LDSTECH

THE CHURCH OF JESUS CHRIST OF LATTER-DAY SAINTS

Unit Testing with TestNG

Jeremy Lund

BUILD UPON THE TRADITION OF SERVICE!

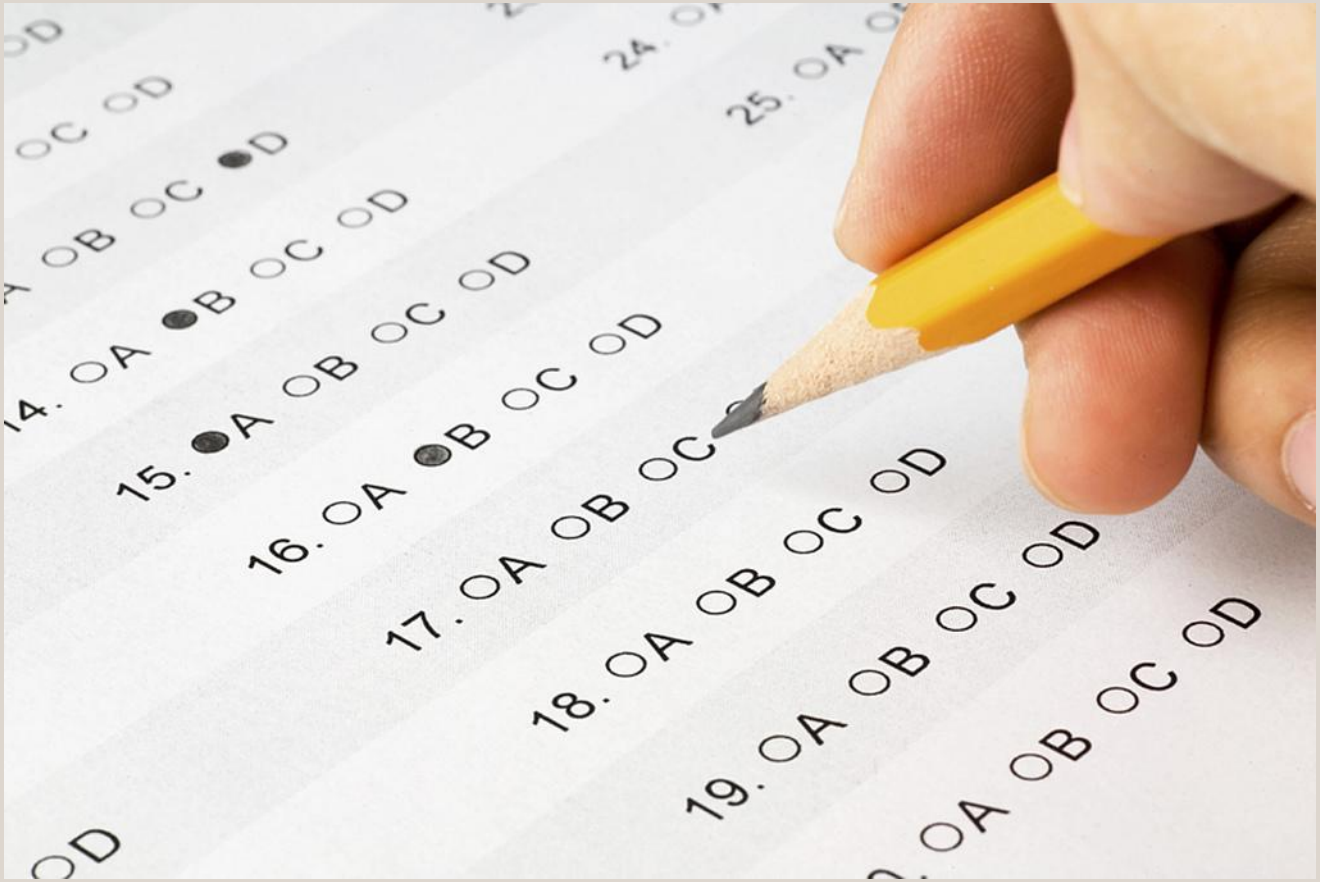
Notes

- This is a training, **NOT** a presentation
- Please ask questions
- This is being recorded
- https://tech.lds.org/wiki/Java_Stack_Training
- Prerequisites
  - Basic Java
  - Installed LDSTech IDE (or other equivalent)

Objectives

- By the end of the training, you should be able to answer the following questions:
  - What is a unit test really, and why should I care about tests?
  - What should I test?
  - How can I use a unit test to maintain bug-free code?
  - How can I use a unit test to fix bugs?
  - How do I write clean, maintainable, flexible tests?

What is a Test?



What is a Test?

"a set of questions, problems, or the like, used as a means of evaluating the abilities, aptitudes, skills, or performance of an individual or group; examination."

What is an Unit?

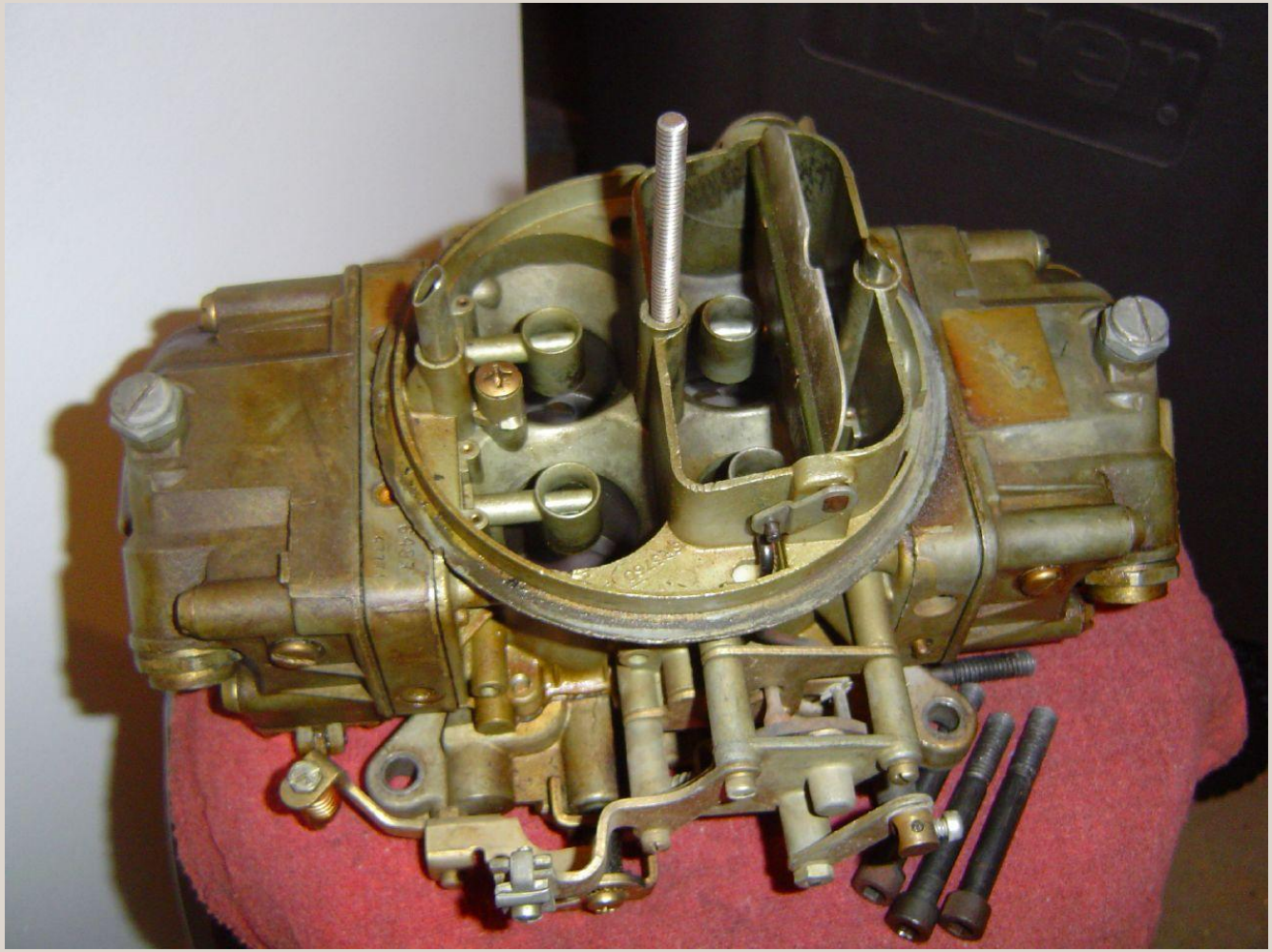
What is an Unit?

- An application
- A set of collaborative objects
- Package
- Class
- Method
- Execution path

What is an Unit?

- ~~An application~~
- ~~A set of collaborative objects~~
- ~~Package~~
- Class *(very rarely)*
- Method
- Execution path

Unit Test



Integration Test

Functional Test

Why Should I Care?

- Technically, it's your job
- Tests help new developers
- Tests help you
- Regression testing
- Fix the problems—not the symptoms

Anatomy of a Unit Test

org.lds.stack.training.testing.examples.SimpleListTest

Available Assertions


```
import org.testng.Assert;

…

Assert.assertNull(expression [, message]);
Assert.assertNotNull(expression [,message]);
Assert.assertTrue(expression [, message]);
Assert.assertFalse(expression, [, message]);
Assert.assertEquals(actual, expected [, message]);
Assert.assertSame(actual, expected [, message]);
Assert.assertNotSame(actual, expected [, message]);
Assert.fail([message] [, Throwable]);
```

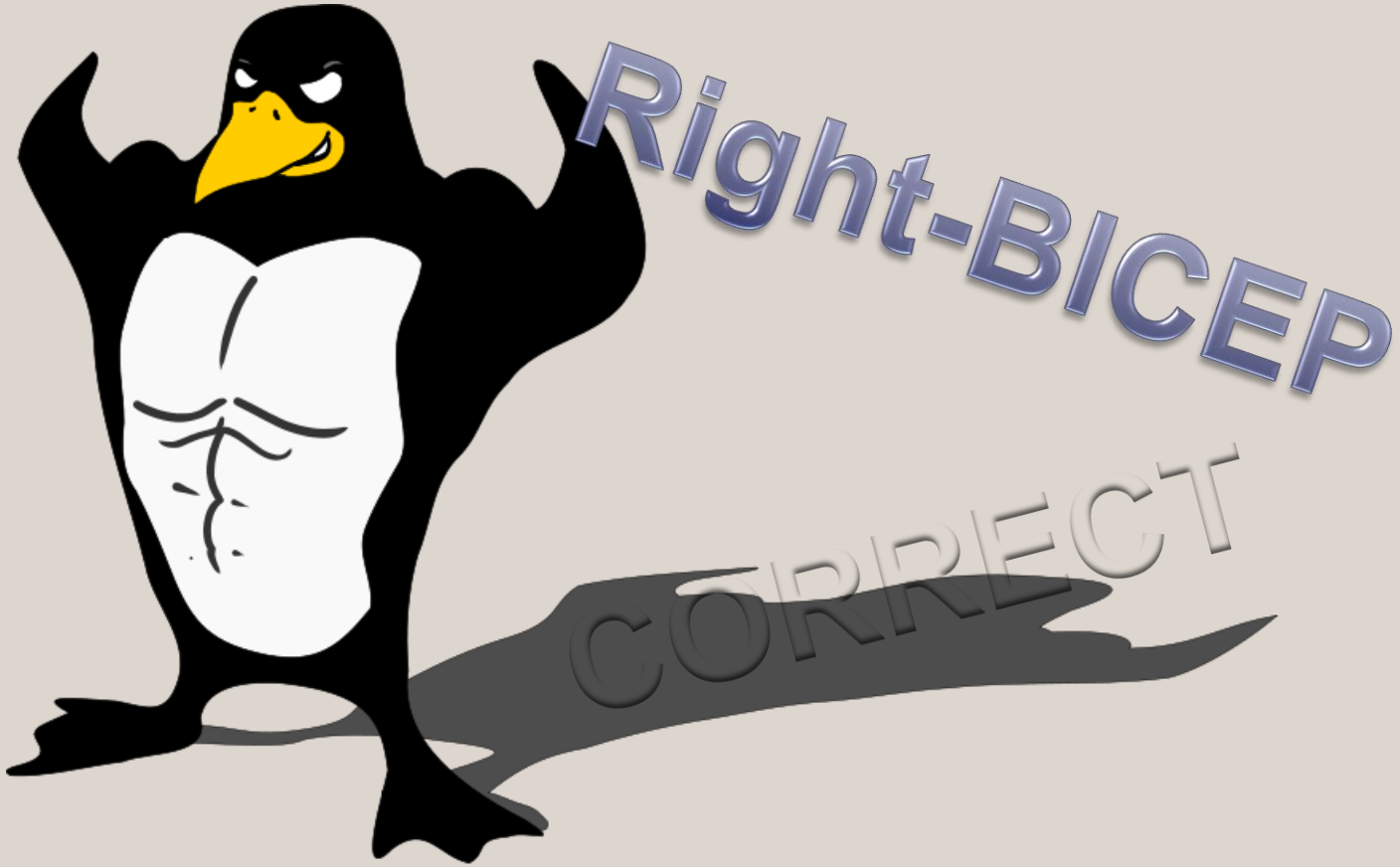
Available Assertions


```
import static org.testng.Assert.*;

…

assertNull(expression [, message]);
assertNotNull(expression [,message]);
assertTrue(expression [, message]);
assertFalse(expression, [, message]);
assertEquals(actual, expected [, message]);
assertSame(actual, expected [, message]);
assertNotSame(actual, expected [, message]);
fail([message] [, Throwable]);
```

Test Fixtures

org.lds.stack.training.testing.examples.SimpleListTest

What Should I Test?



The Right-BICEP

- **Right**
- **B**oundary conditions
- **I**nverse relationships
- **C**ross-check results
- **E**rror conditions
- **P**erformance characteristics

Testing "Right"-ness

- Does it do what I want?
- Does it do what I want all of the time?
- Can I depend on it?
- Does it document my intent?

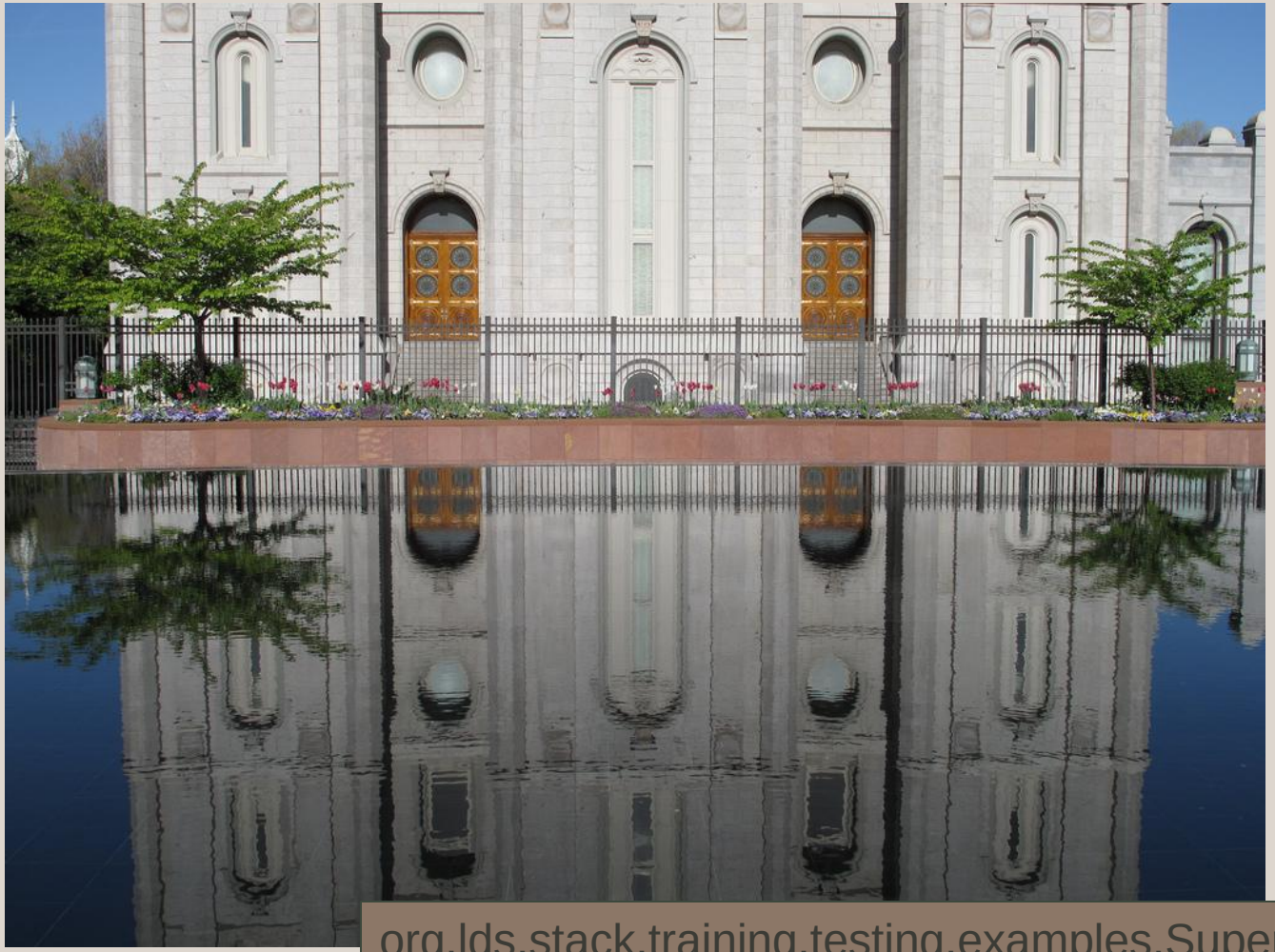
Boundary Conditions

- Empty or missing values
- Values that are out of range
- Poorly-formatted values
- Beginning or end of collections, arrays
- Is the value **CORRECT**?

Boundary Conditions - CORRECT

- Conformance
- Ordering
- Range
- Reference
- Existence
- Cardinality
- Time

Check Inverse Relationships



org.lds.stack.training.testing.examples.SuperMultiplier

Cross-Check Results

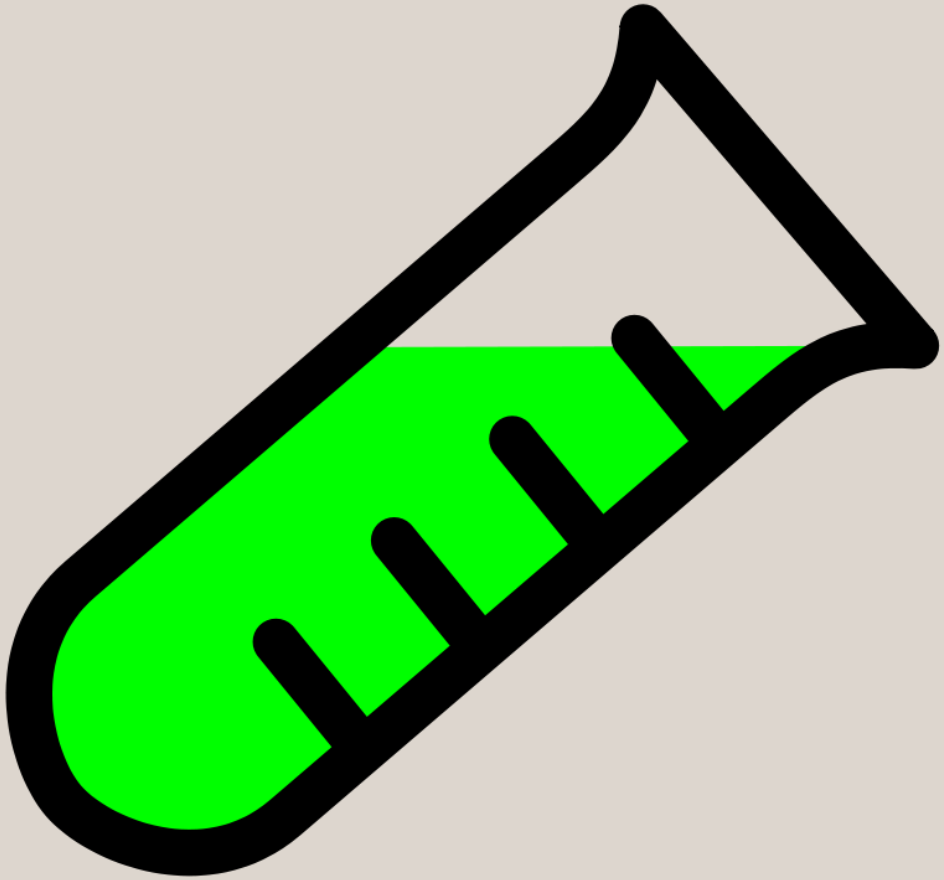
org.lds.stack.training.testing.examples.StringUtil

Force Error Conditions

Performance Characteristics

org.lds.stack.training.testing.examples.PrimeNumberCalculator

Lab 1



Testing Temporal Values

org.lds.stack.training.testing.examples.DatedPrimeNumberCalculator

Testing for Expected Exceptions


```
import static org.testng.Assert.*;
Import static org.testng.annotations.*;
…

@Test(expectedExceptions = IndexOutOfBoundsException.class)
public void testOutOfBoundsOnEmptyList() {
    new ArrayList<Object>().get(0);
}
```

org.lds.stack.training.testing.examples. ArrayListOutOfBoundsTest

TestUtils


```
<dependencies>
  …
  <dependency>
    <groupId>org.lds.stack.test</groupId>
    <artifactId>stack-test-unit</artifactId>
  </dependency>
  …
</dependencies>
```

```
import org.lds.stack.test.unit.TestUtils;
…
@Test
public void testMe() {
  TestUtils.testProperties(object);
}
```

org.lds.stack.training.testing.examples. ExampleBeanTest

TestUtils


```
import org.lds.stack.test.unit.TestUtils;
…
TestUtils.testProperties(Object);
TestUtils.testPropertiesExclude(Object, String…);
TestUtils.testPropertiesInclude(Object, String…);

TestUtils.testEquals(Serializable);
TestUtils.testEquals(Class<?>);
TestUtils.testEquals(Object, Object);
```

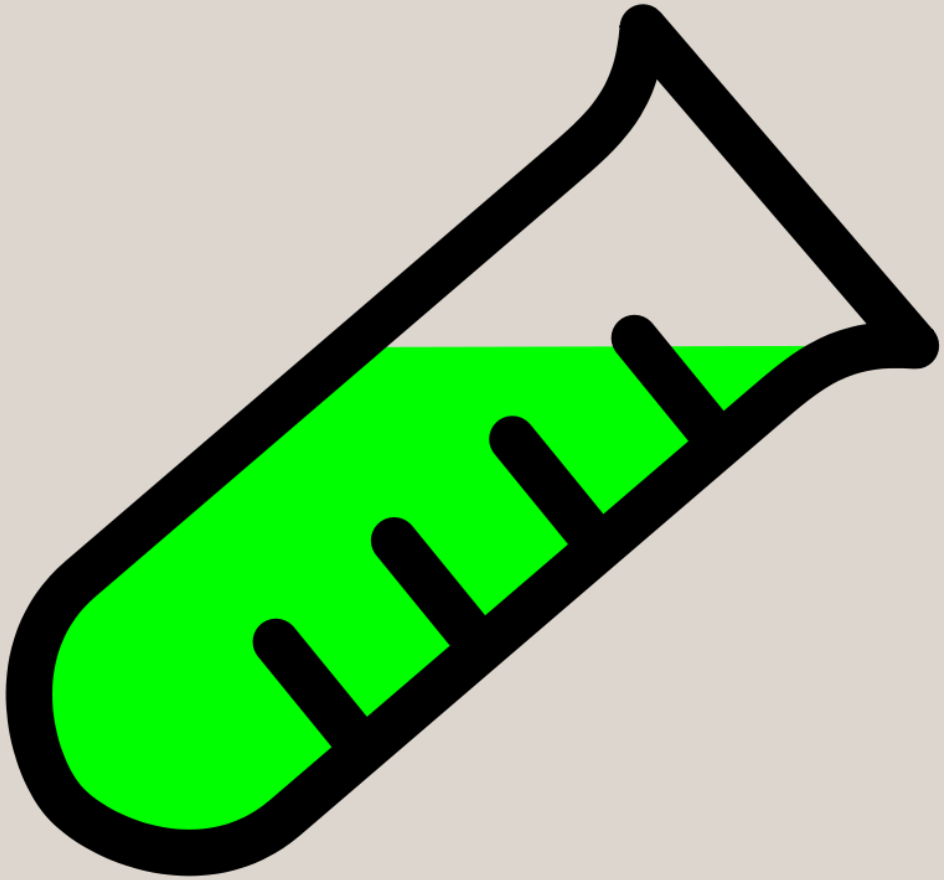
Using Tests to Maintain Bug-Free Code

- Write a failing test to test new feature.
- Write the new feature.
- When the test passes, you're done.
- Refine, refactor, improve.
- Run the tests again.

Using Unit Tests to Fix Bugs

- Write a new failing test to expose the bug.
- Write a fix.
- When the test passes, you're done.
- Refine, refactor, improve.
- Run the tests again.

Lab 2



Writing Clean, Maintainable Unit Tests

“Listening to Your Tests”

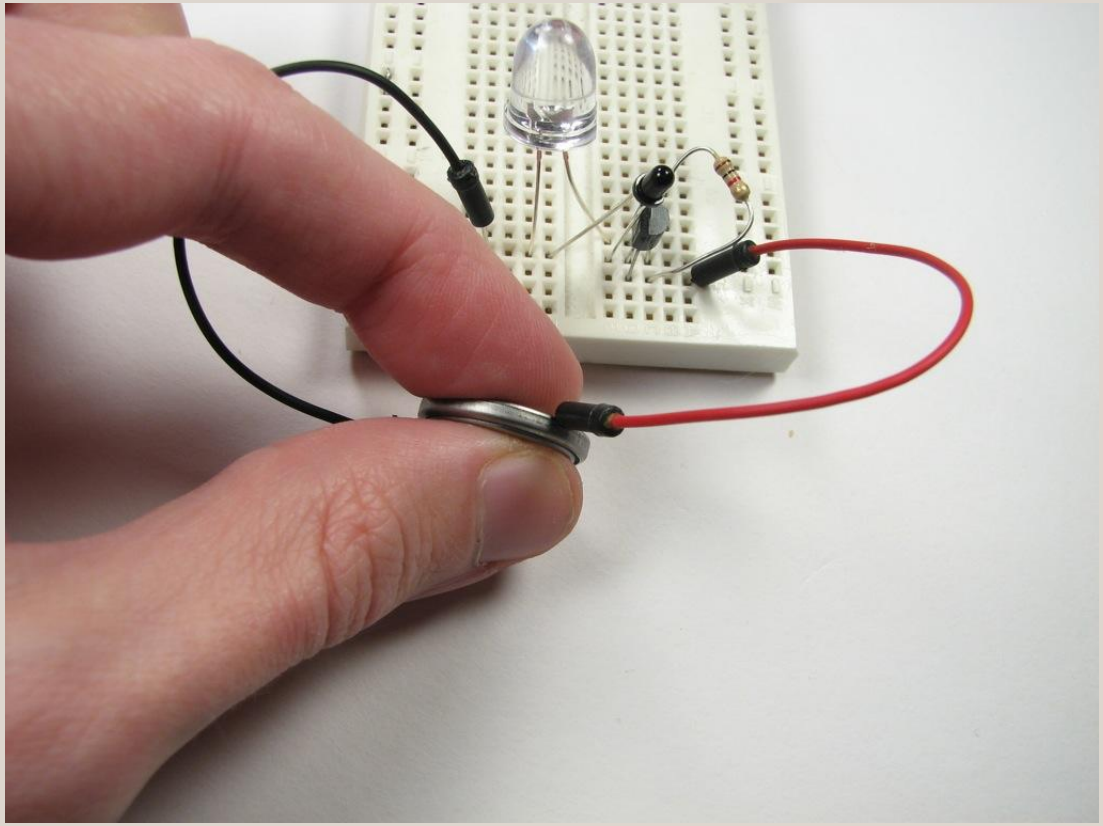
Testability

org.lds.stack.training.testing.examples.PasswordChecker

Tips and Tricks

Tip#1: One Assertion Per Test

Tip #2: Write a Test, Then Fix it



Tip #3: Keep Tests Independent

Tip #4: Use Good Design in Code & Tests

Tip #5: Test for Basic Correctness

"Given the myth of perfect technology, do we compute the right answer?"

- Do I ask collaborators the right questions?
- Can I handle the collaborators responses?
- Do not use integration tests to show basic correctness.

Resources

- “Pragmatic Unit Testing in Java with JUnit” - Andrew Hunt and David Thomas
- “Growing Object-Oriented Software, Guided By Tests” - Steve Freeman and Nat Pryce
- “Guide: Writing Testable Code” - http://misko.hevery.com/code-reviewers-guide/

Creative Commons

Name	Author	URL
Confusing signals	Luis Argerich	http://www.flickr.com/photos/lrargerich/2984777106/
Exam	Alberto G.	http://www.flickr.com/photos/albertogp123/5843577306/
big engine, little car!	stephen bowler	http://www.flickr.com/photos/50826080@N00/2668301430/
Holley Carb	Ryan Frost	http://www.flickr.com/photos/ryanfrost/1198884099/
Double Quad Engine	Nick Young	http://www.flickr.com/photos/braintoad/1389718928/
Volkswagen Nils electric car concept at the Frankfurt Motor Show IAA 2011	Autoviva	http://www.flickr.com/photos/autovivacom/6143741096/

Creative Commons

Name	Author	URL
1960 Bathroom	Ruthanne Reid	http://www.flickr.com/photos/doortoriver/2994024026/
Reflection	Irwin-Scott	http://www.flickr.com/photos/irwin-scott/3606795148/
Making a list	Gerry Thomasen	http://www.flickr.com/photos/gerrythomasen/349799269/
Stopwatch	William Warby	http://www.flickr.com/photos/wwarby/3296379139/
Slide Rule	The Last Cookie	http://www.flickr.com/photos/cabeel/2197888941/
Calculator	JakeandLindsay Sherbert	http://www.flickr.com/photos/jakeandlindsay/5639214967/

Creative Commons

Name	Author	URL
hourglass 4	Erik Fitzpatrick	http://www.flickr.com/photos/22244945@N00/3278869535/
LG Snowboard FIS World Cup	LGEPR	http://www.flickr.com/photos/lge/5435939492/
Maybe A U-Turn Would Be Best	Bex Ross	http://www.flickr.com/photos/bexross/2636100175/
one	andrechinn	http://www.flickr.com/photos/andrec/2893549851/
Test out	Windell Oskay	http://www.flickr.com/photos/oskay/2104862955/
Engineer working on plans for Lake Union area, circa 1960s	Seattle Municipal Archives	http://www.flickr.com/photos/seattlemunicipalarchives/2713475713/

Creative Commons

Name	Author	URL
Picture 011	william S	http://www.flickr.com/photos/freerideallianc e/211488004/
dishwasher	Jo Bourne	http://www.flickr.com/photos/66992990@N 00/4819555022/
august 09	Megan Westerby	http://www.flickr.com/photos/meganwest/3 831943787/
muscle	hector gomez	http://openclipart.org/detail/71467/muscle- by-hector-gomez
City water testing laboratory, 1948	Seattle Municipal Archives	http://www.flickr.com/photos/seattlemunici palarchives/3739366791/
Lab icon 2	pitr	http://openclipart.org/detail/22628/lab- icon-2-by-pitr