


JUnit Jumpstart



Struts University Series



JUnit Jumpstart

- Why do we test applications?
 - How do we test applications?
 - Why do we need a testing framework?
 - How do we get started with JUnit?
- 

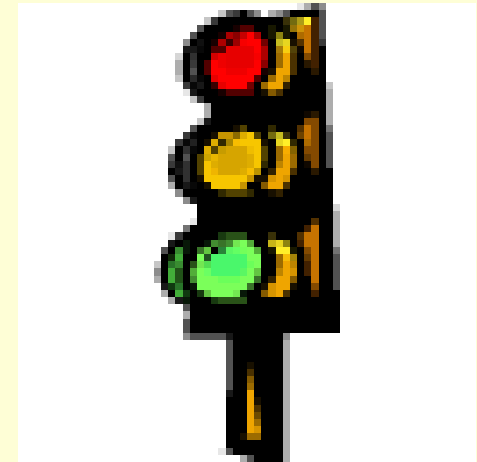
JUnit Jumpstart

- Writing simple tests by hand
- Writing better tests with JUnit
- Installing JUnit and running tests



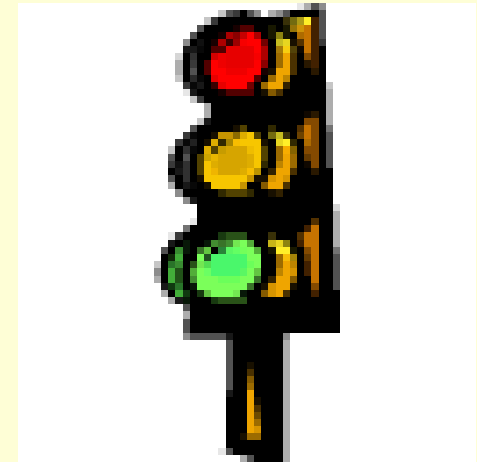
Proving it works

- “All code is tested.”
- Play tests
- Step-thru tests
- Test programs



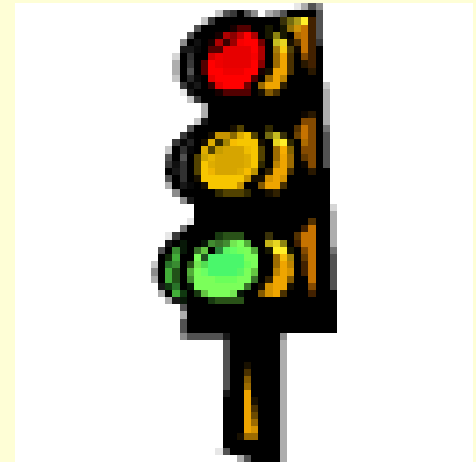
Proving it works

- “All code is tested.”
- Play tests
- Step-thru tests
- Test programs



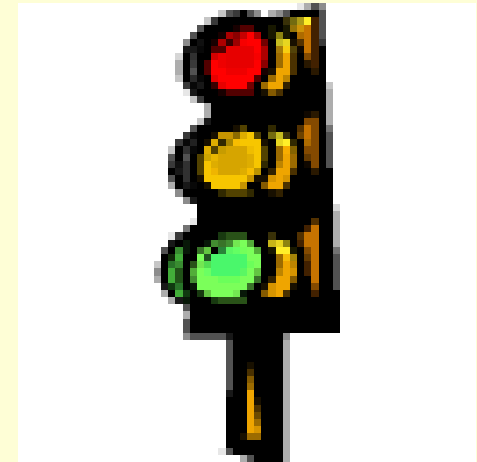
Proving it works

- “All code is tested.”
- Play tests
- Step-thru tests
- Test programs



Proving it works


- “All code is tested.”
- Play tests
- Step-thru tests
- Test programs






Starting from scratch

```
public class Calculator {  
    public double add(  
        double number1, double number2) {  
        return number1 + number2;  
    }  
}
```






Calculator Responsibilities

- add method – Take two doubles and return the sum as a double.
 - subtract method – TODO
 - multiply method – TODO
 - divide method – TODO
 - ...
- 




“Too simple to break?”

- JUnit practice: “Test anything that’s not too simple to break.”
 - Is this implementation of add is too simple to break?
 - Yes, but future implementations may not be
- 



“Too simple to break?”

- JUnit practice: “Test anything that’s not too simple to break.”
 - Is this implementation of add is too simple to break?
 - Yes, but future implementations may not be
 - Beck: “Any program feature without an automated test doesn’t exist.”
- 

Why write test programs?

Objective
test

+

Replicable
test


=

Simple test
program



Code under test


```
public class Calculator {  
    public double add(  
        double number1, double number2) {  
        return number1 + number2;  
    }  
}
```

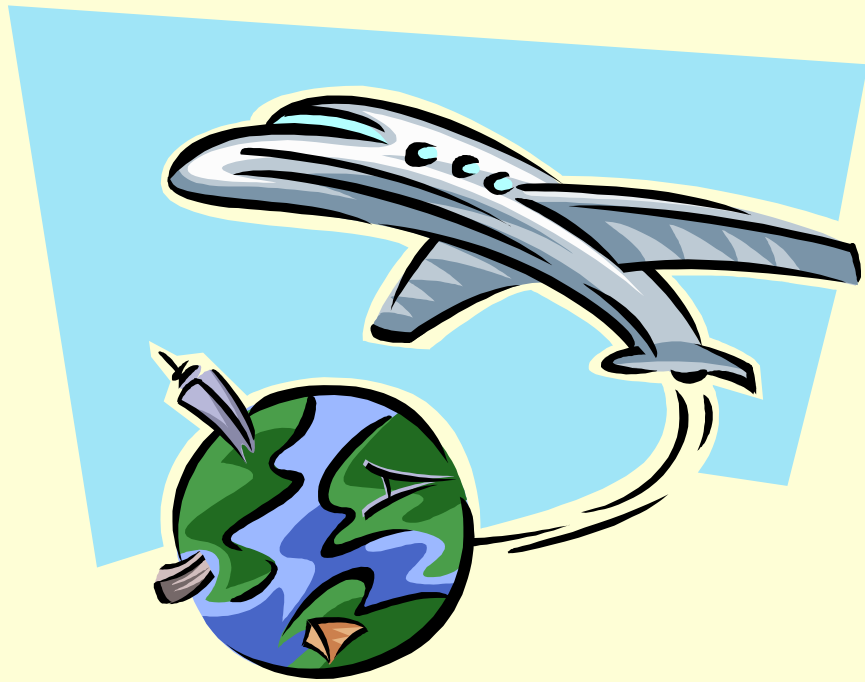




Proving add works

```
public class TestCalculator {  
    public static void main(String[] args) {  
        Calculator calculator = new Calculator();  
        double result = calculator.add(10,50);  
        if (result != 60) {  
            System.out.println("Bad result: " +  
result);  
        }  
    }  
}
```

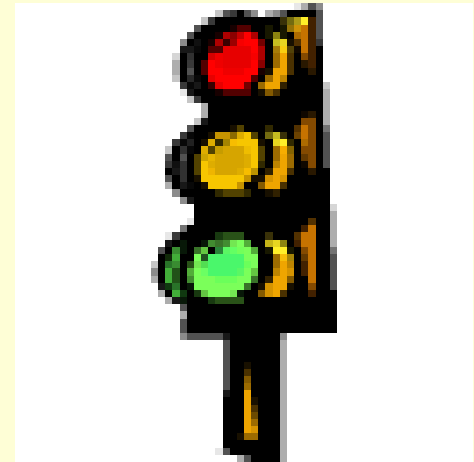




Review

“All code is tested.”

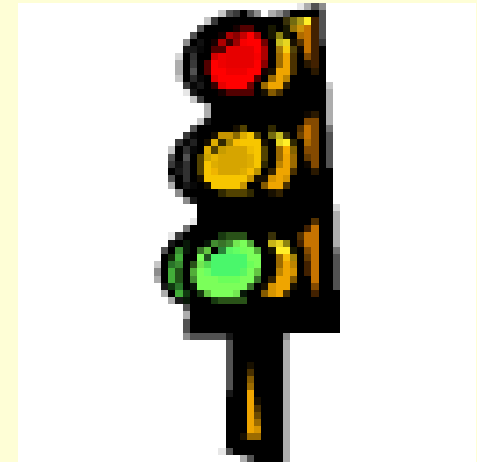
- **** tests
- ****_**** tests
- Test ****



Review


“All code is tested.”

- Play tests
- Step-thru tests
- Test programs






Review

- JUnit practice: “Test anything that’s not too simple to break.”
 - Beck: “Any program feature without an automated test doesn’t exist.”
- 




Review

- JUnit practice: “Test anything that’s not too ***** to break.”
 - Beck: “Any program feature without an ***** test doesn’t exist.”
- 




Review

- Unit tests demonstrate that the code works the way ********* it to work.
 - A unit test is (simple or complicated).
 - A unit test proves (one or several) expectation(s) at a time.
 - If code is a theory, then a unit test is a *********.
- 



Review

- Unit tests demonstrate that the code works the we expect it to work.
 - A unit test is (simple or complicated).
 - A unit test proves (one or several) expectation(s) at a time.
 - If code is a theory, then a unit test is a hypothesis.
- 


JUnit Jumpstart

- Writing simple tests by hand
- **Writing better tests with JUnit**
- Installing JUnit and running tests






A better infrastructure

- Test is fine, as far as it goes
 - Infrastructure poor
 - Need to add more tests
 - Need automatic monitoring
 - Testing program, not ourselves
- 



A better infrastructure


```
public class TestCalculator {  
    public static void main(String[] args) {  
        Calculator calculator = new Calculator();  
        double result = calculator.add(10, 50);  
        if (result != 60) {  
            System.out.println("Bad result: " +  
result);  
        }  
    }  
}
```





A better infrastructure


```
public class TestCalculator {  
  
    public void testAdd() {  
        Calculator calculator = new Calculator();  
        double result = calculator.add(10, 50);  
        if (result != 60) {  
            throw new RuntimeException("Bad result: " +  
                result);  
        }  
    }  
  
    // ... more tests here ...  
}
```





A better infrastructure

```
public static void main(String[] args) {
    TestCalculator test = new TestCalculator();
    try {
        test.testAdd();
        // ... More tests here ...
    } catch (Throwable e) {
        errors++;
        e.printStackTrace();
    }
    if (test.errors > 0) {
        throw new RuntimeException("There were " +
            test.errors
            + " error(s)");
    }
}
```






Testing with JUnit

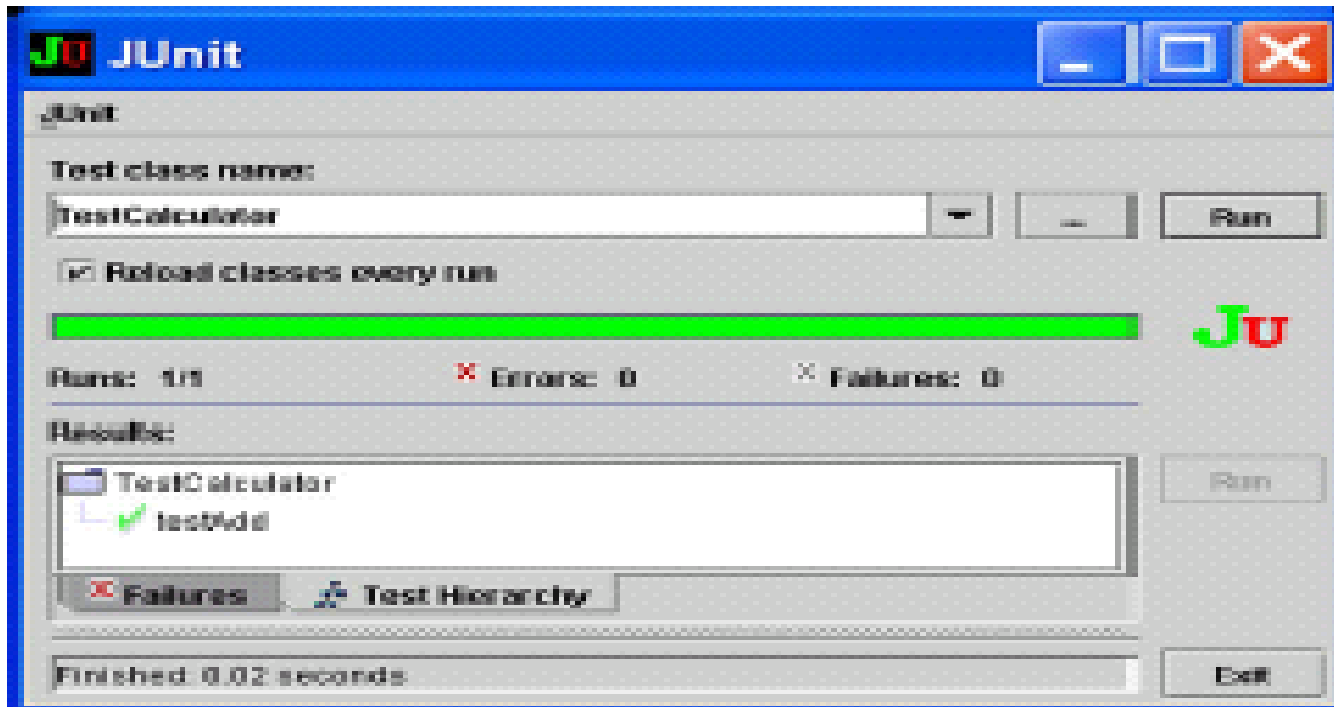
```
import junit.framework.TestCase;
public class TestCalculator extends TestCase {

    public void testAdd() {
        Calculator calculator = new Calculator();
        double result = calculator.add(10, 50);
        assertEquals(60, result, 0);
    }

    // ... more test* methods here ...

}
```








Be Assertive

```
assertNull("Expected user to be created", user);
```

```
assertNotNull(user);
```

```
assertSame("Expected users to match", user1, user2);
```

```
assertEquals("user", user.getUsername());
```





Core Assertions

 `assertTrue`

 `assertNull`

 `assertSame`

 `assertEquals`

 `assertFalse`

 `assertNotNull`

 `assertNotSame`

 `fail`





Core Assertions

 `assertTrue`

 `assertNull`

 `assertSame`

 `assertEquals`

 `assertFalse`

 `assertNotNull`

 `assertNotSame`


 `fail`

 `assertEquals` has 20 forms to cover various types!






JUnit Goals

- Each unit test must run independently of all other unit tests
 - Errors must be detected and reported test-by-test
 - It must be easy to define which unit tests will run
- 



JUnit Goals

The framework must help us

- write useful tests
 - create tests that retain value
 - lower costs by reusing code
- 


JUnit Today

- Junit – Gama and Beck, 1997
- Framework – semi complete application
- xUnit frameworks: C#, Perl, PHP, Visual Basic, et al
- Website: www.junit.org





Programmer's Unit Test

- A programmer's **unit test** examines the behavior of a distinct “unit of work”. In a Java application, methods often represent units of work.
 - By contrast, **integration** and **acceptance** tests examine component interactions.
- 



IEEE Unit Test

“Testing of individual hardware or software units *or groups of related units.*”


(emphasis added)





Unit of Work

A task that is not dependant on the completion of any other task.

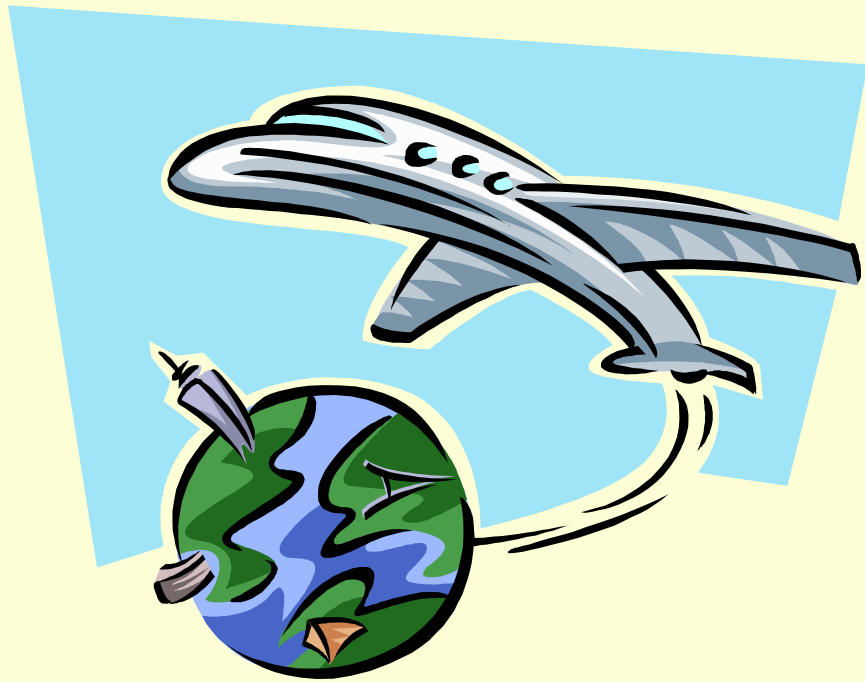




API Contact


A view of an Application Programming Interface (API) as a formal agreement between the caller and the callee.








Review

- A programmer's ****** **** examines the behavior of a distinct “unit of work”.
 - In a Java application, ********* often represent units of work.
 - By contrast, ********* and ********* tests examine component interactions.
- 




Review

- A programmer's unit test examines the behavior of a distinct “unit of work”.
 - In a Java application, methods often represent units of work.
 - By contrast, integration and acceptance tests examine component interactions.
- 



Review


The JUnit framework must help us

- write `*****` tests
 - create tests that retain `*****`
 - lower costs by `*****` code
- 



Review

The JUnit framework must help us


- write useful tests
 - create tests that retain value
 - lower costs by reusing code
- 



Review

A ******** of ******** is a task that is not dependent on the completion of any other task.

An ***** ******* is a view of an Application Programming Interface as a formal agreement between the caller and the callee.






Review

A unit of work is a task that is not dependent on the completion of any other task.

An API Contract is a view of an Application Programming Interface as a formal agreement between the caller and the callee.



JUnit Jumpstart

- Writing simple tests by hand
- Writing better tests with JUnit
- **Installing JUnit and running tests**





Setting up JUnit

- Download from junit.org
- Unzip distribution
 - `c:\junit`
 - `/opt/JUnit`
 - `/opt/Maven/junit/releases`






Testing JUnit

Windows:

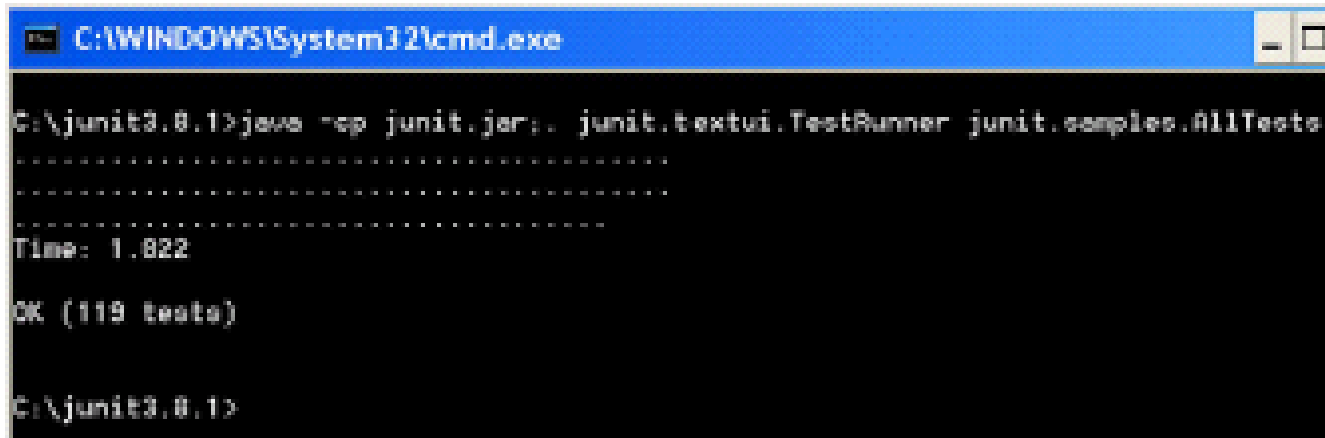
- `java -cp junit.jar;. junit.textui.TestRunner
junit.samples.AllTests`

UNIX:

- `java -cp junit.jar:. junit.textui.TestRunner
junit.samples.AllTests`



Testing JUnit



```
C:\WINDOWS\system32\cmd.exe

C:\junit3.8.1>java -cp junit.jar;. junit.textui.TestRunner junit.samples.AllTests
.....
.....
.....
Time: 1.822

OK (119 tests)

C:\junit3.8.1>
```



Testing JUnit


To run the text test runner, open a shell in `C:\junit\junit3.8.1` on Windows or in `/opt/junit/junit3.8.1` on UNIX, and type the appropriate command:

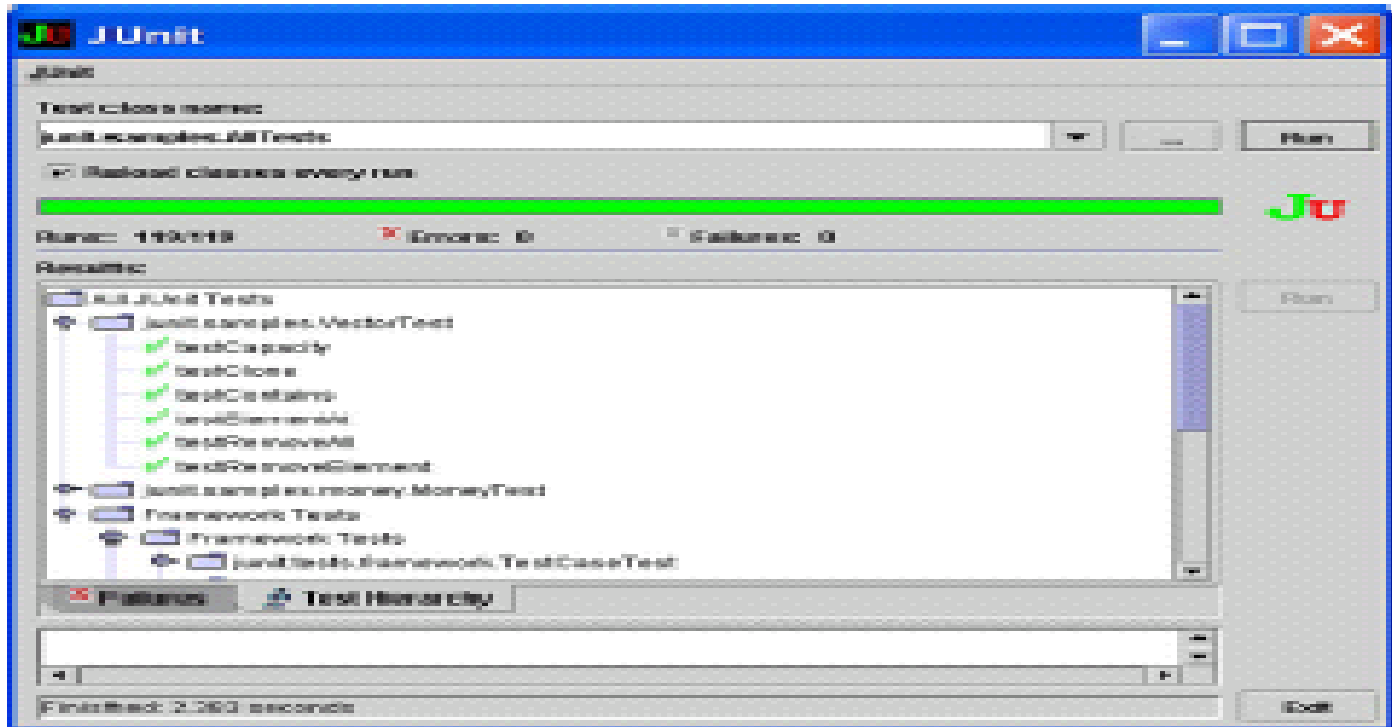
Windows:

```
java -cp junit.jar;. junit.swingui.TestRunner  
junit.samples.AllTests
```

UNIX:


```
java -cp junit.jar:. junit.swingui.TestRunner  
junit.samples.AllTests
```








Summary

- We all test the code we write
 - Simple test programs are simple
 - JUnit makes testing even simpler
 - JUnit preserves our investment in testing
- 



Summary

- We all test the code we write
 - Simple test programs are simple
 - JUnit makes testing even simpler
 - JUnit preserves our investment in testing
 - “Test until fear turns to boredom”
- 

JUnit in Action

JUnit Distilled


- JUnit Jumpstart
- Exploring JUnit
- Sampling JUnit
- Examining software tests
- Automating JUnit





JUnit in Action


Testing Strategies

- Coarse-grained testing with stubs
 - Testing in isolation with mock objects
 - In-container testing with Cactus
- 



JUnit in Action

Testing Components

- Unit-testing servlets and filters
 - Unit-testing JSPs and taglibs
 - Unit-testing database applications
 - Unit-testing EJBs
- 



Struts University Series