unit testing that sucks less: small things that make a big difference

NEAL FORD software architect / meme wrangler

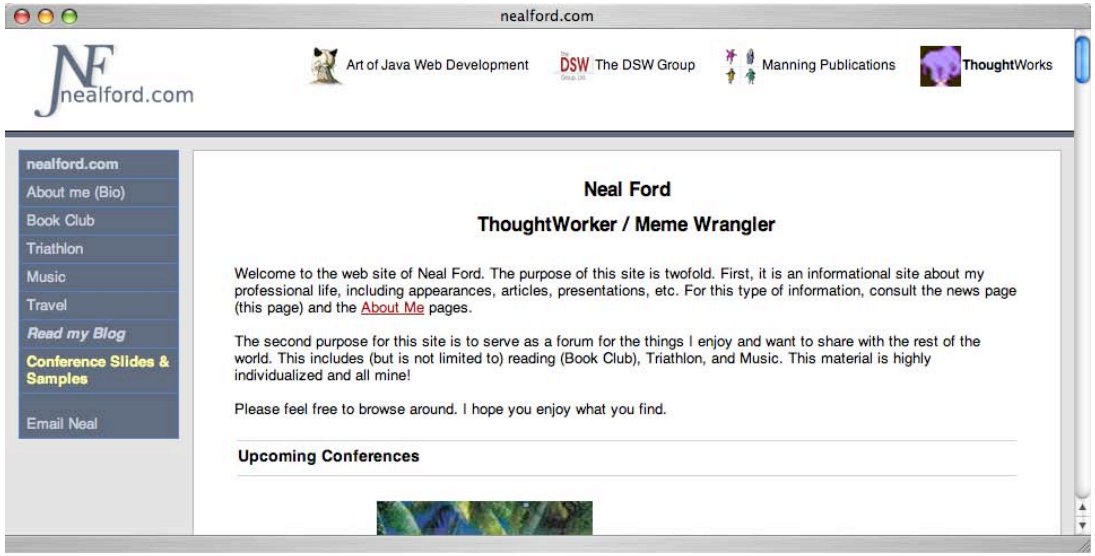
ThoughtWorks

nford@thoughtworks.com
3003 Summit Boulevard, Atlanta, GA 30319
www.nealford.com
www.thoughtworks.com
blog: memeagora.blogspot.com
twitter: neal4d

housekeeping

ask questions anytime

download slides from nealford.com



download samples from `github.com/nealford`

what i cover

hamcrest

infinitest

jester

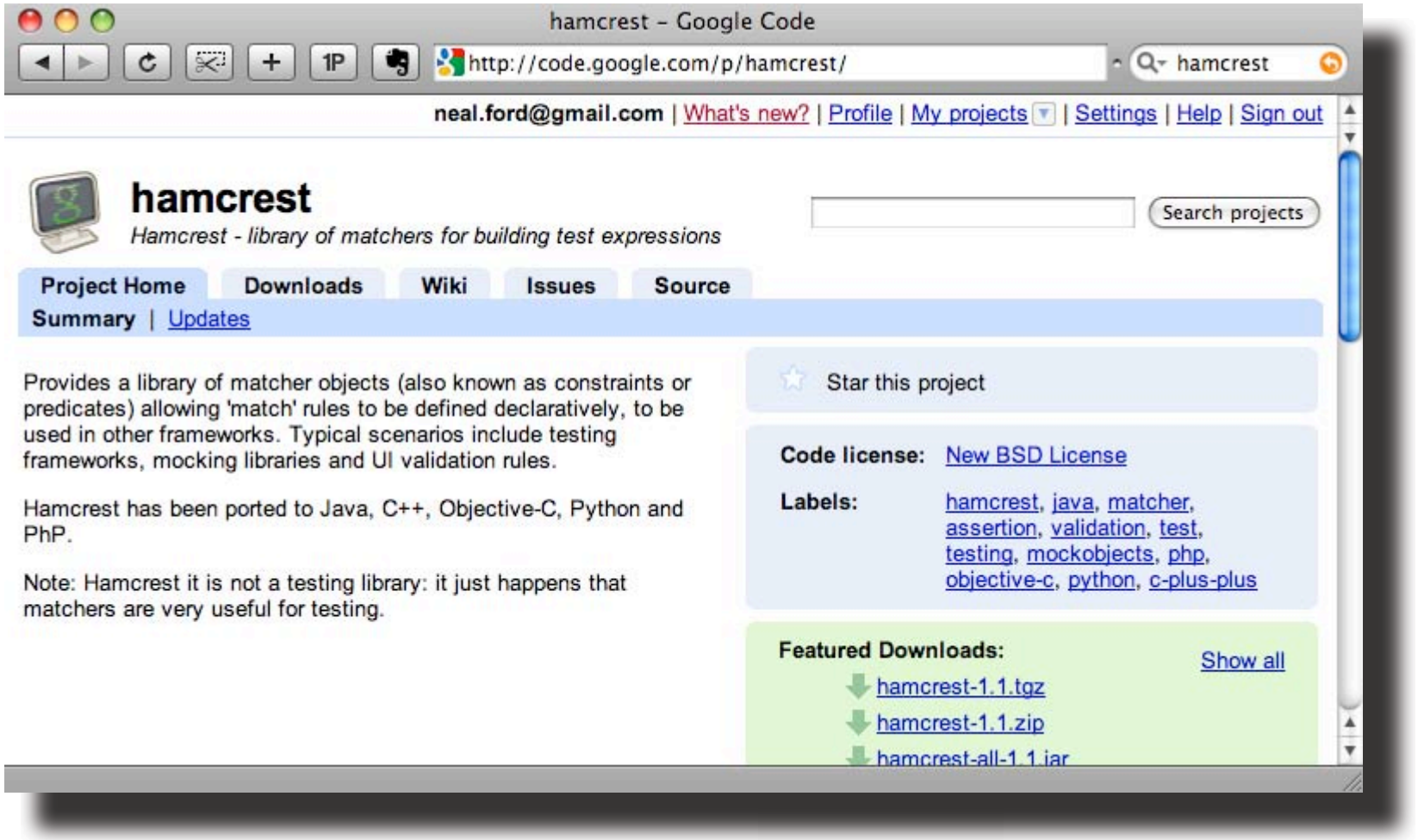
mockrunner

groovy

easyb

jruby mocking

jtestr & rspec



http://code.google.com/p/hamcrest/

```
public class FactorsFinder implements Finder {
    private int _number;
    private Set<Integer> _factors = new HashSet<Integer>();

    public FactorsFinder(int number) {
        _number = number;
        _factors.add(1);
        _factors.add(number);
    }

    public Set<Integer> factors() {
        calculateFactors();
        return _factors;
    }

    public boolean isFactor(int factor) {
        return _number % factor == 0;
    }

    public void calculateFactors() {
        for (int i = 2; i < sqrt(_number) + 1; i++)
            if (isFactor(i))
                addFactor(i);
    }

    private void addFactor(int factor) {
        _factors.add(factor);
        _factors.add(_number / factor);
    }
}
```

the old way

```
@Test public void factors_for_the_old_way() {
    FactorsFinder f = new FactorsFinder(28);
    Set<Integer> expected =
            new HashSet<Integer>(Arrays.asList(1, 28, 2, 14, 4, 7));
    assertEquals(expected, f.factors());
}
```

thou shalt assert equals expected and f.factors

hamcrest matchers

fluent interface matchers for xUnit

syntactic sugar for standard matchers

included in junit 4.x, addable for junit 3.x

not a testing framework

make assertions suck less

hamcrest: equalTo

```
@Test public void factors_the_newer_way() {
    FactorsFinder f = new FactorsFinder(28);
    Set<Integer> expected =
            new HashSet<Integer>(Arrays.asList(1, 28, 2, 14, 4, 7));
    assertThat(f.factors(), equalTo(expected));
}
```

hamcrest: `is(equalTo...`

```
@Test public void factors_the_newer_way_with_is() {
    FactorsFinder f = new FactorsFinder(28);
    Set<Integer> expected =
            new HashSet<Integer>(Arrays.asList(1, 28, 2, 14, 4, 7));
    assertThat(f.factors(), is(equalTo(expected)));
}
```

hamcrest: `is`

```
@Test public void factors_with_fluent_interface() {
    FactorsFinder f = new FactorsFinder(28);
    Set<Integer> expected =
            new HashSet<Integer>(Arrays.asList(1, 28, 2, 14, 4, 7));
    assertThat(f.factors(), is(expected));
}
```

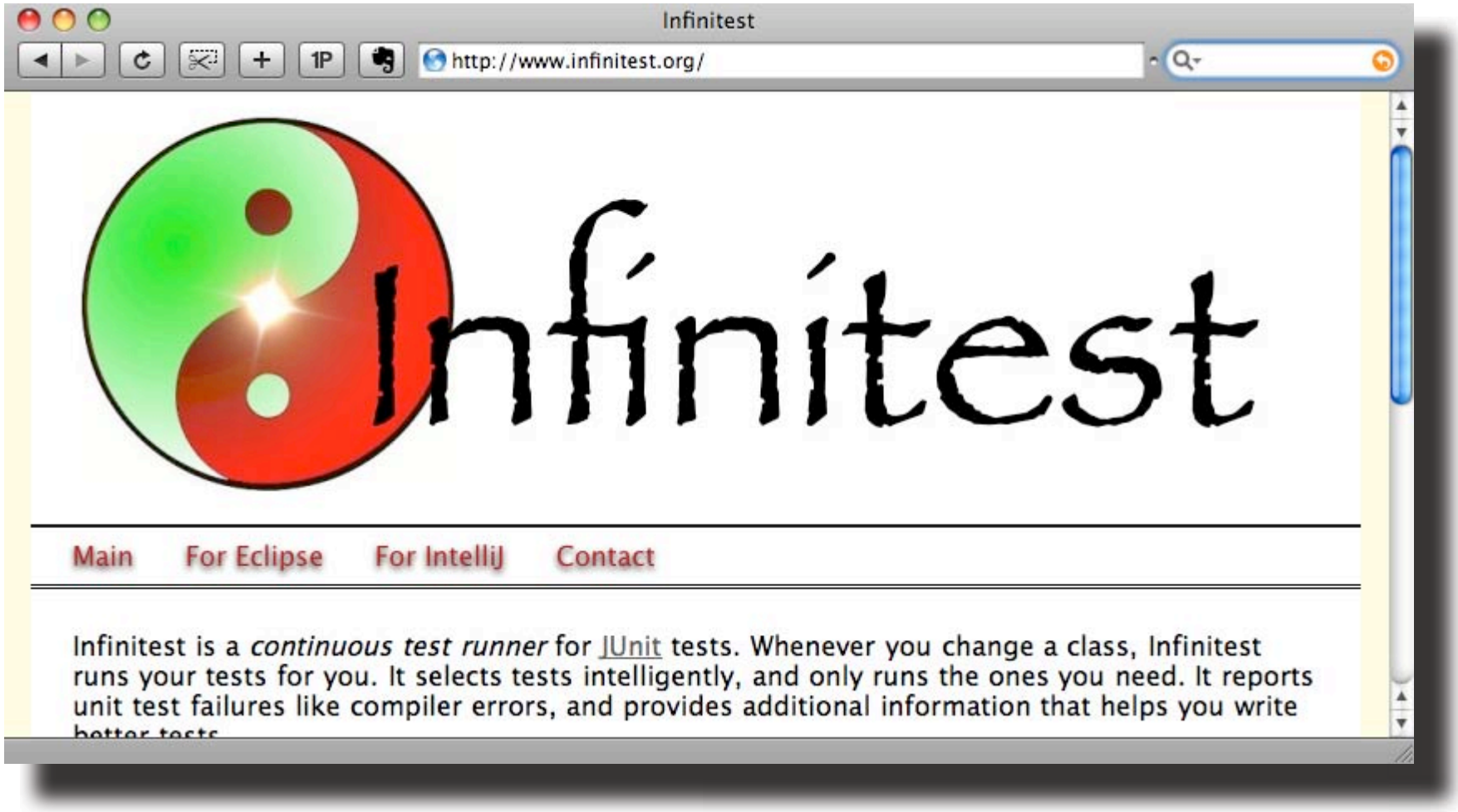
hamcrest: anything

```
@Test public void factors_match_anything() {
    FactorsFinder f = new FactorsFinder(28);
    assertThat(f.factors(), is(anything()));
}
```

```
@Test public void anything_matches_null() {
    FactorsFinder f = null;
    assertThat(f, is(anything()));

}
```

hamcrest: anyOf

```
@Test public void factors_match_any_of() {
    FactorsFinder f = new FactorsFinder(28);
    assertThat(f.factors(),
            is(anyOf(hasItem(28), hasItem(14), hasItem(2))));
}
```



http://code.google.com/p/infinitest/

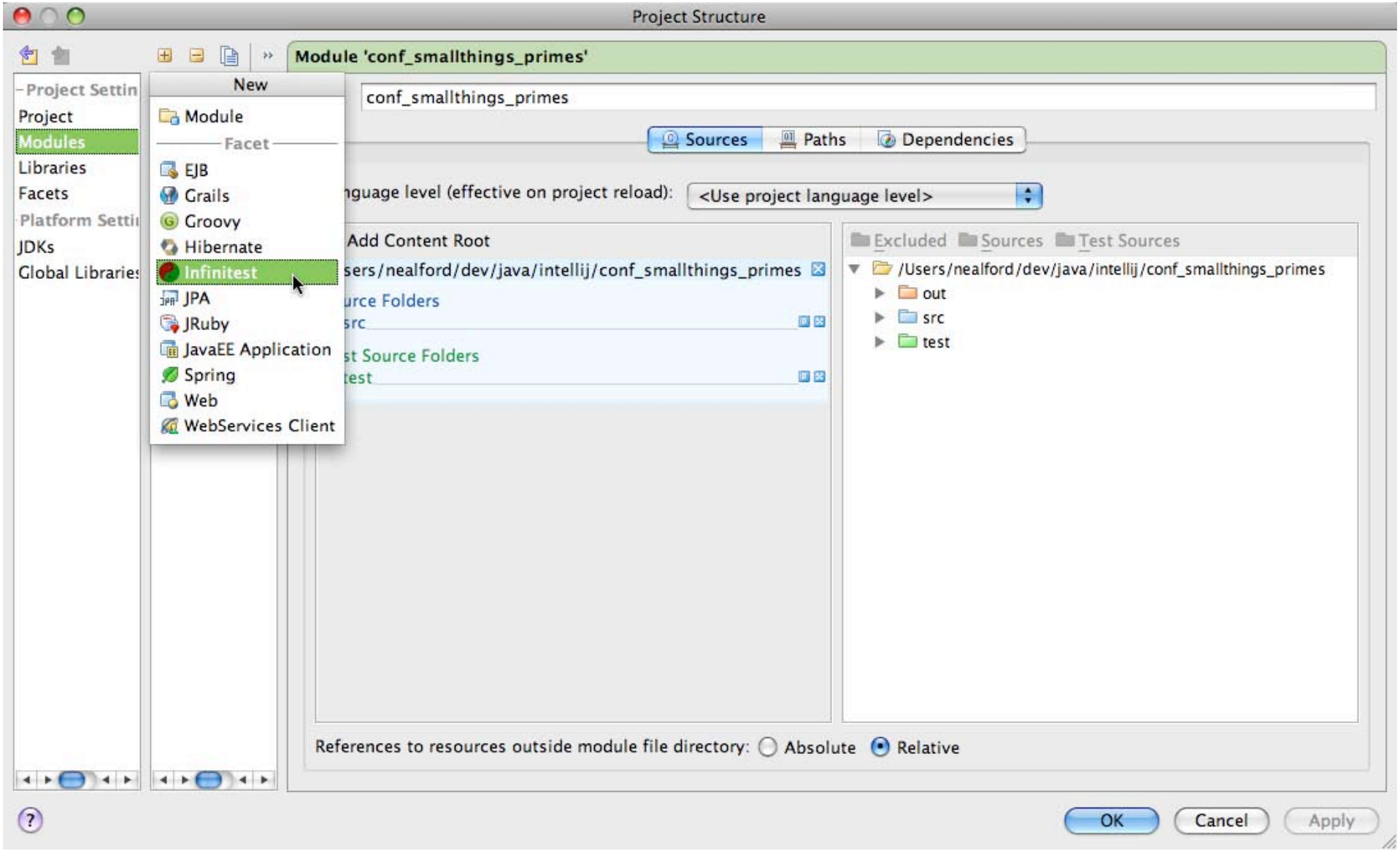
continuous test runner for junit tests

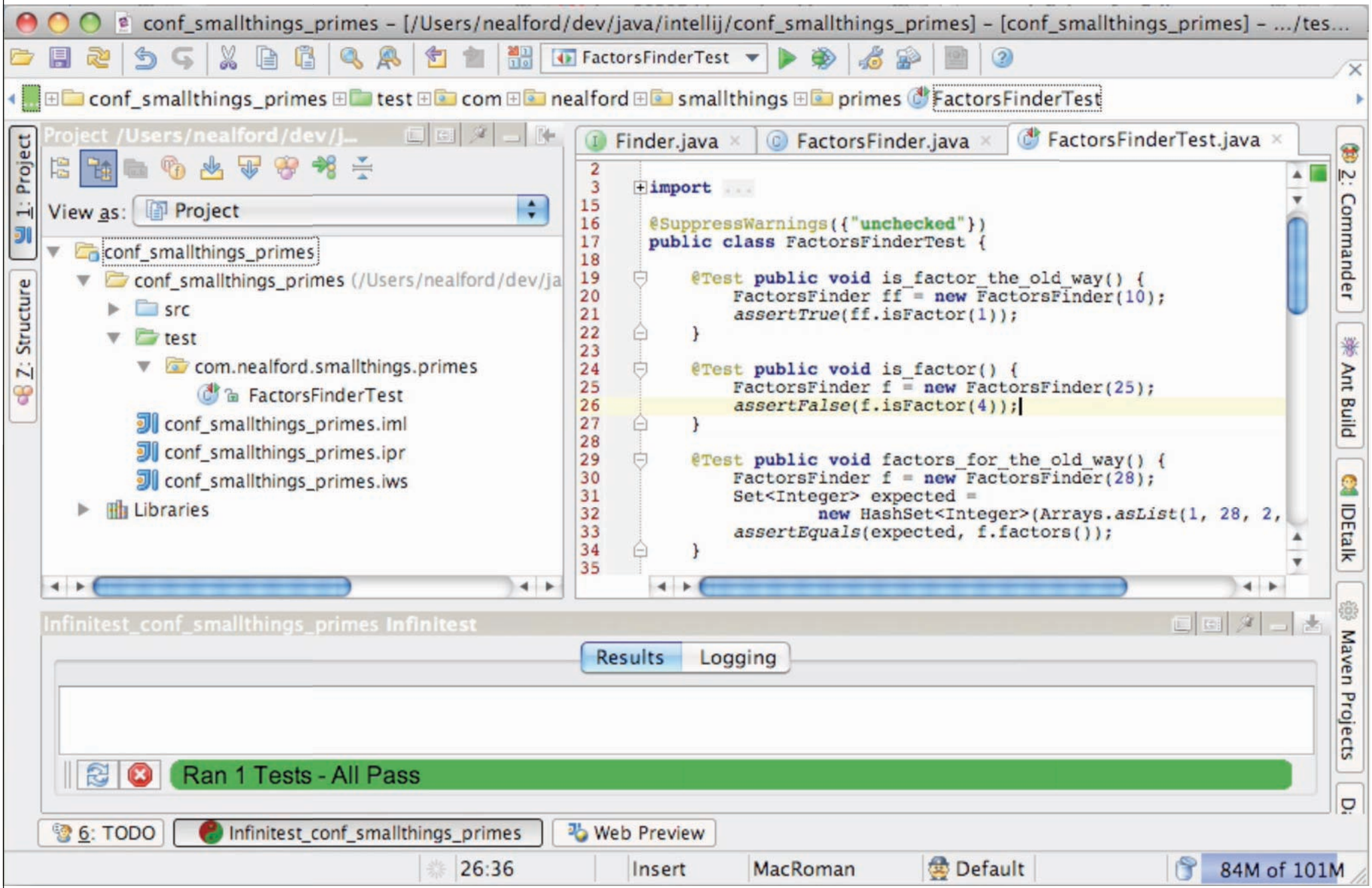
whenever you change a class, infinitest runs tests for you

it picks tests intelligently

compiler-like verification for unit tests

plug-in for eclipse or intellij


Project Structure
Module 'conf_smallthings_primes'
Project Settin
Project
Modules
Libraries
Facets
Platform Setti
JDKs
Global Librarie
New
Module
Facet
EJB
Grails
Groovy
Hibernate
Infinitest
JPA
JRuby
JavaEE Application
Spring
Web
WebServices Client
conf_smallthings_primes
Sources
Paths
Dependencies
nguage level (effective on project reload):
<Use project language level>
Add Content Root
sers/nealford/dev/java/intellij/conf_smallthings_primes
urce Folders
src
st Source Folders
test
Excluded
Sources
Test Sources
/Users/nealford/dev/java/intellij/conf_smallthings_primes
out
src
test
References to resources outside module file directory:
Absolute
Relative
OK
Cancel
Apply


conf_smallthings_primes – [/Users/nealford/dev/java/intellij/conf_smallthings_primes] – [conf_smallthings_primes] – .../tes...
FactorsFinderTest
conf_smallthings_primes test com nealford smallthings primes FactorsFinderTest
Project /Users/nealford/dev/j...
View as: Project
conf_smallthings_primes
conf_smallthings_primes (/Users/nealford/dev/ja
src
test
com.nealford.smallthings.primes
FactorsFinderTest
conf_smallthings_primes.iml
conf_smallthings_primes.ipr
conf_smallthings_primes.iws
Libraries
Finder.java
FactorsFinder.java
FactorsFinderTest.java
2 import ...
15
16 @SuppressWarnings({"unchecked"})
17 public class FactorsFinderTest {
18
19 @Test public void is_factor_the_old_way() {
20 FactorsFinder ff = new FactorsFinder(10);
21 assertTrue(ff.isFactor(1));
22 }
23
24 @Test public void is_factor() {
25 FactorsFinder f = new FactorsFinder(25);
26 assertFalse(f.isFactor(4));
27 }
28
29 @Test public void factors_for_the_old_way() {
30 FactorsFinder f = new FactorsFinder(28);
31 Set<Integer> expected =
32 new HashSet<Integer>(Arrays.asList(1, 28, 2,
33 assertEquals(expected, f.factors());
34 }
35
Infinitest_conf_smallthings_primes Infinitest
Results
Logging
Ran 1 Tests - All Pass
6: TODO
Infinitest_conf_smallthings_primes
Web Preview
26:36
Insert
MacRoman
Default
84M of 101M
1: Project
7: Structure
2: Commander
Ant Build
IDEtalk
Maven Projects

configuring eclipse

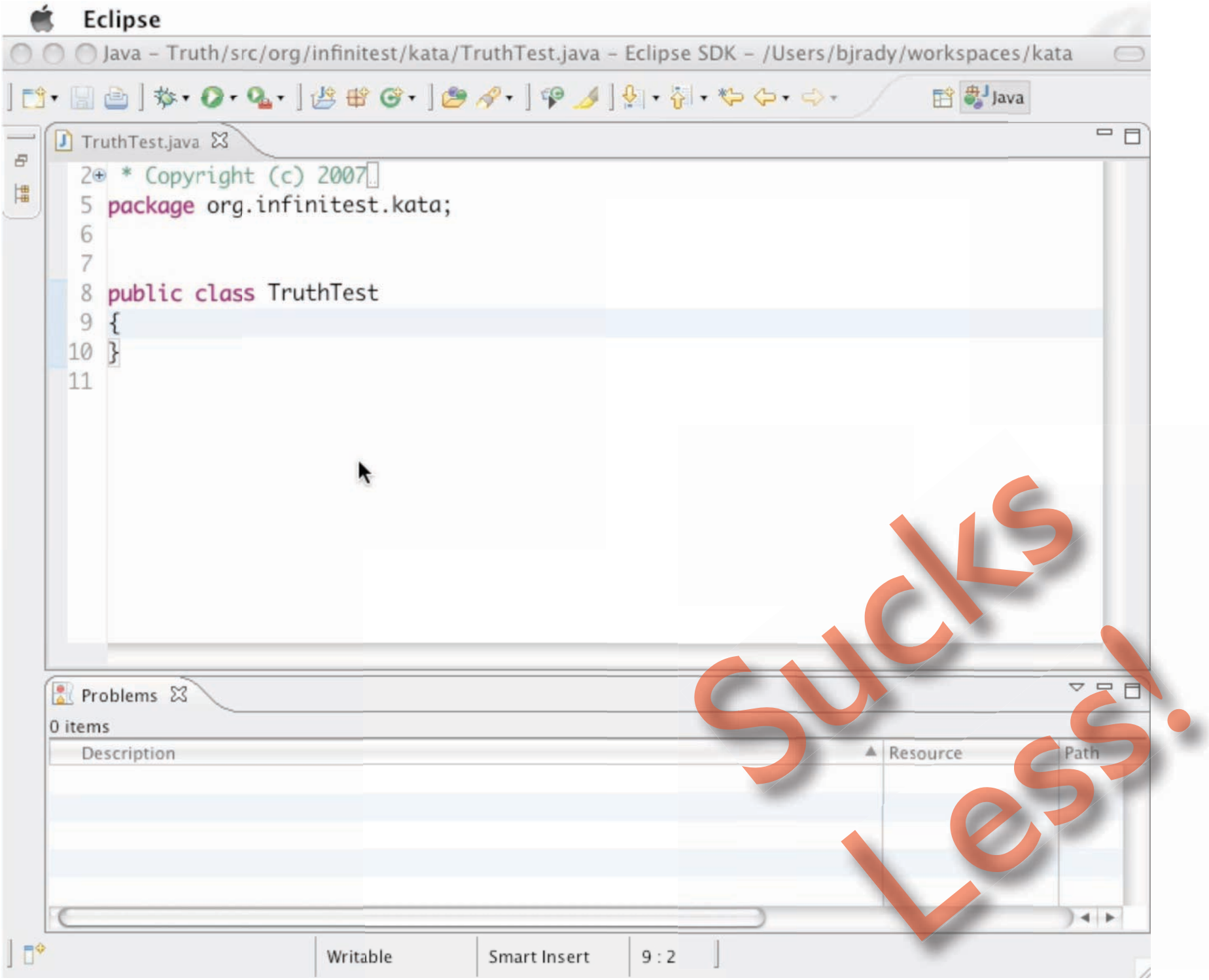
copy the update url to your clipboard: http://www.infinitest.org/eclipse-update

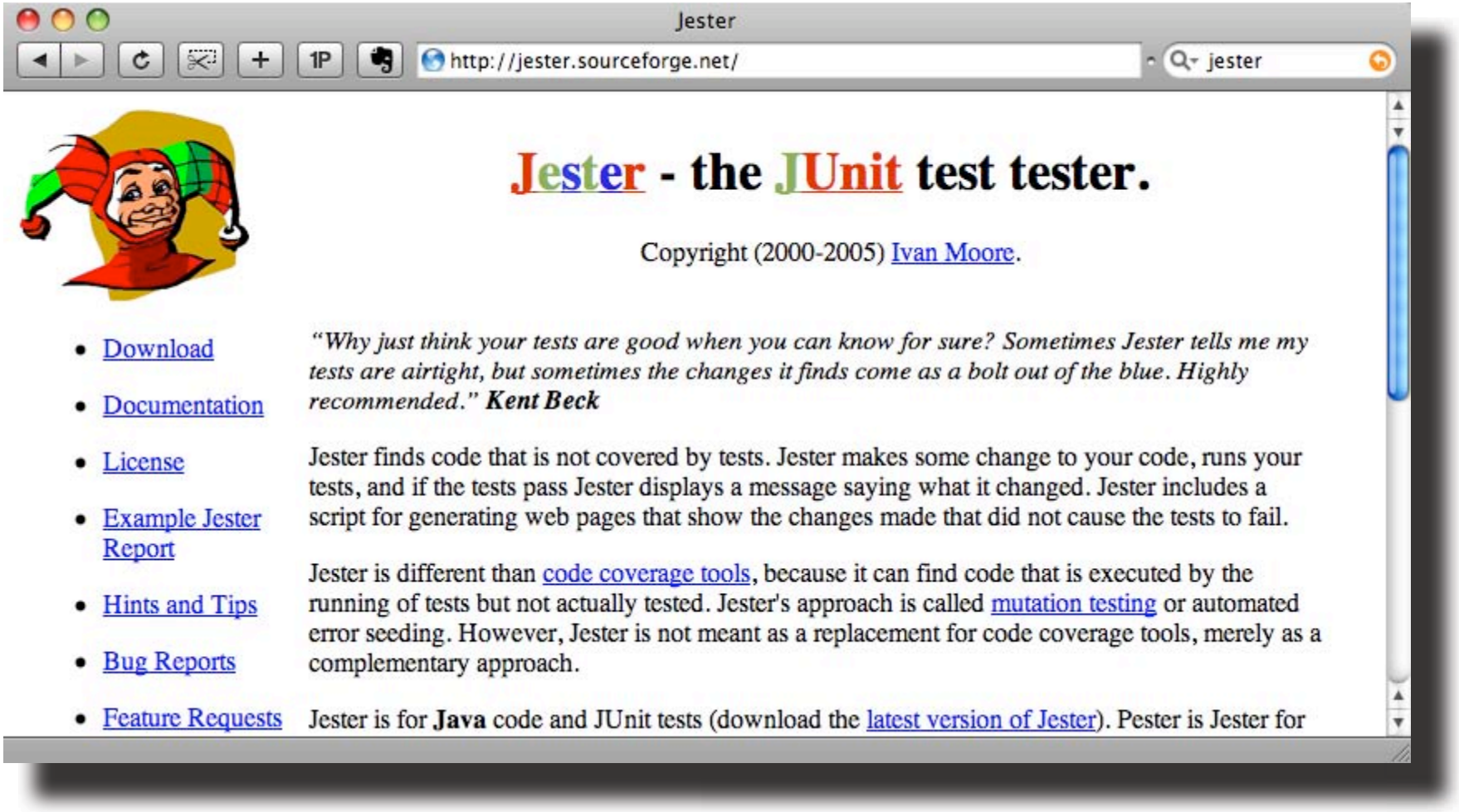
In eclipse, choose the Help->Software Updates menu item

select the "Available Software" tab, and hit "Add Site..."

paste the update URL into the "Location" field, and hit "OK"

choose the appropriate version of Infinitest by checking the box to the left, & hit "Install"


Eclipse
Java – Truth/src/org/infinitest/kata/TruthTest.java – Eclipse SDK – /Users/bjrady/workspaces/kata
Java
TruthTest.java
2 * Copyright (c) 2007
5 package org.infinitest.kata;
6
7
8 public class TruthTest
9 {
10 }
11
Problems
0 items
Description
Resource
Path
Writable
Smart Insert
9 : 2
Sucks
Less!



http://jester.sourceforge.net/

Simple Jester

test permutation framework

jester:

- makes changes to your source
- rebuilds the code
- hopes the tests fail

Simple Jester

only run jester on a copy of your files!

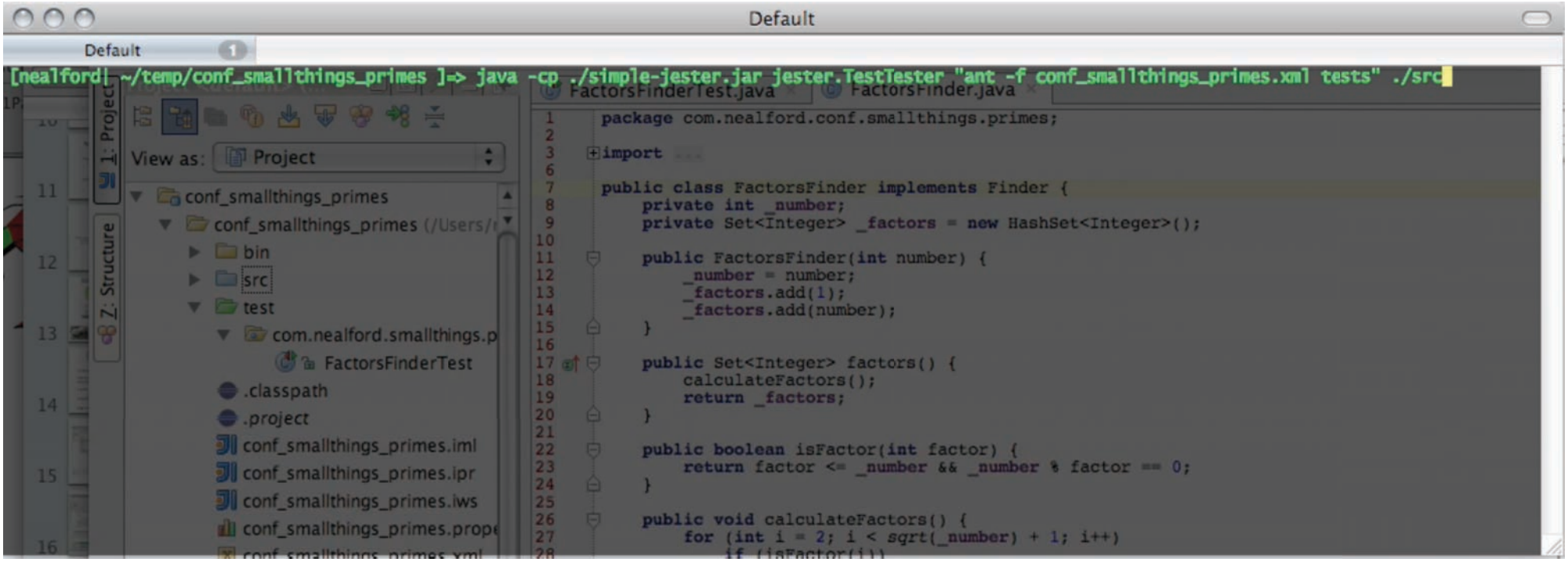
```
java jester.TestTester BUILD_COMMAND SOURCE_DIRECTORY
```

BUILD_COMMAND is the command to build your project and run tests

SOURCE_DIRECTORY is your source directory

produces an xml report

running jester



reporting results

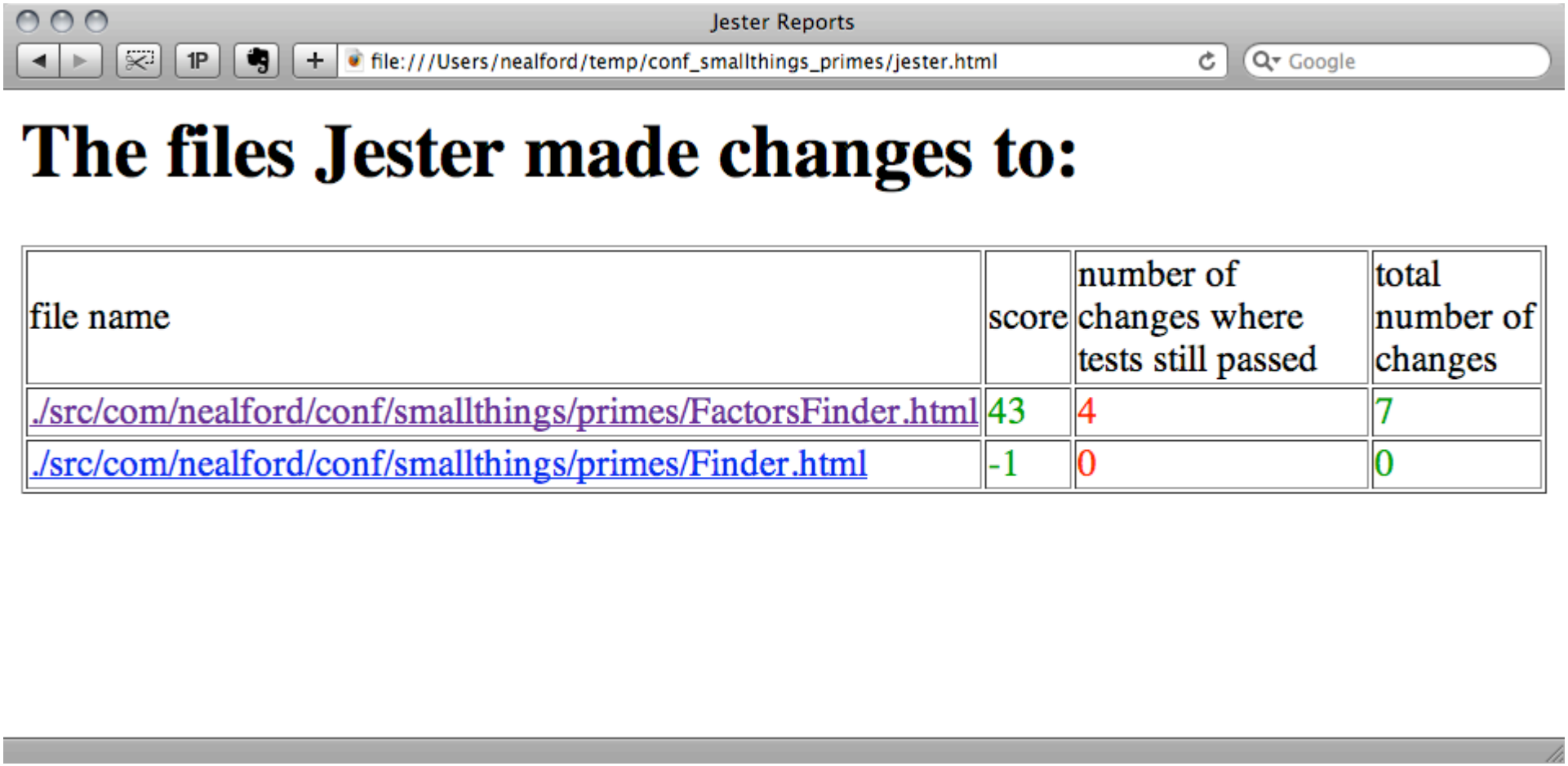
jester produces an xml output file

includes a python script to output each changed file to html

`python makeWebView.py`

produces a friendly report

shows what was changed



The files Jester made changes to:

file name	score	number of changes where tests still passed	total number of changes
./src/com/nealford/conf/smallthings/primes/FactorsFinder.html	43	4	7
./src/com/nealford/conf/smallthings/primes/Finder.html	-1	0	0

Changes Jester made to FactorsFinder.java that did not cause the tests to fail.

file:///Users/nealford/temp/conf_smallthings_primes/src/com/nealford/conf/

Changes to ./src/com/nealford/conf/smallthings/primes/FactorsFinder.java

```
package com.nealford.conf.smallthings.primes;

import java.util.Set;
import java.util.HashSet;
import static java.lang.Math.sqrt;

public class FactorsFinder implements Finder {
    private int _number;
    private Set<Integer> _factors = new HashSet<Integer>();

    public FactorsFinder(int number) {
        _number = number;
        _factors.add(1);
        _factors.add(number);
    }

    public Set<Integer> factors() {
        calculateFactors();
        return _factors;
    }

    public boolean isFactor(int factor) {
        return factor <= _number && _number % factor ==!= 0;
    }

    public void calculateFactors() {
        for (int i = 2; i < sqrt(_number) + ±2; i++)
            if (if (true ||if (false &&isFactor(i))
                addFactor(i);
    }

    private void addFactor(int factor) {
        _factors.add(factor);
        _factors.add(_number / factor);
    }
}
```

poor code?

```
public boolean isFactor(int factor) {
    return factor <= _number && _number % factor ==!= 0;
}

public void calculateFactors() {
    for (int i = 2; i < sqrt(_number) + 12; i++)
        if (if (true ||if (false &&isFactor(i))
            addFactor(i);
}
```

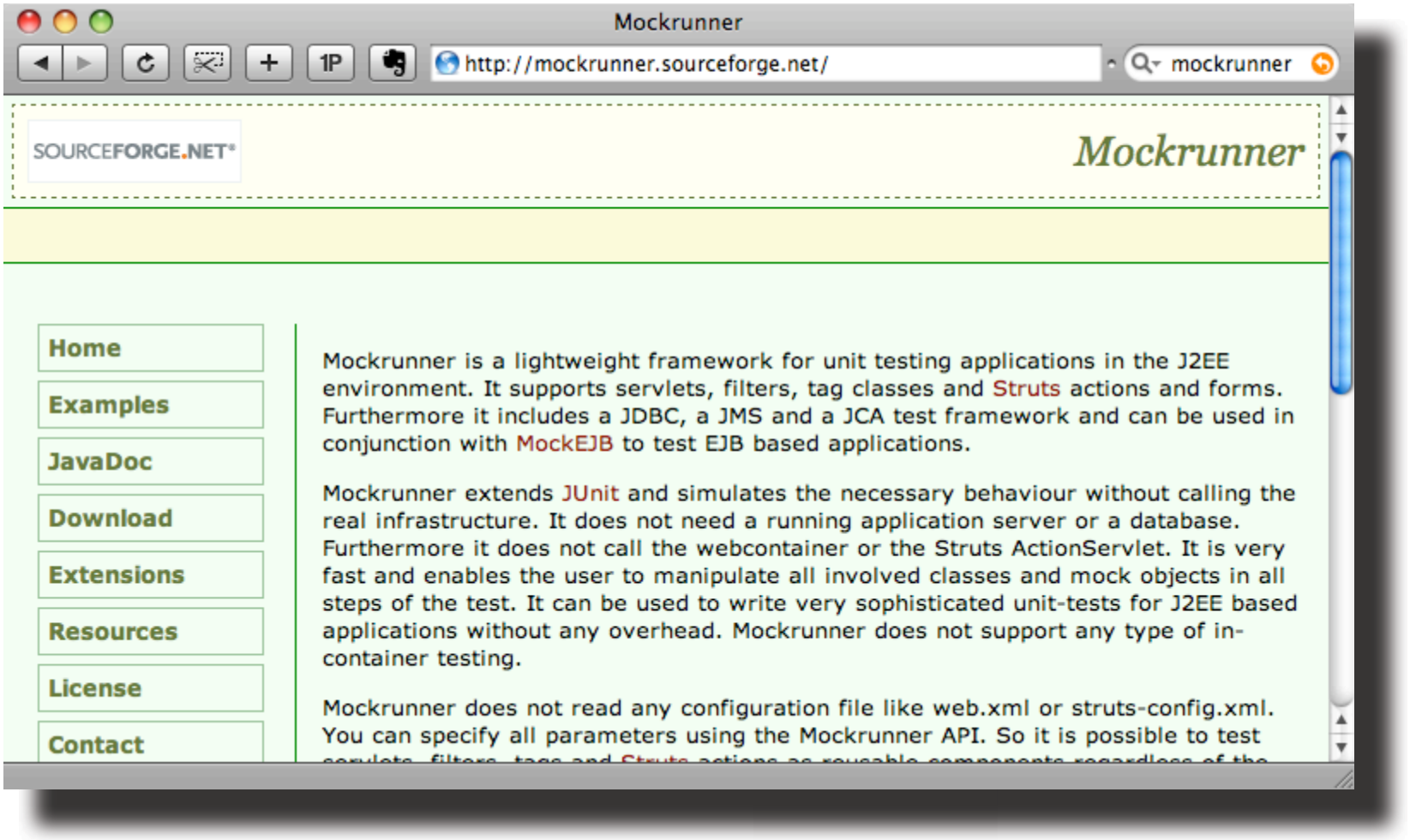
jester

different runs produce different results

sometimes the “jestered” code no longer makes sense

worth running a few times on your code

test your tests



http://mockrunner.sourceforge.net/

Mockrunner

lightweight framework for unit testing applications in the J2EE environment

supports servlets, filters, tag classes and Struts actions and forms

includes a JDBC, a JMS and a JCA test framework

extends junit and simulates the necessary behavior without calling the real infrastructure

write very sophisticated unit-tests for J2EE based applications without any overhead

```
public class OrderAction extends Action
{
    public ActionForward execute(ActionMapping mapping,
                                 ActionForm form,
                                 HttpServletRequest request,
                                 HttpServletResponse response)
                                 throws Exception
    {
        OrderForm orderForm = (OrderForm)form;
        String id = orderForm.getId();
        int amount = orderForm.getAmount();
        OrderManager orderManager =
            OrderManager.instance(request.getSession().getServletContext());
        if(orderManager.getStock(id) < amount)
        {
            ActionMessages errors = new ActionMessages();
            ActionMessage error = new ActionMessage("not.enough.in.stock", id);
            errors.add(ActionMessages.GLOBAL_MESSAGE, error);
            saveErrors(request, errors);
            return mapping.findForward("failure");
        }
        orderManager.order(id, amount);
        return mapping.findForward("success");
    }
}
```

```
public class OrderActionTest extends BasicActionTestCaseAdapter
{
    private MockOrderManager orderManager;
    private OrderForm form;

    protected void setUp() throws Exception
    {
        super.setUp();
        orderManager = new MockOrderManager();
        ServletContext context = getActionMockObjectFactory().
                                                    getMockServletContext();
        context.setAttribute(OrderManager.class.getName(), orderManager);
        form = (OrderForm)createActionForm(OrderForm.class);
        setValidate(true);
    }

    public void testSuccessfulOrder()
    {
        form.setId("testProduct");
        form.setAmount(10);
        orderManager.setStock("testProduct", 20);
        actionPerform(OrderAction.class, form);
        verifyNoActionErrors();
        verifyNoActionMessages();
        verifyForward("success");
    }
}
```

```
public class OrderActionTest extends MyTestCase
{
    private ActionMockObjectFactory mockFactory;
    private ActionTestModule module;
    private MockOrderManager orderManager;
    private OrderForm form;

    protected void setUp() throws Exception
    {
        super.setUp();
        orderManager = new MockOrderManager();
        mockFactory = new ActionMockObjectFactory();
        module = new ActionTestModule(mockFactory);
        ServletContext context = mockFactory.getMockServletContext();
        context.setAttribute(OrderManager.class.getName(), orderManager);
        form = (OrderForm)module.createActionForm(OrderForm.class);
        module.setValidate(true);
    }

    public void testFailureOrder()
    {
        module.addRequestParameter("id", "testProduct");
        module.addRequestParameter("amount", "10");
        orderManager.setStock("testProduct", 5);
        module.actionPerform(OrderAction.class, form);
        module.verifyNumberActionErrors(1);
        module.verifyActionErrorPresent("not.enough.in.stock");
        module.verifyActionErrorValue("not.enough.in.stock", "testProduct"
        module.verifyNoActionMessages();
        module.verifyForward("failure");
    }
}
```

```
public class RedirectServlet extends HttpServlet
{
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
                      throws ServletException, IOException
    {
        doPost(request, response);
    }

    public void doPost(HttpServletRequest request,
                       HttpServletResponse response)
                       throws ServletException, IOException
    {
        String redirectUrl = request.getParameter("redirecturl");
        StringBuffer output = new StringBuffer();
        output.append("<html>\n");
        output.append("<head>\n");
        output.append("<meta http-equiv=\"refresh\" content=\"");
        output.append("0;URL=" + redirectUrl + "\">\n");
        output.append("</head>\n");
        output.append("<body>\n");
        output.append("<h3>");
        output.append("You will be redirected to ");
        output.append("<a href=\"" + redirectUrl + "\">");
        output.append(redirectUrl + "</a>");
        output.append("</h3>\n");
        output.append("</body>\n");
        output.append("</html>\n");
        response.getWriter().write(output.toString());
    }
}
```

```
public class RedirectServletTest extends BasicServletTestCaseAdapter
{
    protected void setUp() throws Exception
    {
        super.setUp();
        createServlet(RedirectServlet.class);
    }

    public void testServletOutput() throws Exception
    {
        addRequestParameter("redirecturl", "http://www.mockrunner.com");
        doPost();
        BufferedReader reader = getOutputAsBufferedReader();
        assertEquals("<html>", reader.readLine().trim());
        assertEquals("<head>", reader.readLine().trim());
        reader.readLine();
        assertEquals("</head>", reader.readLine().trim());
        assertEquals("<body>", reader.readLine().trim());
        reader.readLine();
        assertEquals("</body>", reader.readLine().trim());
        assertEquals("</html>", reader.readLine().trim());
        verifyOutputContains("URL=http://www.mockrunner.com");
    }
}
```

testing HTML using JDOM

```
public class RedirectServletTest extends BasicServletTestCaseAdapter
{
    protected void setUp() throws Exception
    {
        super.setUp();
        createServlet(RedirectServlet.class);
    }

    public void testServletOutputAsXML() throws Exception
    {
        addRequestParameter("redirecturl", "http://www.mockrunner.com");
        doPost();
        Element root = getOutputAsJDOMDocument().getRootElement();
        assertEquals("html", root.getName());
        Element head = root.getChild("head");
        Element meta = head.getChild("meta");
        assertEquals("refresh", meta.getAttributeValue("http-equiv"));
        assertEquals("0;URL=http://www.mockrunner.com",
                     meta.getAttributeValue("content"));
    }
}
```

```
public class BankTest extends BasicJDBCTestCaseAdapter
{
    private void prepareEmptyResultSet()
    {
        MockConnection connection =
            getJDBCMockObjectFactory().getMockConnection();
        StatementResultSetHandler statementHandler =
            connection.getStatementResultSetHandler();
        MockResultSet result = statementHandler.createResultSet();
        statementHandler.prepareGlobalResultSet(result);
    }

    public void testWrongId() throws SQLException
    {
        prepareEmptyResultSet();
        Bank bank = new Bank();
        bank.connect();
        bank.transfer(1, 2, 5000);
        bank.disconnect();
        verifySQLStatementExecuted("select balance");
        verifySQLStatementNotExecuted("update account");
        verifyNotCommitted();
        verifyRolledBack();
        verifyAllResultSetsClosed();
        verifyAllStatementsClosed();
        verifyConnectionClosed();
    }
}
```

```
public class BankTest extends BasicJDBCTestCaseAdapter
{
    private void prepareResultSet()
    {
        MockConnection connection =
            getJDBCMockObjectFactory().getMockConnection();
        StatementResultSetHandler statementHandler =
            connection.getStatementResultSetHandler();
        MockResultSet result = statementHandler.createResultSet();
        result.addRow(new Integer[] {new Integer(10000)});
        statementHandler.prepareGlobalResultSet(result);
    }

    public void testTransferOk() throws SQLException
    {
        prepareResultSet();
        Bank bank = new Bank();
        bank.connect();
        bank.transfer(1, 2, 5000);
        bank.disconnect();
        verifySQLStatementExecuted("select balance");
        verifySQLStatementExecuted("update account");
        verifySQLStatementParameter("update account", 0, 1, new Integer(-5000));
        verifySQLStatementParameter("update account", 0, 2, new Integer(1));
        verifySQLStatementParameter("update account", 1, 1, new Integer(5000));
        verifySQLStatementParameter("update account", 1, 2, new Integer(2));
        verifyCommitted();
        verifyNotRolledBack();
        verifyAllResultSetsClosed();
        verifyAllStatementsClosed();
        verifyConnectionClosed();
    }
}
```

mocking JMS

```
public class MockJmsFixture extends BasicJMSTestCaseAdapter {
  private MockConnection mockConnection;
  private MockSession mockSession;
  private MockTopic mockTopic;
  private TopicSubscriber topicSubscriber;
  private Message message;
```

creating the fixture

```
public MockJmsFixture() throws Exception {
  setUp();
  mockConnection = new MockConnection(getDestinationManager(),
          getConfigurationManager());
  mockSession = new MockSession(mockConnection,
          false, Session.AUTO_ACKNOWLEDGE);
  mockTopic = new MockTopic("ird.OS_ADC_EVTPUB_DEV.event");
  mockTopic.addSession(mockSession);
  topicSubscriber = mockSession.createDurableSubscriber(
          mockTopic, "blah");

}
```

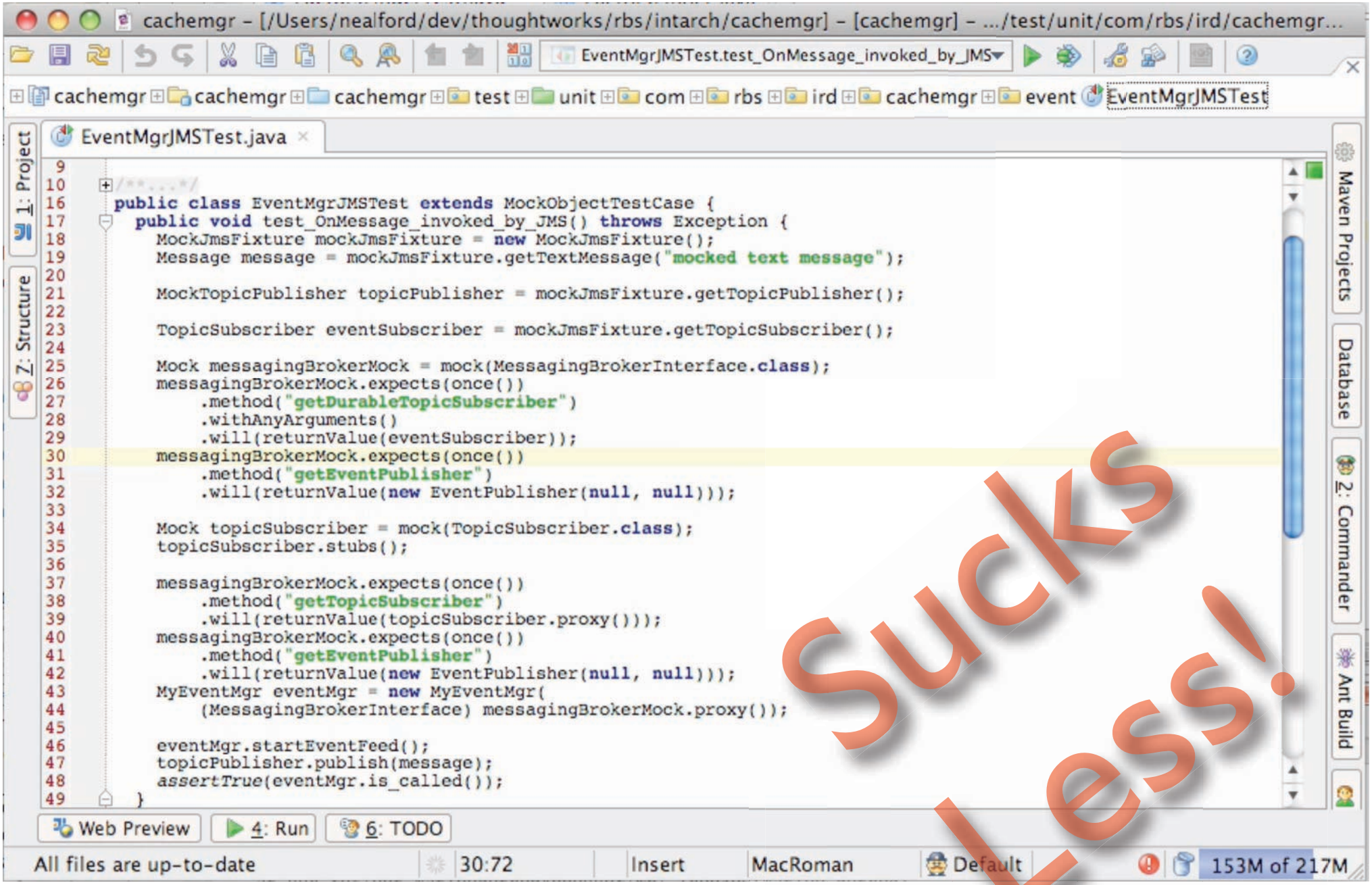
the test

```
public void test_OnMessage_invoked_by_JMS() throws Exception {
  MockJmsFixture mockJmsFixture = new MockJmsFixture();
  Message message = mockJmsFixture.getTextMessage("mocked text message");

  MockTopicPublisher topicPublisher = mockJmsFixture.getTopicPublisher();

  TopicSubscriber eventSubscriber = mockJmsFixture.getTopicSubscriber();

  Mock messagingBrokerMock = mock(MessagingBrokerInterface.class);
  messagingBrokerMock.expects(once())
       .method("getDurableTopicSubscriber")
       .withAnyArguments()
       .will(returnValue(eventSubscriber));
  messagingBrokerMock.expects(once())
       .method("getEventPublisher")
       .will(returnValue(new EventPublisher(null, null)));
```

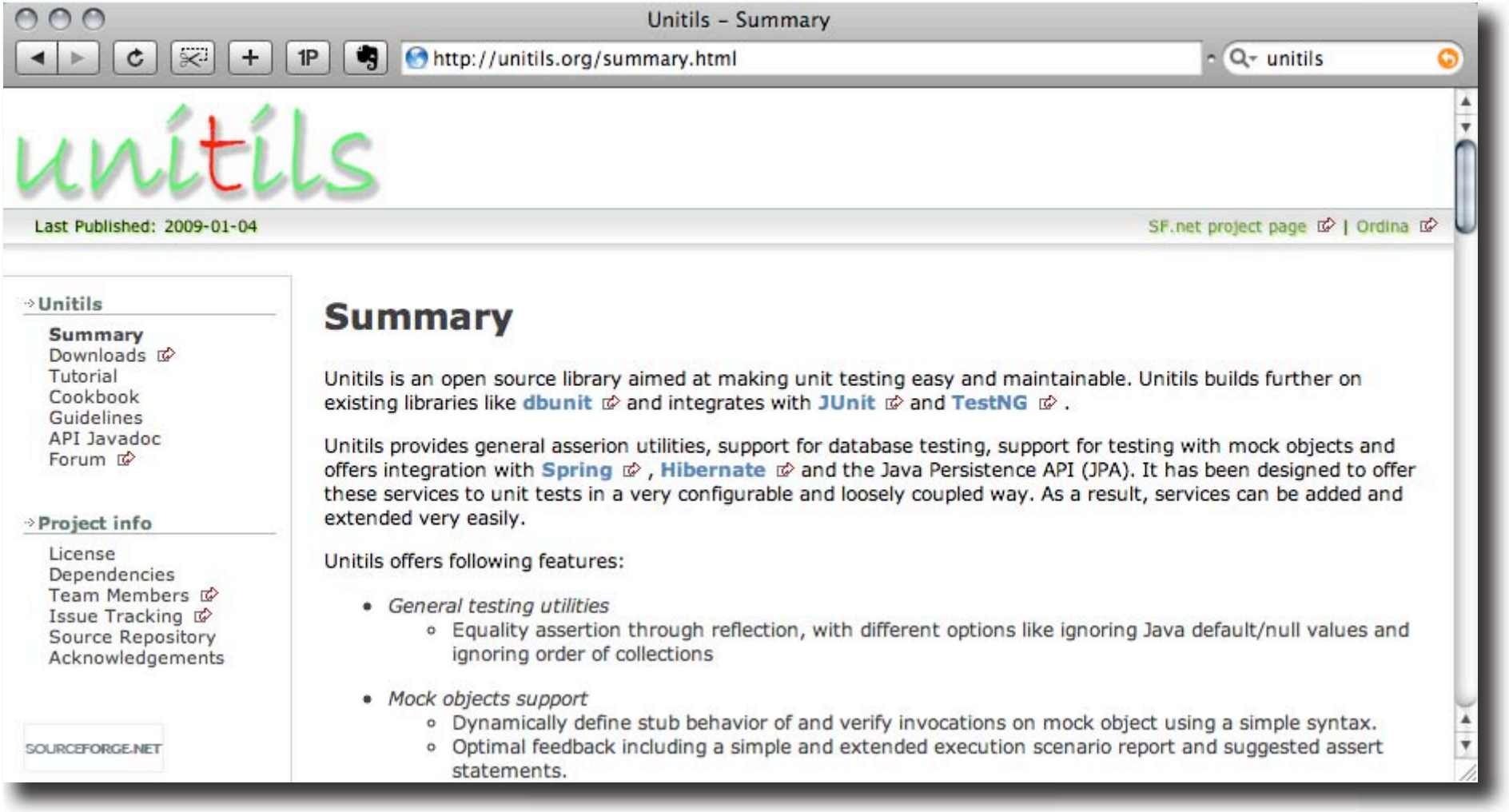
```
    Mock topicSubscriber = mock(TopicSubscriber.class);
    topicSubscriber.stubs();

    messagingBrokerMock.expects(once())
        .method("getTopicSubscriber")
        .will(returnValue(topicSubscriber.proxy()));
    messagingBrokerMock.expects(once())
        .method("getEventPublisher")
        .will(returnValue(new EventPublisher(null, null)));
    MyEventMgr eventMgr = new MyEventMgr(
        (MessagingBrokerInterface) messagingBrokerMock.proxy());

    eventMgr.startEventFeed();
    topicPublisher.publish(message);
    assertTrue(eventMgr.is_called());
}
```

stubbing via inheritance

```
private class MyEventMgr extends EventMgr {
  private boolean _called;

  MyEventMgr(MessagingBrokerInterface messagingBroker) {
    super(messagingBroker);
  }

  @Override
  public void onMessage(Message msg) {
    _called = true;
  }

  public boolean is_called() {
    return _called;
  }
}
```


cachemgr – [/Users/nealford/dev/thoughtworks/rbs/intarch/cachemgr] – [cachemgr] – .../test/unit/com/rbs/ird/cachemgr...
EventMgrJMSTest.test_OnMessage_invoked_by_JMS
cachemgr cachemgr cachemgr test unit com rbs ird cachemgr event EventMgrJMSTest
EventMgrJMSTest.java
1: Project
7: Structure
Maven Projects
Database
2: Commander
Ant Build
/**...*/
public class EventMgrJMSTest extends MockObjectTestCase {
public void test_OnMessage_invoked_by_JMS() throws Exception {
MockJmsFixture mockJmsFixture = new MockJmsFixture();
Message message = mockJmsFixture.getTextMessage("mocked text message");
MockTopicPublisher topicPublisher = mockJmsFixture.getTopicPublisher();
TopicSubscriber eventSubscriber = mockJmsFixture.getTopicSubscriber();
Mock messagingBrokerMock = mock(MessagingBrokerInterface.class);
messagingBrokerMock.expects(once())
.method("getDurableTopicSubscriber")
.withAnyArguments()
.will(returnValue(eventSubscriber));
messagingBrokerMock.expects(once())
.method("getEventPublisher")
.will(returnValue(new EventPublisher(null, null)));
Mock topicSubscriber = mock(TopicSubscriber.class);
topicSubscriber.stubs();
messagingBrokerMock.expects(once())
.method("getTopicSubscriber")
.will(returnValue(topicSubscriber.proxy()));
messagingBrokerMock.expects(once())
.method("getEventPublisher")
.will(returnValue(new EventPublisher(null, null)));
MyEventMgr eventMgr = new MyEventMgr(
(MessagingBrokerInterface) messagingBrokerMock.proxy());
eventMgr.startEventFeed();
topicPublisher.publish(message);
assertTrue(eventMgr.is_called());
}
Web Preview
4: Run
6: TODO
All files are up-to-date
30:72
Insert
MacRoman
Default
153M of 217M
Sucks
Less!

unitils



http://unitils.org/

open source set of utility classes to make typical java scenarios easier to test

offers support to hibernate, spring, JPA

mock objects

persistence layer testing support

spring integration

assertion utilities

```
public class User {

    private long id;
    private String first;
    private String last;

    public User(long id, String first, String last) {
        this.id = id;
        this.first = first;
        this.last = last;
    }
}

User user1 = new User(1, "John", "Doe");
User user2 = new User(1, "John", "Doe");
assertEquals(user1, user2);
```

asserting **user1 == user2**

testing identity

```
public boolean equals(Object object) {
    if (object instanceof User) {
        return id == ((User) object).id;
    }
    return false;
}
```

equals method in User

```
User user1 = new User(1, "John", "Doe");
User user2 = new User(1, "Jane", "Smith");
assertEquals(user1, user2);}
```

what is tested?

```
User user1 = new User(1, "John", "Doe");
User user2 = new User(1, "John", "Doe");
assertEquals(user1.getId(), user2.getId());
assertEquals(user1.getFirst(), user2.getFirst());
assertEquals(user1.getLast(), user2.getLast());
```

more comprehensive

reflection assertions

```
User user1 = new User(1, "John", "Doe");
User user2 = new User(1, "John", "Doe");
assertEquals(user1.getId(), user2.getId());
assertEquals(user1.getFirst(), user2.getFirst());
assertEquals(user1.getLast(), user2.getLast());
```

```
User user1 = new User(1, "John", "Doe");
User user2 = new User(1, "John", "Doe");
assertReflectionEquals(user1, user2);
```

loops over all fields in both objects and compares their values using reflection

lenient assertions

```
List<Integer> myList = Arrays.asList(3, 2, 1);
assertReflectionEquals(Arrays.asList(1, 2, 3), myList, LENIENT_ORDER);
```

```
User actualUser   = new User("John", "Doe",
        new Address("First street", "12", "Brussels"));
User expectedUser = new User("John", null,
        new Address("First street", null, null));
assertReflectionEquals(expectedUser, actualUser, IGNORE_DEFAULTS);
```

```
Date actualDate =   new Date(44444);
Date expectedDate = new Date();
assertReflectionEquals(expectedDate, actualDate, LENIENT_DATES);
```

dbUnit support

dbUnit files to be loaded for this test

```
@DataSet
public class UserDAOTest extends UnitilsJUnit4 {

    @Test
    public void testFindByName() {
        User result = userDao.findByName("doe", "john");
        assertPropertyLenientEquals("userName", "jdoe", result);
    }

    @Test
    public void testFindByMinimalAge() {
        List<User> result = userDao.findByMinimalAge(18);
        assertPropertyLenientEquals("firstName", Arrays.asList("jack"), result);
    }
}
```

dbUnit support

```xml
<?xml version='1.0' encoding='UTF-8'?>
<dataset>

    <usergroup name="admin" />
    <user userName="jdoe"  name="doe"   firstname="john"   userGroup="admin" />

    <usergroup name="sales" />
    <user userName="smith" name="smith" userGroup="sales" />

</dataset>
```

this data will be loaded prior to test run

hibernate support

```
@HibernateSessionFactory("hibernate.cfg.xml")
public class BaseDaoTest extends UnitilsJUnit4 {
}

public class UserDaoTest extends BaseDaoTest {

    @HibernateSessionFactory
    private SessionFactory sessionFactory;
}


@HibernateSessionFactory("hibernate.cfg.xml")
public class HibernateMappingTest extends UnitilsJUnit4 {

    @Test
    public void testMappingToDatabase() {
        HibernateUnitils.assertMappingWithDatabaseConsistent();
    }
}
```

spring support

sometimes useful to have spring around during testing

management of ApplicationContext configuration

injection of Spring beans in unit tests

make use of a hibernate SessionFactory configured in Spring

reference the Unitils DataSource in Spring configuration

spring support

```
public class UserServiceTest extends UnitilsJUnit4 {

    @SpringApplicationContext({"spring-config.xml", "spring-test-config.xml"})
    private ApplicationContext applicationContext;

}
```

ApplicationContext

```
@SpringBean("userService")
private UserService userService;

@SpringBeanByName
private UserService userService;

@SpringBeanByType
private UserService userService;
```

injection

```java
public class AlertServiceTest extends UnitilsJUnit4 {
    AlertService alertService;
    Message alert1, alert2;
    List<Message> alerts;
    Mock<SchedulerService> mockSchedulerService;
    Mock<MessageService> mockMessageService;

    @Before
    public void init() {
        alertService = new AlertService(
                mockSchedulerService.getMock(), mockMessageService.getMock());
        alert1 = new Alert(...); alert2 = new Alert(...);
        alerts = Arrays.asList(alert1, alert2);
    }

    @Test
    public void testSendScheduledAlerts() {
        mockSchedulerService.returns(alerts).getScheduledAlerts(null));
        alertService.sendScheduledAlerts();

        mockMessageService.assertInvoked().sendMessage(alert1);
        mockMessageService.assertInvoked().sendMessage(alert2);
    }
}
```

auto creation of mocks

expectations

verification

Sucks less!

spock – Google Code

neal.ford@gmail.com | My favorites | Profile | Sign out

spock

all your bugs are belong to us

Search projects

Project Home | Downloads | Wiki | Issues | Source

Summary | Updates

Spock is a testing and specification framework for Java and Groovy developers. What makes it stand out from the crowd is its beautiful and highly expressive specification language. Thanks to its JUnit runner, Spock is compatible with most IDEs, build tools, and continuous integration servers. Spock is inspired from JUnit, jMock, RSpec, Groovy, Scala, Vulcans, and other fascinating life forms.

```
@Speck
class HelloSpock {
    def "can you figure out what I'm up to?"() {
        expect:
        name.size() == size

        where:
        name << ["Kirk", "Spock", "Scotty"]
        size << [4, 5, 6]
    }
}
```

Read ten reasons why Spock is for you, see more examples, learn how to write a

Star this project

Code license: Apache License 2.0

Labels: Test, Specification, Unit, Integration, TDD, BDD, Java, Groovy, Spock, Framework

Featured downloads: Show all

spock-core-0.1.jar

Featured wiki pages: Show all

Examples

FAQ

GettingStarted

http://code.google.com/p/spock/

spock

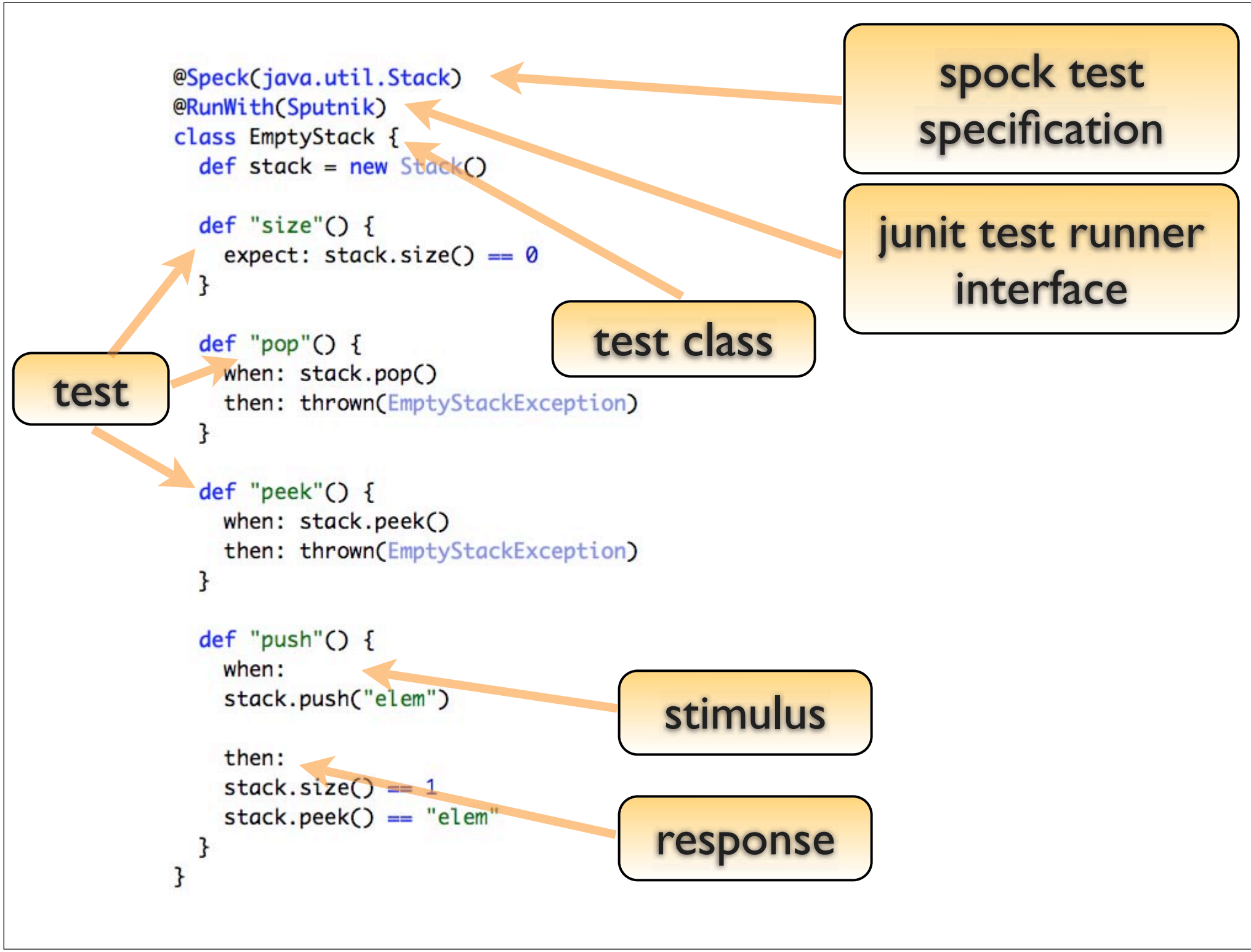
testing & specification framework for Java & Groovy

highly expressive specification language

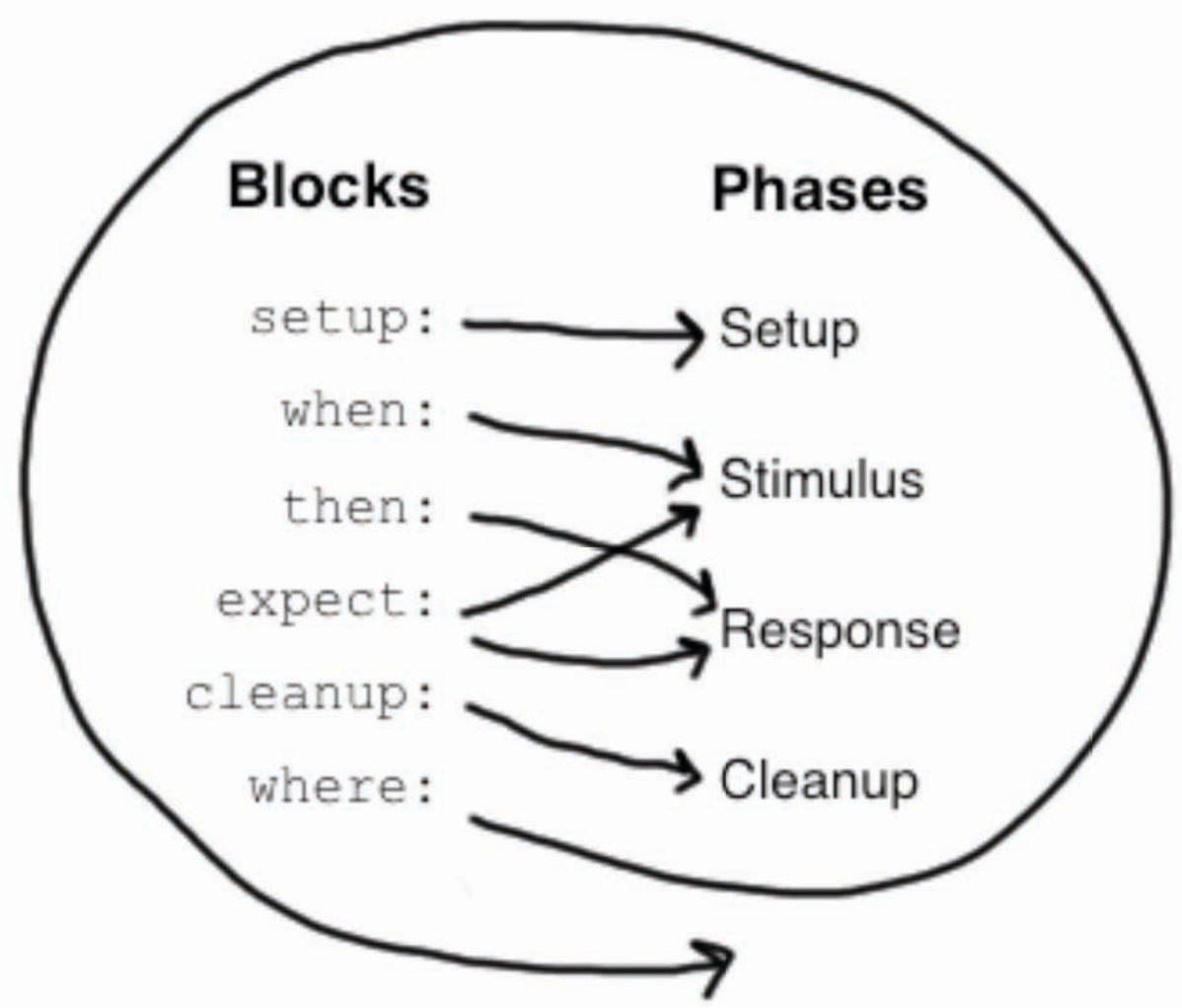
nicely detailed error messages

```
setup - stimulus - response - cleanup
```

compatible with junit test runner


@Speck(java.util.Stack)
@RunWith(Sputnik)
class EmptyStack {
def stack = new Stack()
def "size"() {
expect: stack.size() == 0
}
def "pop"() {
when: stack.pop()
then: thrown(EmptyStackException)
}
def "peek"() {
when: stack.peek()
then: thrown(EmptyStackException)
}
def "push"() {
when:
stack.push("elem")
then:
stack.size() == 1
stack.peek() == "elem"
}
}
spock test specification
junit test runner interface
test class
test
stimulus
response

blocks & outcomes

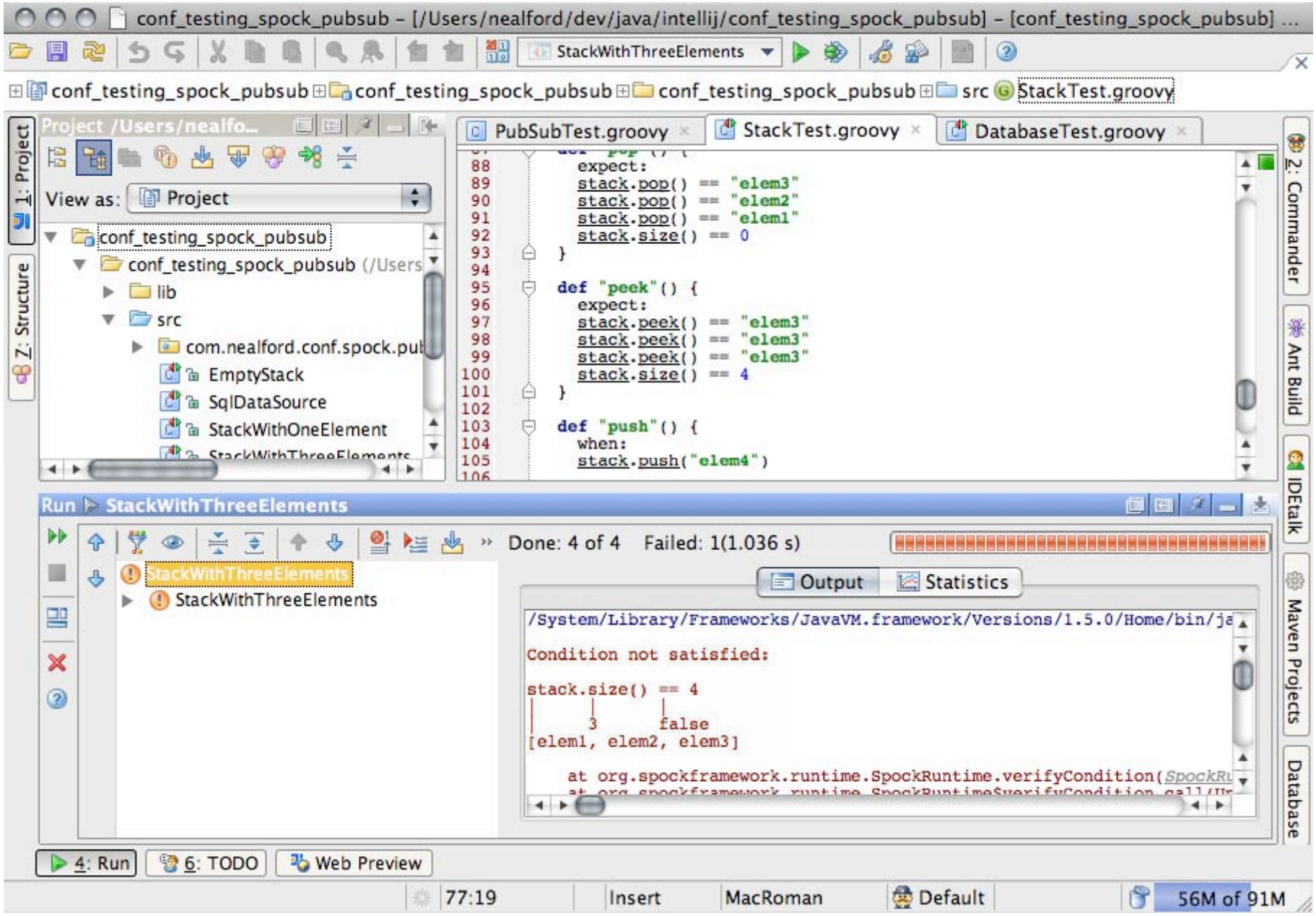


```groovy
@Speck(java.util.Stack)
@RunWith(Sputnik)
class StackWithOneElement {
  def stack = new Stack()

  def setup() {
    stack.push("elem")
  }

  def "size"() {
    expect: stack.size() == 1
  }

  def "pop"() {
    when:
    def x = stack.pop()

    then:
    x == "elem"
    stack.size() == 0
  }

  def "peek"() {
    when:
    def x = stack.peek()

    then:
    x == "elem"
    stack.size() == 1
  }
}
```

setup

simple statements of fact

pretty failures!



junit vs. spock

Spock	JUnit
Specification	Test class
`setup()`	`@Before`
`cleanup()`	`@After`
`setupSpeck()`	`@BeforeClass`
`cleanupSpeck()`	`@AfterClass`
Feature	Test
Parameterized feature	Theory
Condition	Assertion
Exception condition	`@Test(expected=...)`
`@FailsWith`	`@Test(expected=...)`
Interaction	Mock expectation (EasyMock, JMock, ...)

tdd private methods?

```
private void calculateFactors() {
    for (int i = 2; i < sqrt(_number) + 1; i++)
        if (isFactor(i))
            addFactor(i);
}

private void addFactor(int factor) {
    _factors.add(factor);
    _factors.add(_number / factor);
}

private int sumOfFactors() {
    int sum = 0;
    for (int i : _factors)
        sum += i;
    return sum;
}
```

solution #1:

make them all public
(or package) scope

solution #2:

use reflection

reflection helpers

```
@Test
public void is_1_a_factor_of_10() {
    Classifier6 c = new Classifier6(10);
    assertTrue(isFactor(c, 1));
}

private boolean isFactor(Classifier6 c, int factor) {
    try {
        Method m = Classifier6.class.getDeclaredMethod(
                "isFactor", int.class);
        m.setAccessible(true);
        return (Boolean) m.invoke(c, factor);
    } catch (Throwable t) {
        fail();
    }
    return false;
}
```


groovy

reflection

```
@Test public void is_factor_via_reflection() {
  def m = Classifier6.class.getDeclaredMethod("isFactor", int.class)
  m.accessible = true
  assertTrue m.invoke(new Classifier6(10), 10)
  assertTrue m.invoke(new Classifier6(25), 5)
  assertFalse m.invoke(new Classifier6(25), 6)
}
```

dirty secret: private is ignored!

```
@Test public void is_factor() {
    assertTrue new Classifier6(10).isFactor(1)
    assertTrue new Classifier6(25).isFactor(5)
    assertFalse new Classifier6(25).isFactor(6)
}
```

technically, a bug...

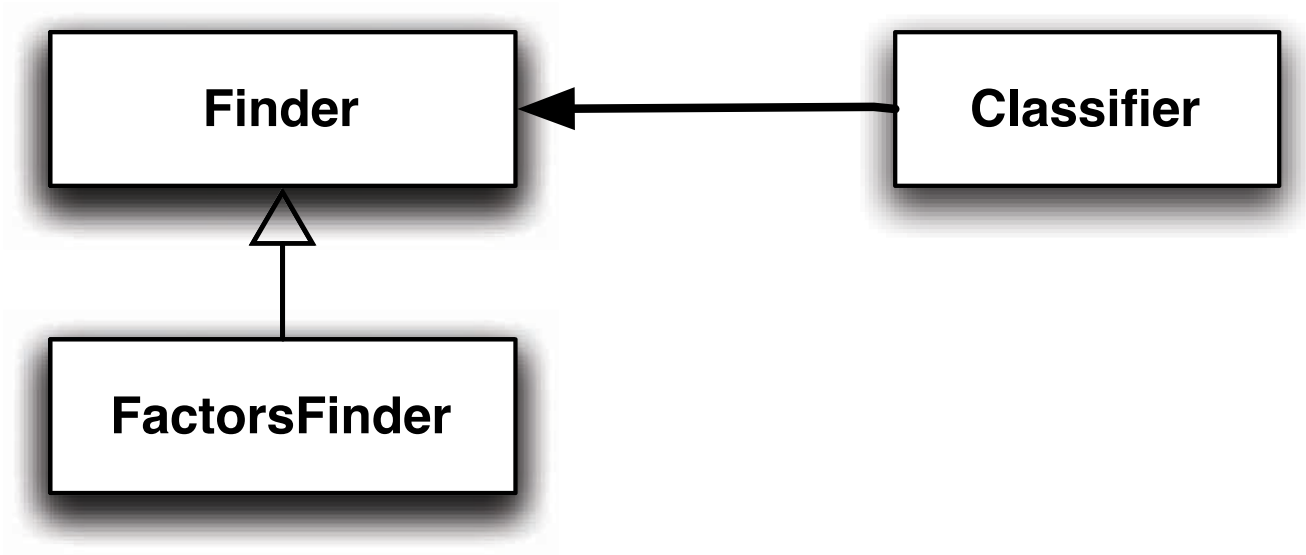
...no great hurry to fix it (insanely useful!)


mocking


mocking in
JAVA™

changes:

`FactorsFinder` class harvests factors



`Classifier` uses external factors

```
public class Classifier7 {
    private int _number;
    private Finder _factors;

    public Classifier7(int number, Finder factors) {
        _number = number;
        _factors = factors;
    }

    public boolean isPerfect() {
        return sumOfFactors() - _number == _number;
    }

    public int sumOfFactors() {
        int sum = 0;
        for (int i : _factors.factors())
            sum += i;
        return sum;
    }
}
```

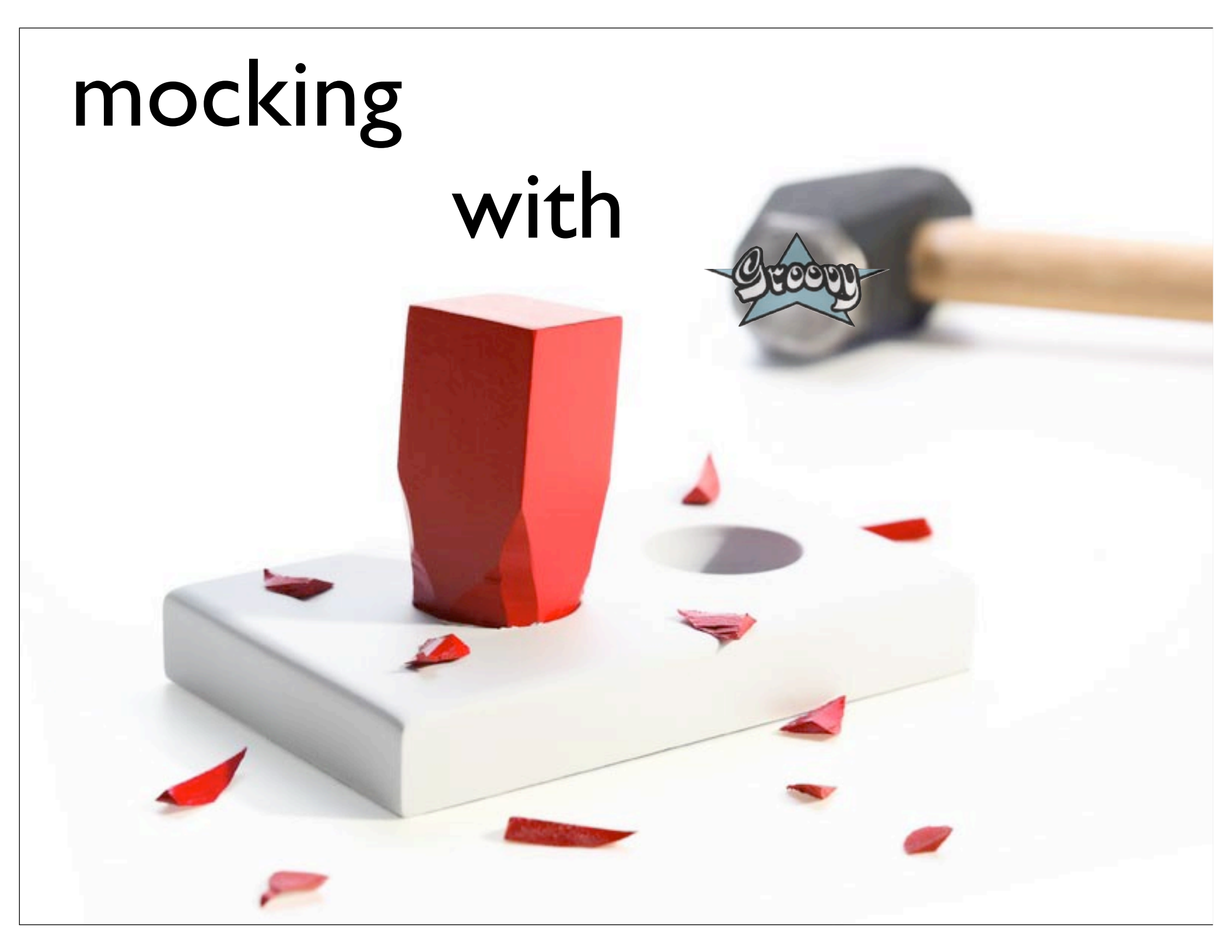
```
public class FactorsFinder implements Finder {
    private int _number;
    private Set<Integer> _factors = new HashSet<Integer>();

    public FactorsFinder(int number) {
        _number = number;
    }

    public Set<Integer> factors() {
        calculateFactors();
        return _factors;
    }

    private boolean isFactor(int factor) {
        return _number % factor == 0;
    }

    public void calculateFactors() {
        for (int i = 2; i < sqrt(_number) + 1; i++)
            if (isFactor(i))
                addFactor(i);
    }

    private void addFactor(int factor) {
        _factors.add(factor);
        _factors.add(_number / factor);
    }
}
```

jmock


```
@RunWith(JMock.class)
public class ClassifierWithMockTest {
    Mockery context = new JUnit4Mockery() {{
        setImposteriser(ClassImposteriser.INSTANCE);
    }};

    @Test public void external_factors() {
        final Finder facts = context.mock(Finder.class);
        Classifier7 c = new Classifier7(42, facts);
        final Set<Integer> expected =
                new HashSet(Arrays.asList(1, 2, 3, 6, 7, 21, 14, 42));
        context.checking(new Expectations() {{
            one(facts).factors(); will(returnValue(expected));
        }});
        assertThat(c.sumOfFactors(), is(1 + 2 + 3 + 6 + 7 + 21 + 14 + 42));
        context.assertIsSatisfied();
    }
}
```

Sucks!

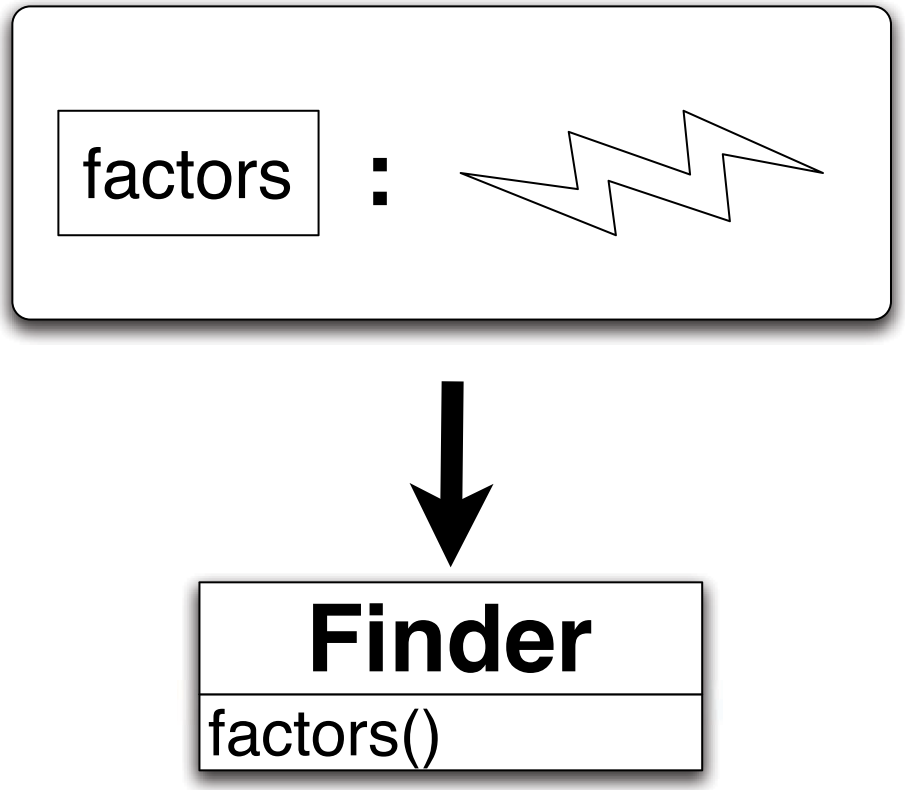
mocking with



java-like syntax

```
@Test public void test_mock_with_simple_syntax() {
  Set expected = new HashSet(Arrays.asList(1, 2, 3, 6, 7, 21, 14, 42))
  def facts = [factors: {expected}] as Finder
  Classifier7 c = new Classifier7(42, facts)
  def sumOfFactors = 0
  expected.each { num ->
    sumOfFactors += num
  }
  assertThat c.sumOfFactors(), is(sumOfFactors)
}
```

```
def facts = [factors: {expected}] as Finder
```

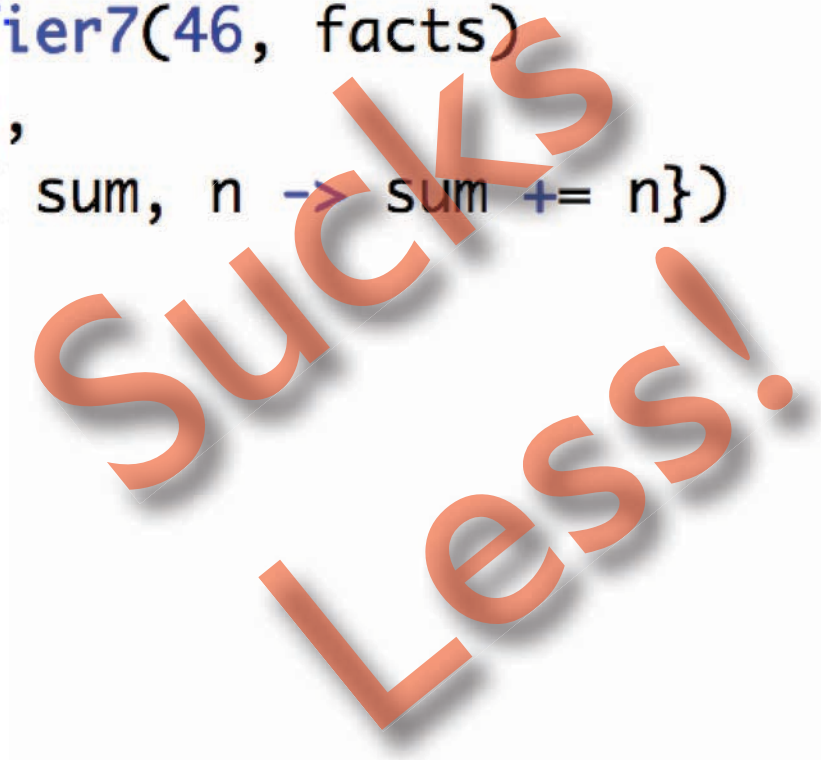


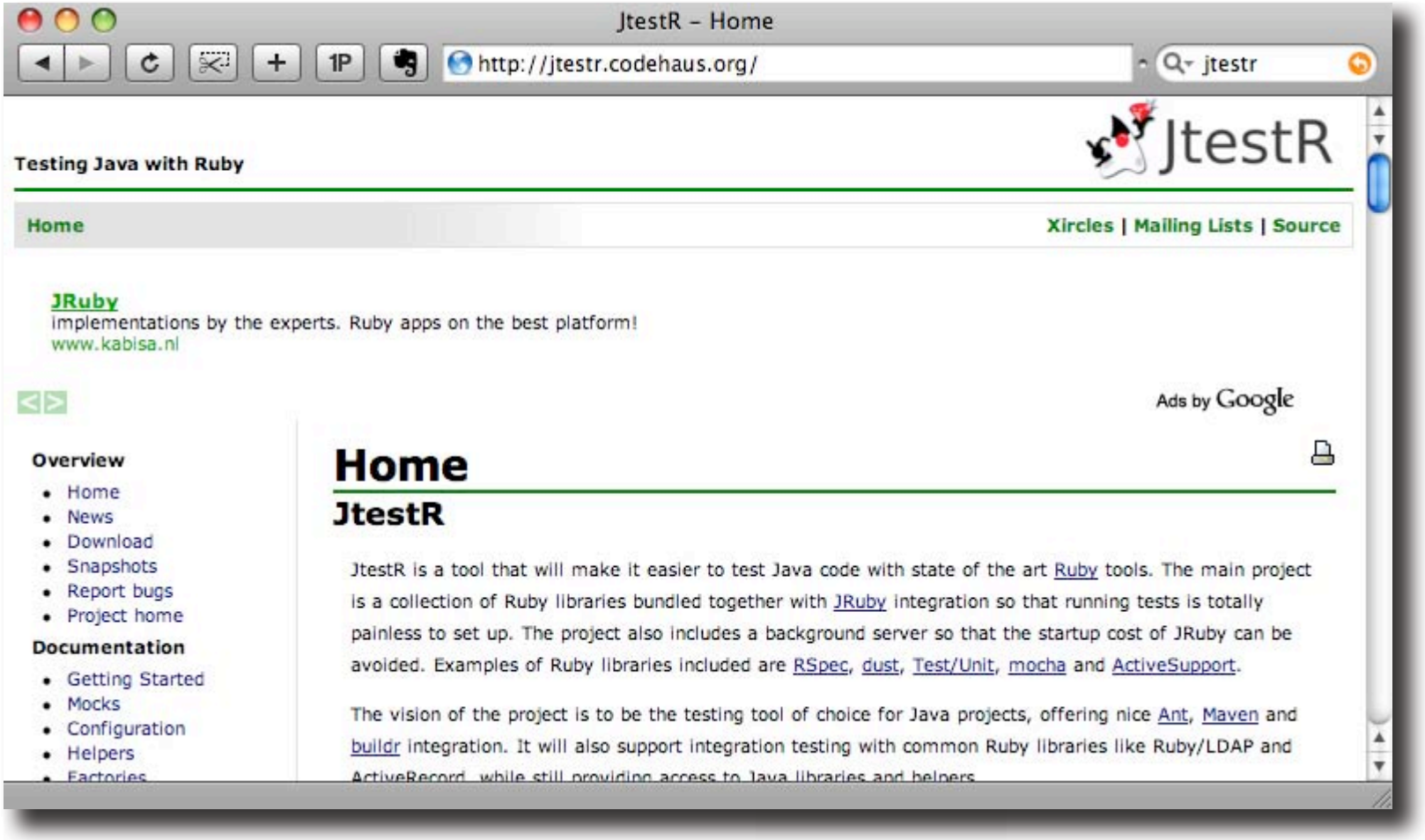
idiomatic


```
@Test public void test_mock_with_simple_syntax() {
  Set expected = new HashSet(Arrays.asList(1, 2, 3, 6, 7, 21, 14, 42))
  def facts = [factors: {expected}] as Finder
  Classifier7 c = new Classifier7(42, facts)
  def sumOfFactors = 0
  expected.each { num ->
    sumOfFactors += num
  }
  assertThat c.sumOfFactors(), is(sumOfFactors)
}
```

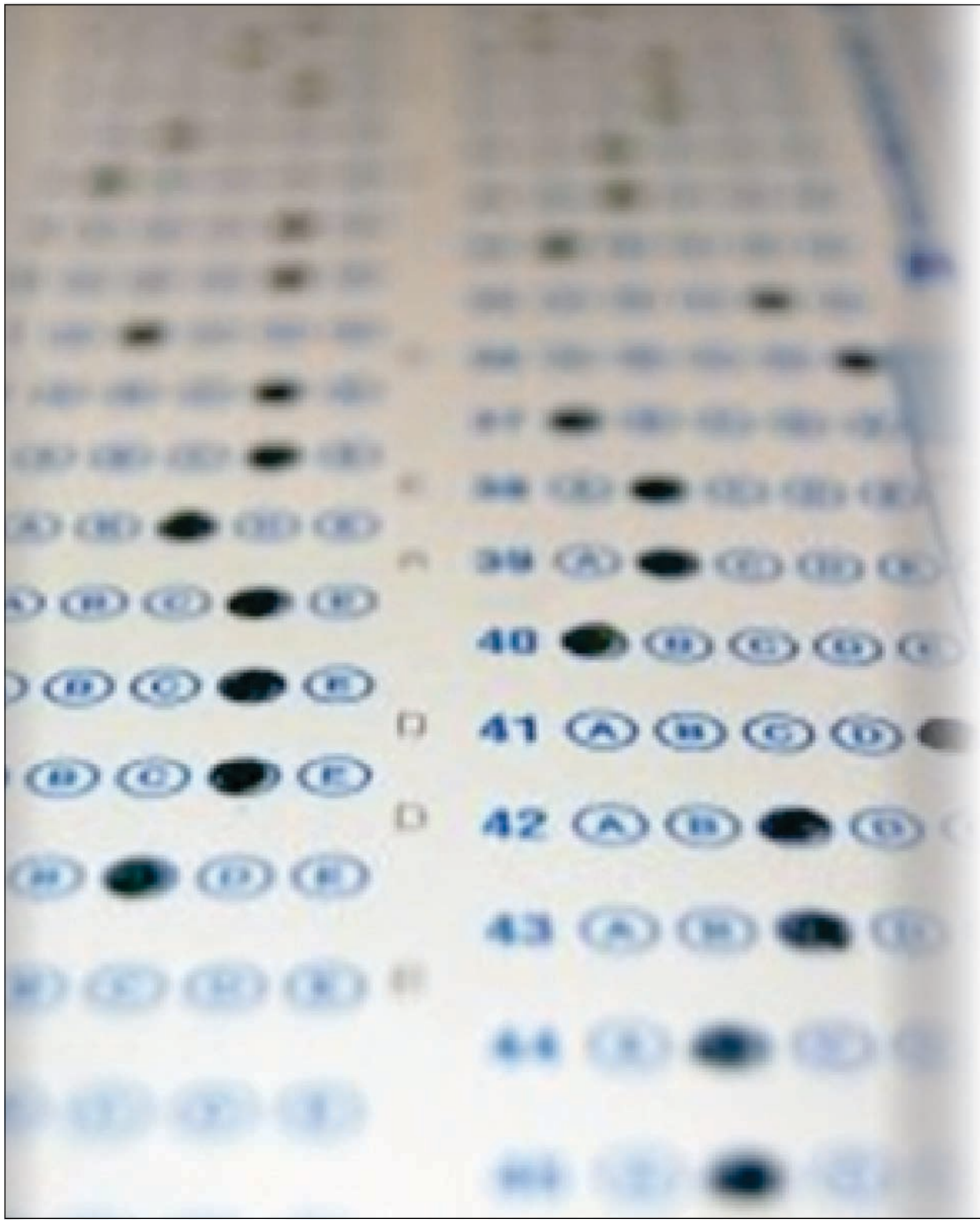
```
@Test public void test_mock_with_factors_finder() {
  def expected = [1, 2, 3, 6, 7, 21, 14, 42]
  def facts = [factors: {expected as Set}] as Finder
  Classifier7 c = new Classifier7ssifier7(42, facts)
  assertThat c.sumOfFactors(),
      is(expected.inject(0) { sum, n -> sum += n})
}
```

```
@Test public void test_mock_with_factors_finder() {
   def expected = [1, 2, 6, 7, 23, 46]
   def facts = [factors: {expected as Set}] as Finder
   Classifier7 c = new Classifier7(46, facts)
   assertThat c.sumOfFactors(),
       is(expected.inject(0) { sum, n -> sum += n})
}
```





http://jtestr.codehaus.org/



testing java with jruby

the java part

```
public interface Order {
    void fill(Warehouse warehouse);

    boolean isFilled();
}
public interface Warehouse {
    public void add(String item, int quantity);

    int getInventory(String product);

    boolean hasInventory(String product, int quantity);

    void remove(String product, int quantity);
}
```

testing `fill()`

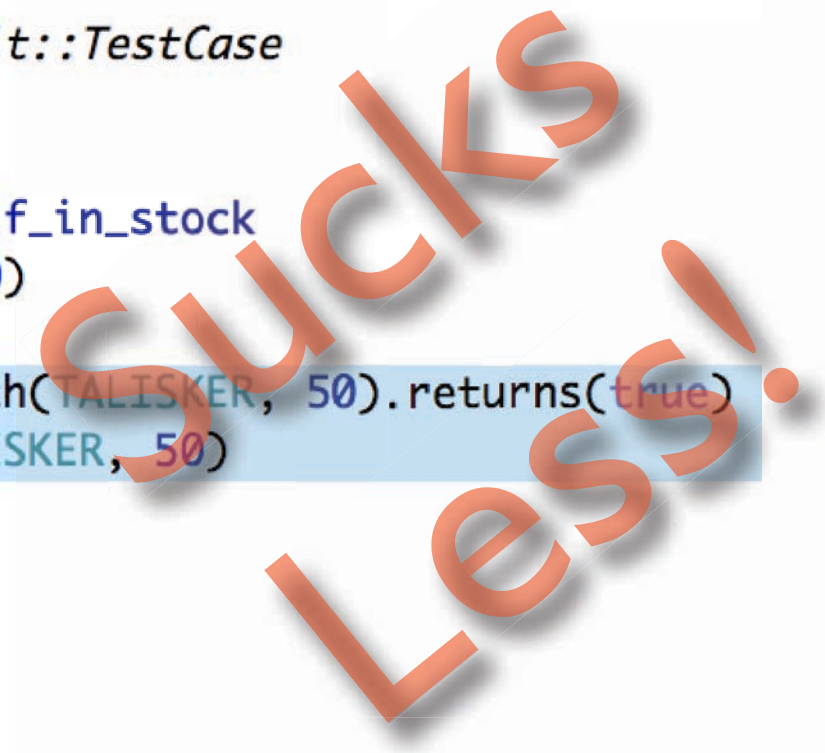
```
public void fill(Warehouse warehouse) {
    if (warehouse.hasInventory(_product, _quantity)) {
        warehouse.remove(_product, _quantity);
        _filled = true;
    } else
        _filled = false;

}
```

jmock

```
@RunWith(JMock.class)
public class OrderInteractionTester {
    private static String TALISKER = "Talisker";
    Mockery context = new JUnit4Mockery();

    @Test public void fillingRemovesInventoryIfInStock() {
        Order order = new OrderImpl(TALISKER, 50);
        final Warehouse warehouse = context.mock(Warehouse.class);

        context.checking(new Expectations() {{
            one (warehouse).hasInventory(TALISKER, 50); will(returnValue(true));
            one (warehouse).remove(TALISKER, 50);
        }});

        order.fill(warehouse);
        assertThat(order.isFilled(), is(true));
        context.assertIsSatisfied();
    }
```

mocha

```
require "java"
require "Warehouse.jar"
%w(OrderImpl Order Warehouse WarehouseImpl).each { |f|
   include_class "com.nealford.conf.jmock.warehouse.#{f}"
}

class OrderInteractionTest < Test::Unit::TestCase
  TALISKER = "Talisker"

  def test_filling_removes_inventory_if_in_stock
    order = OrderImpl.new(TALISKER, 50)
    warehouse = Warehouse.new
    warehouse.stubs(:hasInventory).with(TALISKER, 50).returns(true)
    warehouse.stubs(:remove).with(TALISKER, 50)

    order.fill(warehouse)
    assert order.is_filled
  end
```



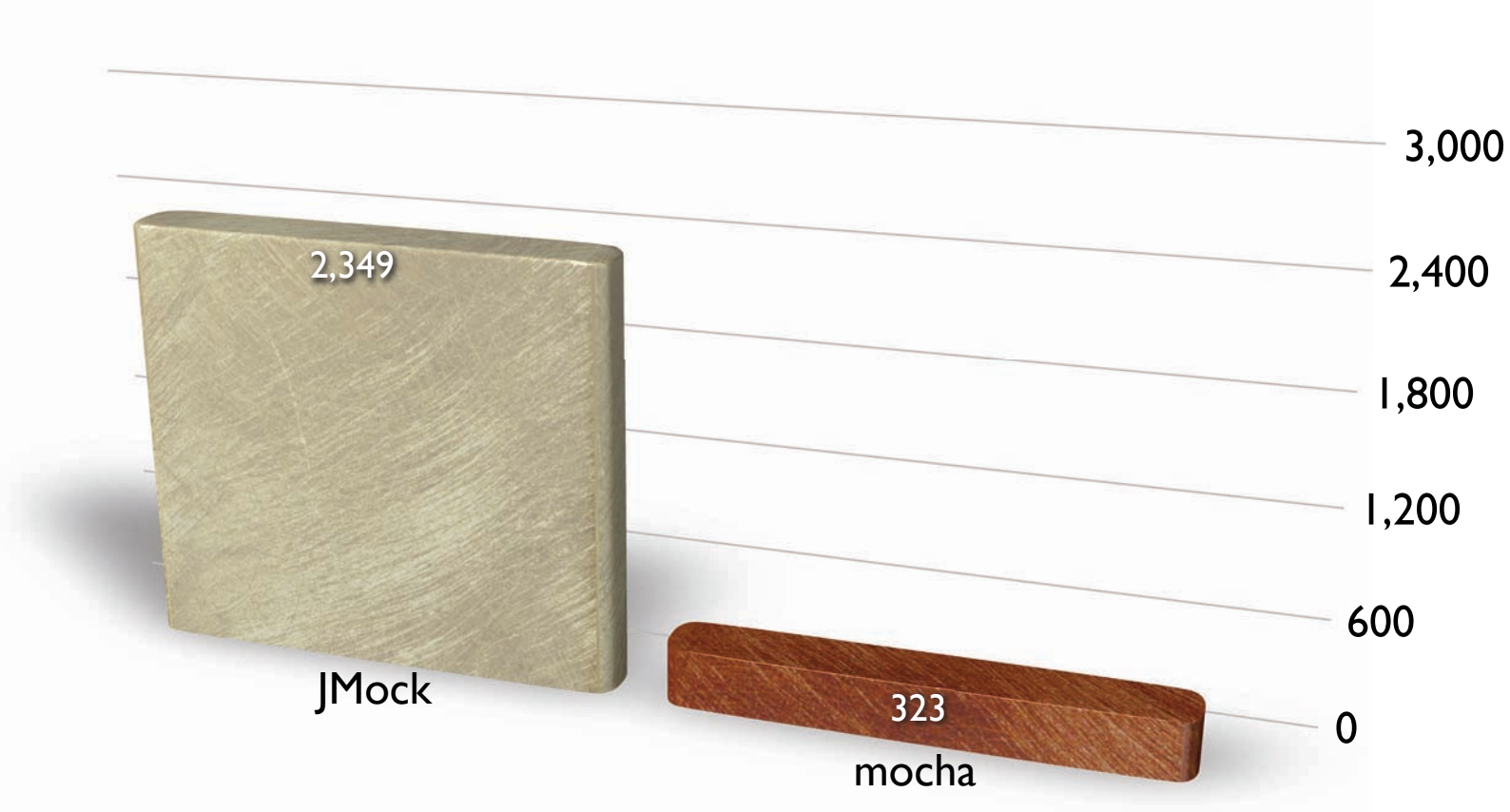
what does it take???

```
class Object

  def expects(symbol)
    method = stubba_method.new(stubba_object, symbol)
    $stubba.stub(method)
    mocha.expects(symbol, caller)
  end

  def stubs(symbol)
    method = stubba_method.new(stubba_object, symbol)
    $stubba.stub(method)
    mocha.stubs(symbol, caller)
  end

  def verify
    mocha.verify
  end

end
```

jmock vs mocha loc

jmock has 7.5 times as many lines of code

jmock vs mocha cc



jmock has 7.2 times the complexity of mocha

rspec

behavior driven development framework

allows you to create readable specifications...

...that execute

inspired by JBehave

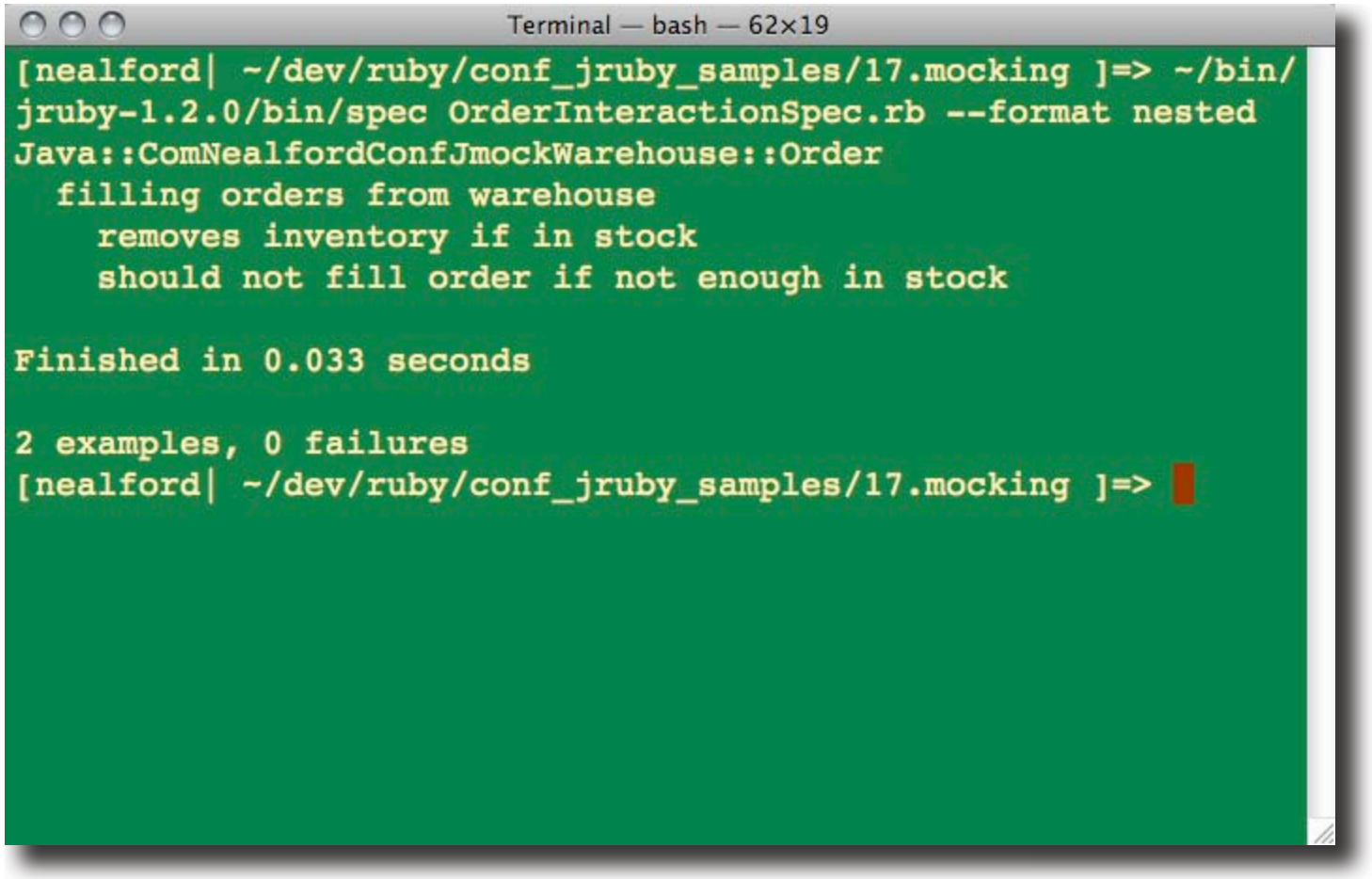
inspiration for tons of similar frameworks

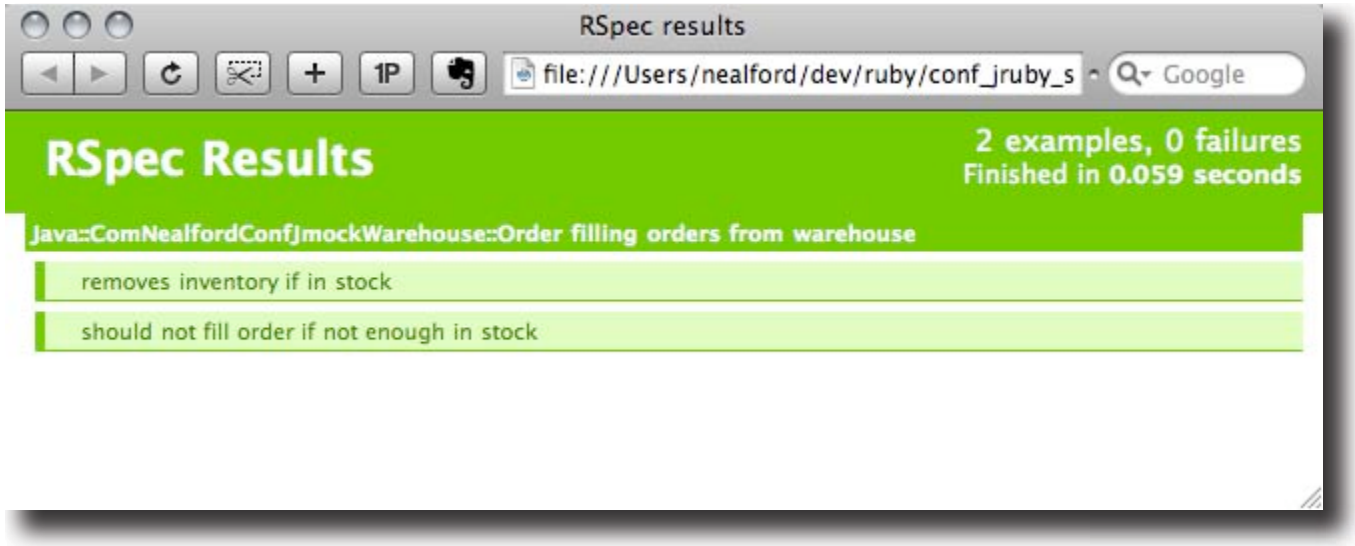
specification

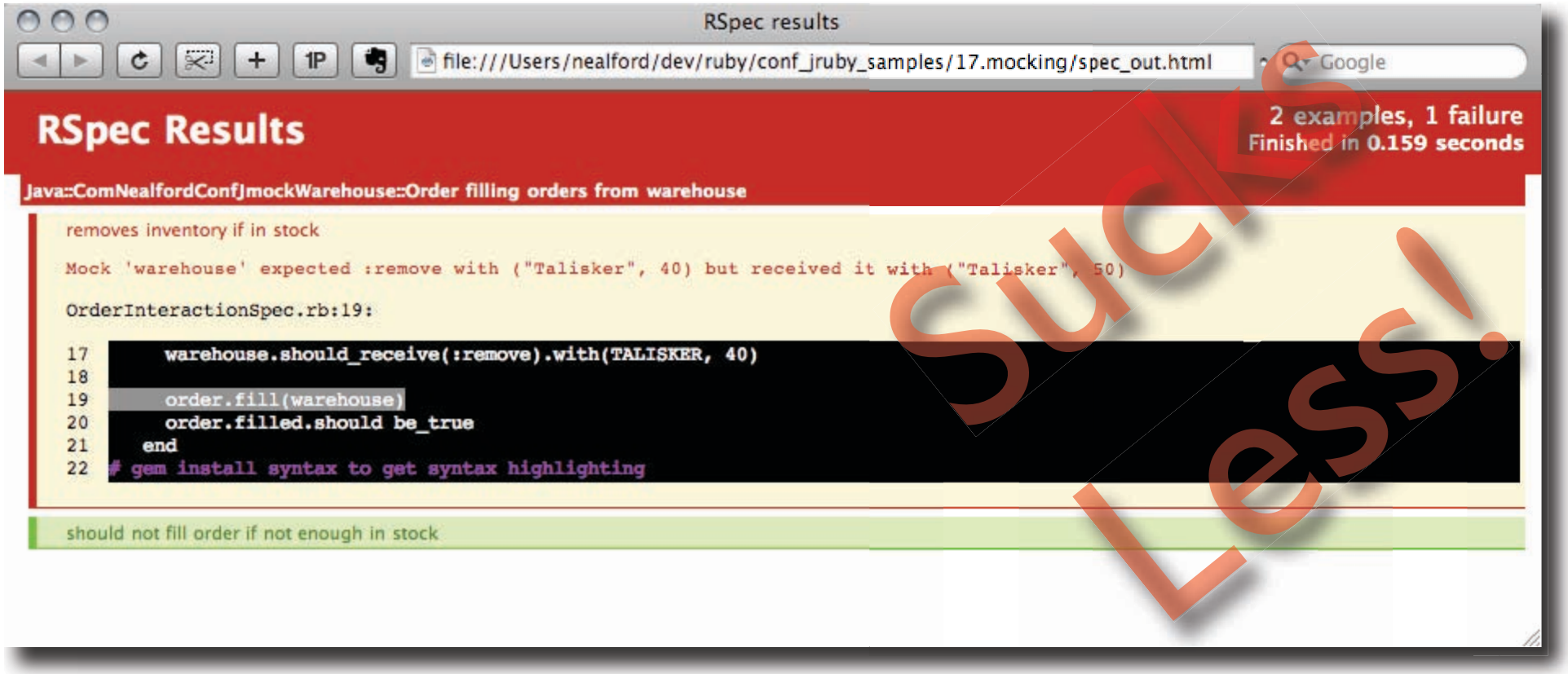
```
describe Order do
 context "filling orders from warehouse" do
   it "removes inventory if in stock" do
     order = OrderImpl.new(TALISKER, 50)
     warehouse = mock("warehouse")
     warehouse.should_receive(:hasInventory).
       with(TALISKER, 50).and_return(true)
     warehouse.should_receive(:remove).with(TALISKER, 50)

     order.fill(warehouse)
     order.filled.should be_true
   end
 end
end
```

```
describe Order do
 context "filling orders from warehouse" do
   it "removes inventory if in stock" do
     order = OrderImpl.new(TALISKER, 50)
     warehouse = mock("warehouse")
     warehouse.should_receive(:hasInventory).
       with(TALISKER, 50).and_return(true)
     warehouse.should_receive(:remove).with(TALISKER, 50)

     order.fill(warehouse)
     order.filled.should be_true
   end

   it "should not fill order if not enough in stock" do
     order = OrderImpl.new(TALISKER, 50)
     warehouse = mock("warehouse")
     warehouse.should_receive(:hasInventory).
       with(TALISKER, 50).and_return(false)

     order.fill(warehouse)
     order.filled.should be_false
   end
 end
end
```

pretty results




RSpec results
file:///Users/nealford/dev/ruby/conf_jruby_s
Google
RSpec Results
2 examples, 0 failures
Finished in 0.059 seconds
Java::ComNealfordConfJmockWarehouse::Order filling orders from warehouse
removes inventory if in stock
should not fill order if not enough in stock


RSpec results
file:///Users/nealford/dev/ruby/conf_jruby_samples/17.mocking/spec_out.html
Google
RSpec Results
2 examples, 1 failure
Finished in 0.159 seconds
Java::ComNealfordConfJmockWarehouse::Order filling orders from warehouse
removes inventory if in stock
Mock 'warehouse' expected :remove with ("Talisker", 40) but received it with ("Talisker", 50)
OrderInteractionSpec.rb:19:
17 warehouse.should_receive(:remove).with(TALISKER, 40)
18
19 order.fill(warehouse)
20 order.filled.should be_true
21 end
22 # gem install syntax to get syntax highlighting
should not fill order if not enough in stock
Sucks Less!

? 's

please fill out the session evaluations

samples at `github.com/nealford`

NEAL FORD software architect / meme wrangler

ThoughtWorks

nford@thoughtworks.com
3003 Summit Boulevard, Atlanta, GA 30319
www.nealford.com
www.thoughtworks.com
blog: memeagora.blogspot.com
twitter: neal4d