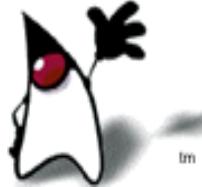


# Agenda



- Software Testing
- Testing JEE applications
  - Unit Testing
  - Integration Testing
  - User Interface Testing

# Software Testing



- Functional Testing
  - Unit testing
  - Integration testing
  - System Testing
  - User Acceptance testing
- Non-Functional Testing
  - Security Testing
  - Performance Testing
  - Compatibility Testing
  - Usability testing
  - Compliance Testing



# Unit Testing

- Test small pieces in isolation
- Method, class, or function level
- Unit tests can be performed manually
- Unit tests often lend themselves to automation
- JUnit is an **automated** unit testing framework for Java



# JUnit features

- JUnit Classes
- Fixtures
- Test Suites
- Test Runners



# JUnit classes

- A JUnit class is dedicated to a set of tests
- The framework provides methods that will run automatically:
  - `setUp`
  - `tearDown`
  - `@beforeClass`
  - `@afterClass`
  - `@before`
  - `@after`

# JUnit Fixtures



- A fixture is a fixed state of a set of objects forming an environment for repeatable tests
- `setUp()` method, which runs before every test invocation. `@Before`
- `tearDown()` method, which runs after every test method. `@After`



# JUnit tests

- Use the `@Test` annotation to indicate which methods should run as tests

```
@Test  
public void testIt(){  
    private String message = "hello";  
    //do something  
    assertEquals(message, "tester");  
}
```



# Assert methods

- Tests pass or fail based on assertions:

1	<b>void assertEquals(boolean expected, boolean actual)</b> Checks that two primitives/objects are equal.
2	<b>void assertFalse(boolean condition)</b> Checks that a condition is false.
3	<b>void assertNotNull(Object object)</b> Checks that an object isn't null.
4	<b>void assertNull(Object object)</b> Checks that an object is null.
5	<b>void assertTrue(boolean condition)</b> Checks that a condition is true.
6	<b>void fail()</b> Fails a test with no message.



# Test Runner

```
import org.junit.runner.JUnitCore;  
import org.junit.runner.Result;  
import org.junit.runner.notification.Failure;  
  
public class TestRunner {  
    public static void main(String[] args) {  
        Result result = JUnitCore.runClasses(TestJUnit.class);  
        for (Failure failure : result.getFailures())  
            System.out.println(failure.toString());  
    }  
    System.out.println(result.wasSuccessful());  
}  
}
```

# Test Suites



- Runs a group of test classes together:

```
import org.junit.runner.RunWith;  
import org.junit.runners.Suite;  
//JUnit Suite Test  
@RunWith(Suite.class)  
@Suite.SuiteClasses({  
    TestJunit1.class, TestJunit2.class  
})  
public class JunitTestSuite { }
```

# Mocking Frameworks



- Mocking is the process of emulating components and services for the purposes of (unit) testing
- Example of mocking framework: Mockito
- Maven dependency on `mockito-core`



# Mockito example

```
import static org.mockito.Mockito.*;  
// mock creation  
List mockedList = mock(List.class);  
// using mock object - it does not throw any  
"unexpected interaction" exception  
mockedList.add("one");  
mockedList.clear();  
// selective, explicit, highly readable verification  
verify(mockedList).add("one");  
verify(mockedList).clear();
```

# Mockito example



```
// you can mock concrete classes, not only interfaces  
LinkedList mockedList = mock(LinkedList.class);  
  
// stubbing appears before the actual execution  
when(mockedList.get(0)).thenReturn("first");  
  
// the following prints "first"  
System.out.println(mockedList.get(0));  
  
// the following prints "null" because get(999) was not  
// stubbed  
System.out.println(mockedList.get(999));
```

# Mockito documentation



<http://static.javadoc.io/org.mockito/mockito-core/2.23.4/org/mockito/Mockito.html>

# UI Testing frameworks



- Selenium
- Screenster
- TestCraft
- Endtest
- Browsersync
- Protractor
- CasperJS
- Ghost Inspector
- Cypress.io

# Selenium



- Selenium IDE is a Chrome or Firefox addon
- Can create web-based application tests
- Will record your interactions to facilitate testing
- Add verifications and asserts to tests
- Demo inclass
- See also  
[https://www.seleniumhq.org/docs/02\\_selenium\\_ide.jsp](https://www.seleniumhq.org/docs/02_selenium_ide.jsp)