

Compsci230

Tutorials

Introduction of Testing

- Testing is the process of finding differences between the expected behavior specified by system models and the observed behavior of the implemented system. Unit testing finds differences between a specification of an object and its realization as a component
- Three testing methods:
 - Structural testing finds differences between the system design model and a subset of integrated subsystems.
 - Functional testing finds differences between the use case model and the system.
 - performance testing finds differences between nonfunctional requirements and actual system performance.

An Overview of Testing

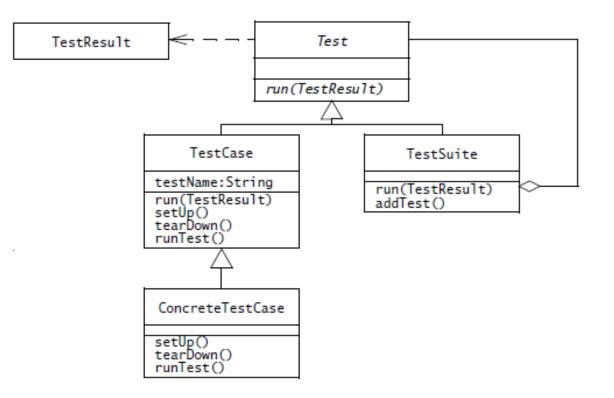
- Reliability is a measure of success with which the observed behavior of a system conforms to the specification of its behavior
- Software reliability is the probability that a software system will not cause system failure for a specified time under specified conditions
- Failure is any deviation of the observed behavior from the specified behavior
- erroneous state (also called an error) means the system is in a state such that further processing by the system will lead to a failure, which then causes the system to deviate from its intended behavior
- A fault, also called "defect" or "bug," is the mechanical or algorithmic cause of an erroneous state

An overview of testing activities

- **Test planning** allocates resources and schedules the testing. This activity should occur early in the development phase so that sufficient time and skill is dedicated to testing. For example, developers can design test cases as soon as the models they validate
- **Usability testing** tries to find faults in the user interface design of the system. Often, systems fail to accomplish their intended purpose simply because their users are confused by the user interface and unwillingly introduce erroneous data.
- **Unit testing** tries to find faults in participating objects and/or subsystems with respect to the use cases from the use case model
- Integration testing is the activity of finding faults by testing individual components in combination. Structural testing is the culmination of integration testing involving all components of the system. Integration tests and structural tests exploit knowledge from the SDD (System Design Document) using an integration strategy described in the Test Plan (TP).
- **System testing** tests all the components together, seen as a single system to identify faults with respect to the scenarios from the problem statement and the requirements and design goals identified in the analysis and system design, respectively:
 - **Functional testing** tests the requirements from the RAD and the user manual.
 - Performance testing checks the nonfunctional requirements and additional design goals from the SDD(System Design Document). Functional and performance testing are done by developers.
 - Acceptance testing and installation testing check the system against the project agreement and is done by the client, if necessary, with help by the developers.

JUnit Introduction

 JUnit is a framework for writing and automating the execution of unit tests for Java classes.



JUnit Introduction

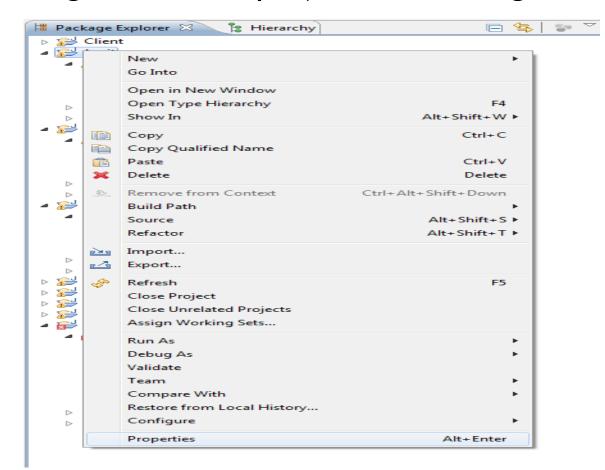
- ConcreteTestCase Class: the setUp() and tearDown() methods of the concrete test case initialize and clean up the testing environment. runTest() method includes the actual test code that exercises the class under test and compares the results with an expected condition.
- TestResult: report the testing results, either success or failure.

How to start JUnit

Using Eclipse create a new project (for example: Calcuator)

| 🖨 Ja | ava - JU | nitTest2/ | src/WordD | ealUtilTest.j | ava - Ecli | ipse | | | | | | - | | |
|--------|----------|-------------|--------------|---------------|------------|-------------|------|------------|----------------------------|--------|--------|-----------|--------|------|
| File | Edit | Source | Refactor | Navigate | Search | Project | Run | Wir | ndow Help | | | | | |
| | New | | | | | Alt+Shift | +N ► | 12 | Java Project | | | | | r. |
| | Open | File | | | | | | C2 | Project | | | | | 5 |
| | Close | | | | | Ctrl | +W | B | Package | | | | | F |
| | Close | All | | | | Ctrl+Shift- | +W | G | Class | | | | | E. |
| | Save | | | | | Ctr | I+S | œ | Interface | | | | | r |
| | Save A | \s | | | | | | G | Enum | | | | | 2 |
| R | Save A | AII | | | | Ctrl+Shift | t+S | @ | Annotation | | | | | |
| | Rever | t | | | | | | ₽ ₽ | Source Folder | | | | | a: |
| | Move | | | | | | | 5 | Java Working Set | | | | | |
| | Renar | | | | | | | | Folder | | | | | |
| ন্ত্রী | Refres | h | | | | | F5 | | File Untitled Text File | | | | | ¢. |
| | Conve | ert Line De | elimiters To | • | | | ► | | JUnit Test Case | | | | | Ę |
| ≞ | Print. | | | | | Ctr | L+ D | | Operational QVT | Trar | nsforn | nation | | ſ |
| - | | | | | | ea | | | Operational QVT | | | | | |
| | | n Workspa | ace | | | | • | Ċ | Task | | - | | | ľ |
| | Restar | t | | | | | | E\$ | Example | | | | | 1 |
| è | Impor | t | | | | | | | | | | | | - |
| 4 | Export | t | | | | | | | Other | 1 1000 | | | Ctrl+N | r F |
| | Prope | rties | | | | Alt+Er | nter | L . | | | | } | assei | rtEq |
| | 1 Wor | dDealUtil | Test.java [J | UnitTest2/s | rc] | | | 1 | | | | // I | est I | last |
| | | | .java [JUnit | | | | | 1 | | | Θ | @Tes | | |
| | 3 Calo | ulatorTes | t.java [Juni | it3/src] | | | | 1 | | | | publ { | ic vo | 010 |
| | 4 Calo | ulator.jav | a [Junit3/s | rc] | | | | L - | | | | - | Strin | ng t |
| | Exit | | | | | | | 1 | | | | | Strin | |
| | - AR | | | | | | | J., . | | | | | assei | съq |

Right Click the project name and go to the properties



Go to the build path -> Add library -> Junit-> JUnit4

| Properties for Junit | | |
|--|--|---|
| type filter text | Java Build Path | ↔ ↔ ↔ ★ |
| Resource Builders Java Build Path Java Code Style Java Compiler Java Editor Javadoc Location Project References Run/Debug Settings Task Repository Task Tags Validation WikiText | Image: Source Projects Image: Source | Add JARs Add External JARs Add Variable Add Library Add Class Folder Add External Class Folder Edit Remove Migrate JAR File |
| ? | | OK Cancel |

Calculator-> New->Junit Test Case

| Package Explorer S Client Junit Src (default packag Calculator,ia | | | D w | <pre>ordDealUtiljava</pre> |
|--|---|---|---|--|
| ▷ ➡ JRE System Librar ▷ ➡ JUnit 3 ▲ JUnitTest2 | New Open Open With Open Type Hierarchy Show In Copy Copy Qualified Name Paste Delete Delete | F3 F4 Alt+Shift+W Ctrl+C Ctrl+V Delete | €1 11 11 11 11 11 11 11 11 11 11 11 11 1 | Java Project Project Package Class Interface Enum Annotation Source Folder Java Working Set |
| ▲ ∰ (default packa ▷ Ŵ OrdDeal ▷ Ŵ OrdDeal ▷ MordDeal ▷ MordDeal ▷ ▲ JRE System Librar ▷ ▲ JUnit 4 ▷ ₩ Q1 | Remove from Context Build Path Source Refactor Import | Ctrl+Alt+Shift+Down Alt+Shift+S Alt+Shift+T | Image: Second system Folder Image: Second system File Image: Second system Untitled Text File Image: Second system JUnit Test Case Image: Second system Operational QVT Transformation Image: Second system Operational QVT Library | File Untitled Text File JUnit Test Case Operational QVT Transformation Operational QVT Library |
| ▷ 🔂 Thread ▲ 🔂 Tur8 ▲ ∰ src | Export References Declarations Refresh Assign Working Sets | F5 | | Task Example Other Ctrl+N assercequars("emproyee_in" } // Test more than one charact |
| ▷ ▲ ClockCom ▷ ▲ ClockTest ▷ ▲ StillClock. ▷ ➡ JRE System Librar ▷ ➡ JUnit 4 | Run As Debug As Validate Team Compare With Replace With Restore from Local History |) | | <pre>@Test public void wordEormat4DBToge { String target = "employee String result = vordDeall assertEquals("employee_a_ }</pre> |
| | Properties | Alt+Enter | | ٠ |

| New JUnit Test | Case | |
|-----------------------------------|--|------------------|
| JUnit Test Ca 🔇 Type already e | | Ξ |
| 🔘 New JUnit 3 te | est 💿 New JUnit 4 test | |
| Source folder: | Junit/src | Browse |
| Package: | (def | fault) Browse |
| Name: | CalculatorTest | |
| Superclass: | java.lang.Object | Browse |
| | subs would you like to create? setUpBeforeClass() tearDownAfterClass() setUp() tearDown() constructor dd comments? (Configure templates and default valu Generate comments | ue <u>here</u>) |
| Class under test: | Calculator | Browse |
| ? | < Back Next > Finish | Cancel |

| Calculator | |
|---|------------|
| add(int) substract(int) multiply(int) divide(int) square(int) squareRoot(int) clear() getResult() Object ^c Object() ^f getClass() hashCode() equals(Object) clone() toString() | Select All |

Example of Calculator

- import static org.junit.Assert.*;
- import org.junit.Before;
- import org.junit.Test;
- import org.junit.Before;
- import org.junit.lgnore;
- import org.junit.Test;
- public class CalculatorTest {
- private static Calculator calculator = new Calculator();
- @Before
- public void setUp() throws Exception {
 - calculator.clear();
- }

}

D

@Test
public void testAdd() {

}

- calculator.add(2);
- calculator.add(3);
- assertEquals(5, calculator.getResult());