Unit Testing and JUnit

Davide Falessi

What is JUnit?

- Open source Java testing framework used to write and run repeatable automated unit tests
- JUnit is open source (junit.org)

What is JUnit?

- JUnit features include:
  - Assertions for testing expected results
  - Test features for sharing common test data
  - Test suites for easily organizing and running tests
  - Graphical and textual test runners
- JUnit is widely used in industry
- JUnit can be used as stand alone Java programs (from the command line) or within an IDE such as Eclipse

JUnit test

- JUnit can be used to test …
  - … an entire object
  - … part of an object – a method or some interacting methods
  - … interaction between several objects
- It is primarily for unit and integration testing, not system testing

Writing Tests for JUnit

- Need to use the methods of the junit.framework.assert class
  - javadoc gives a complete description of its capabilities
- Each test method checks a condition (assertion) and reports to the test runner whether the test failed or succeeded

Assertion types

- *static void assertEquals (boolean expected, boolean actual)*
  Asserts that two booleans are equal
- *static void assertEquals (byte expected, byte actual)*
  Asserts that two bytes are equal
- *static void assertEquals (char expected, char actual)*
  Asserts that two chars are equal
- *static void assertEquals (double expected, double actual, double delta)*
  Asserts that two doubles are equal, within a delta
- *static void assertEquals (float expected, float actual, float delta)*
  Asserts that two floats are equal, within a delta
- *static void assertEquals (int expected, int actual)*
  Asserts that two ints are equal
- For a complete list, see
  - *http://junit.sourceforge.net/javadoc/org/junit/Assert.html*

Simple Example

```
package logic;

public class BasicCalculations {


    public int perfectSum (int a, int b){
        int result;
            result = a + b;
        return (result);
    }

    public double perfectDivision (int a, int b){
        double result;
            result = a / b;
        return (result);
    }
}
```

Simple Example

```
package test;

import logic.BasicCalculations;
import org.junit.Test;
import static org.junit.Assert.*;

public class TestDivision {

    @Test
    public void testDivision() {

        BasicCalculations b = new BasicCalculations();
        double output = b.perfectDivision(11, 10);
        assertEquals((double) 2, output, 1);

    }

}
```

1) Right Click, Build Path, Configure, Add Library, Junit
2) Right Click, Run as, Junit Test

Creating and Running a Test Suite

- Create a test division class.
- Go to test, ->New, Other, Java, Junit, JunitTestSuite, select and run as Junit Test.

Best practices

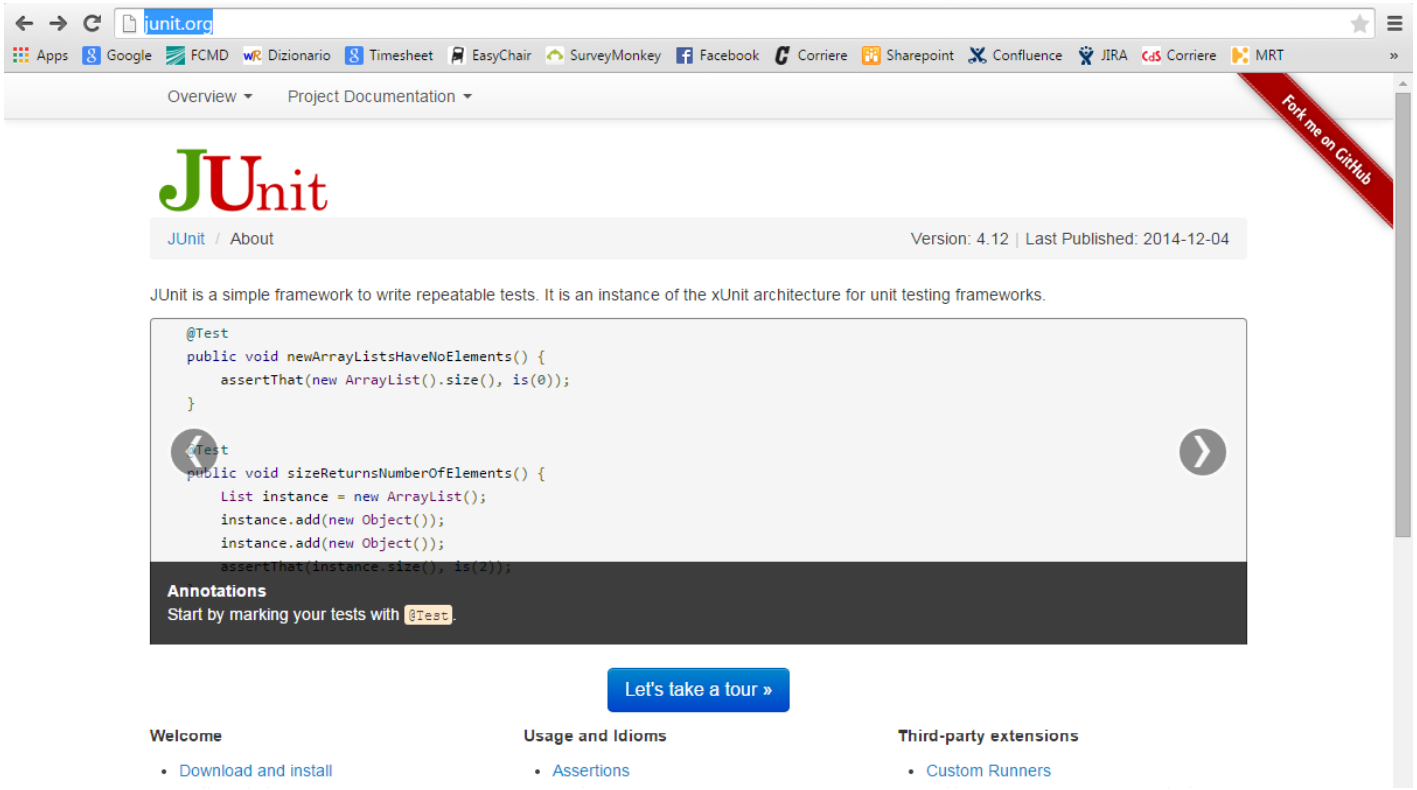
- Write tests in a separate package (e.g., test)
- Write at least one test class for each logic class.
  - Do not write one test class for more than one logic class (it does not apply to type of testing different than unit testing)
- Use multiple test methods for the same method under testing.
- Use one “assert” per each test method.
- Name the test class as TestX where X is the class you are testing (e.g., TestCalculations).
- Name the test method testYZ where Y is the method being tested and Z is a specific case. (e.g., testDivisionByZeros for testing the method Division in case the divisor is zero).

How to install Junit (if required)

Installing JUnit in Eclipse

Installation steps:

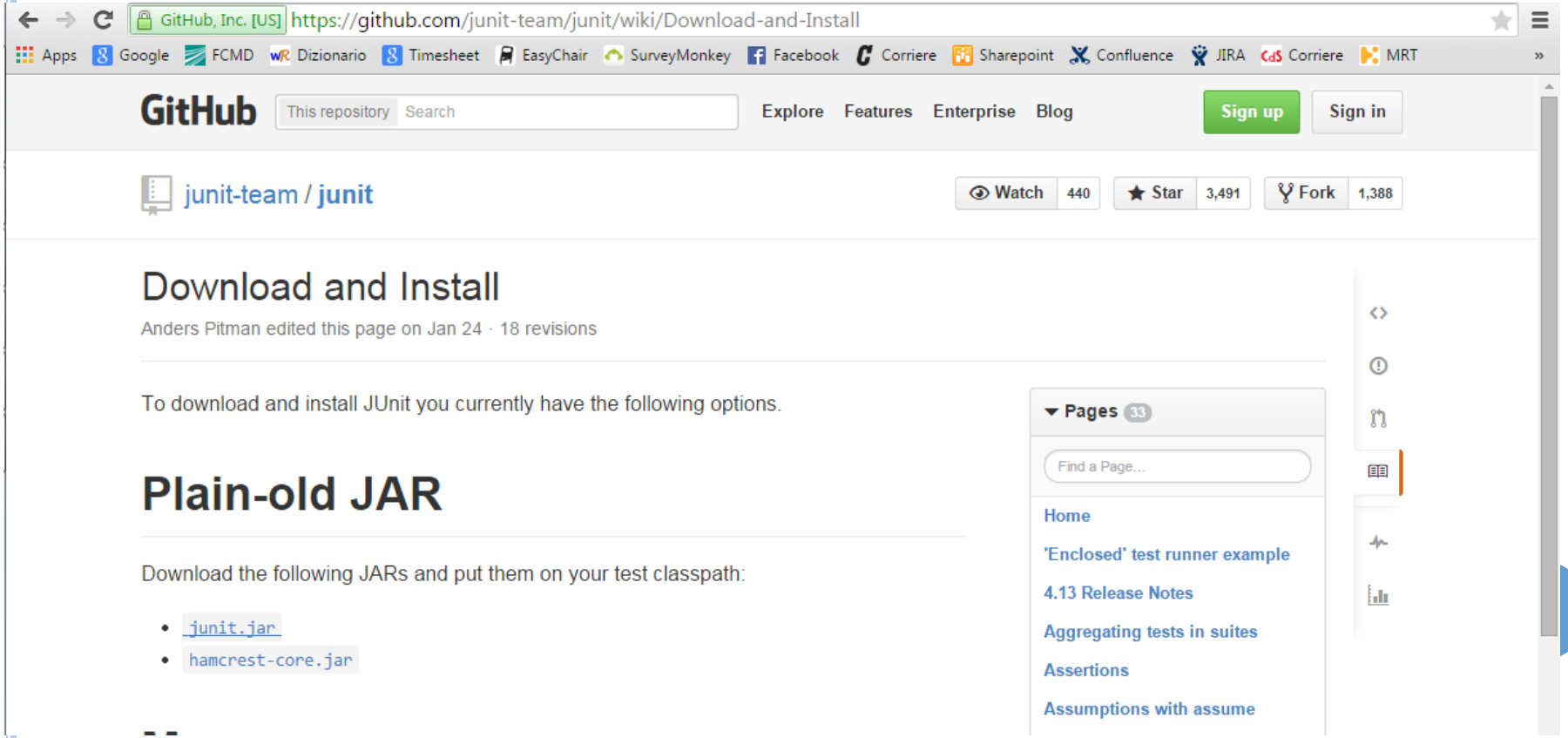
1) Download Junit form JUnit.org



Installing Junit in Eclipse

Installation steps:

2) Select Junit.jar



Installing JUnit in Eclipse

Installation steps:

3) Save the file “junit-4.12.jar” in “C:\JUnit”

4) Set the environment variable JUNIT_HOME to C:\JUnit

4) Open Eclipse

5) Right click on a (new) project, Properties, Java Build Path, Libraries, Add External Jar, select “junit-4.12.jar” from “C:\JUnit”