise ..

Unless interpreted, annotation are just as javadoc

... like Mockito

```
/**
 * an Integration Test for MyService (not a Unit test)
 * @RunWith(SpringJUnit4ClassRunner.class)
 * @ContextConfiguration(classes = TestConfig.class)
 * .. as a javadoc this would have no effect
 */
```


Run Test with Spring

```
import org.junit.runner.RunWith;
import org.springframework.test.context.ContextConfiguration;
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;

/**
 * an Integration Test for MyService (not a Unit test)
 */
@RunWith(SpringJUnit4ClassRunner.class)
@ContextConfiguration(classes = TestConfig.class) // => @Configuration + @ComponentScan
public class MyServiceIT {

    @Inject
    protected MyService sut;

    @Test
    public void testMethodWithDependencies() {
        // Ante
        // Action
        MyDTO res = sut.methodWithDependencies();
        // Assert
        Assert.assertNotNull(res);
    }
}
```

The screenshot shows the bottom portion of an IDE window. At the top, a toolbar contains icons for 'Proble', 'Missing', 'Consol', 'Javado', 'JUnit' (active), 'Declar', 'Search', 'Progres', 'Error L', 'Call Hie', 'History', and 'Covera'. Below the toolbar, a status bar displays 'nished after 0.332 seconds'. A progress bar shows 'Runs: 1/1', 'Errors: 0', and 'Failures: 0'. At the bottom, a tab for 'fr.an.tests.testjunit.MyServiceIT [Runner: JUnit 4] (0.006 s)' is active, with a 'Failure Trace' button to its right.

Conclusion

Conclusion

- Junit ecosystem is amazing
 - Junit, Maven, Eclipse, Mockito, SpringFramework all works together
 - TDD - For better code
- Questions ?
 - Arnaud.nauwynck@gmail.com
- This document:
<http://arnaud.nauwynck.free.fr/>