ThoughtWorks®

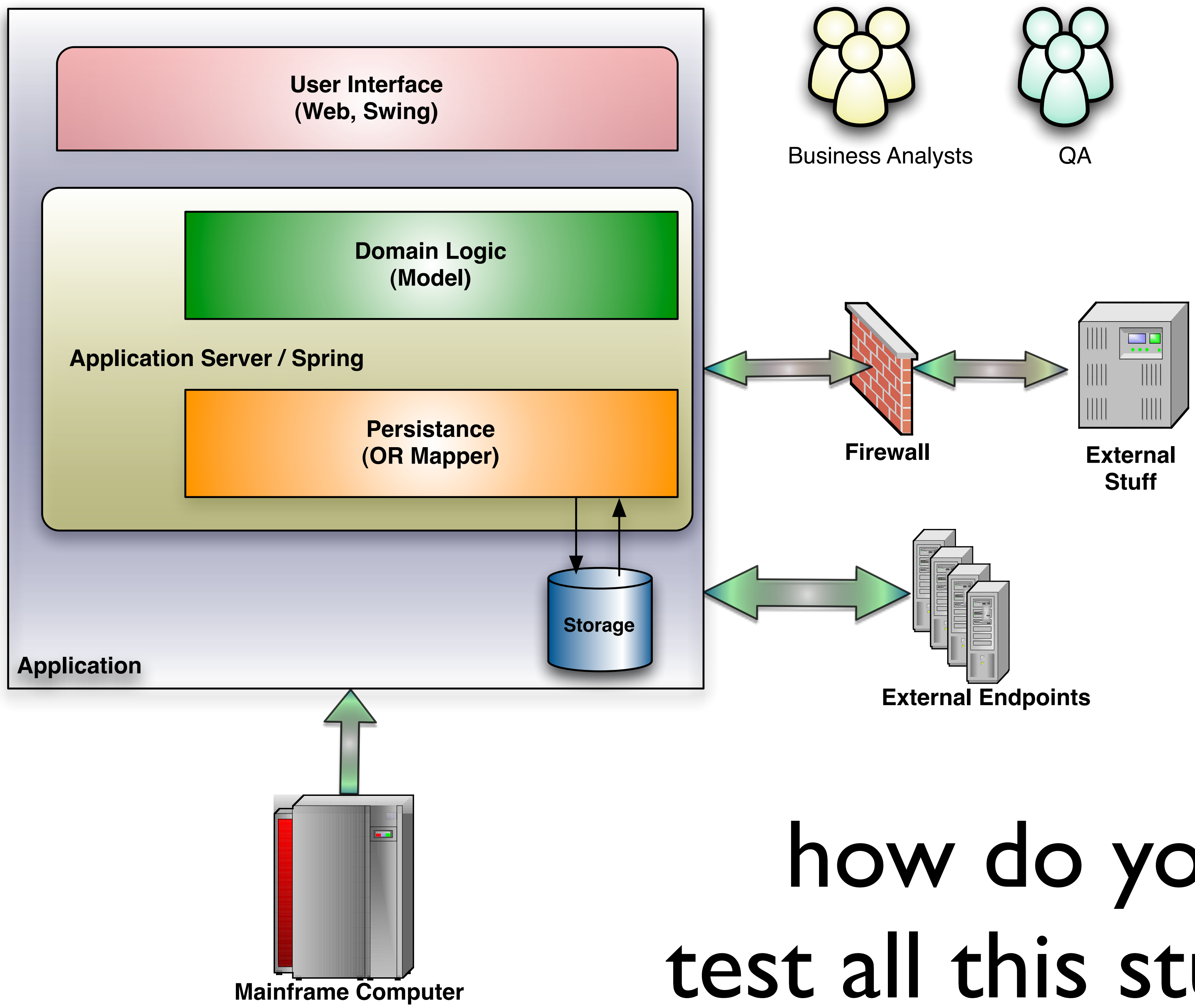
testing the entire stack

NEAL FORD software architect / meme wrangler

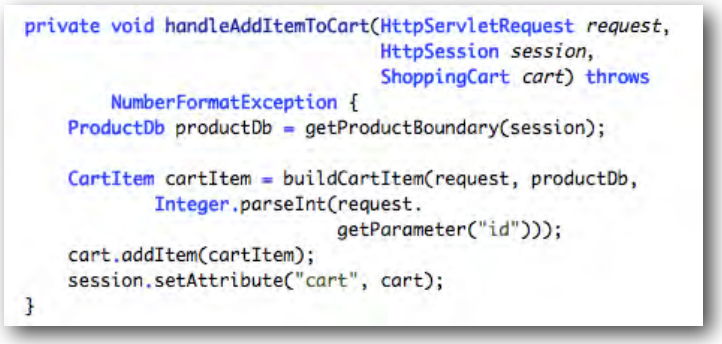
ThoughtWorks®

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

NF



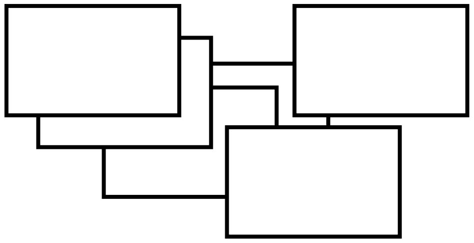
how do you
test all this stuff?!?



```
private void handleAddItemToCart(HttpServletRequest request,
                                  HttpSession session,
                                  ShoppingCart cart) throws
        NumberFormatException {
    ProductDb productDb = getProductBoundary(session);

    CartItem cartItem = buildCartItem(request, productDb,
            Integer.parseInt(request.
                    getParameter("id")));
    cart.addItem(cartItem);
    session.setAttribute("cart", cart);
}
```

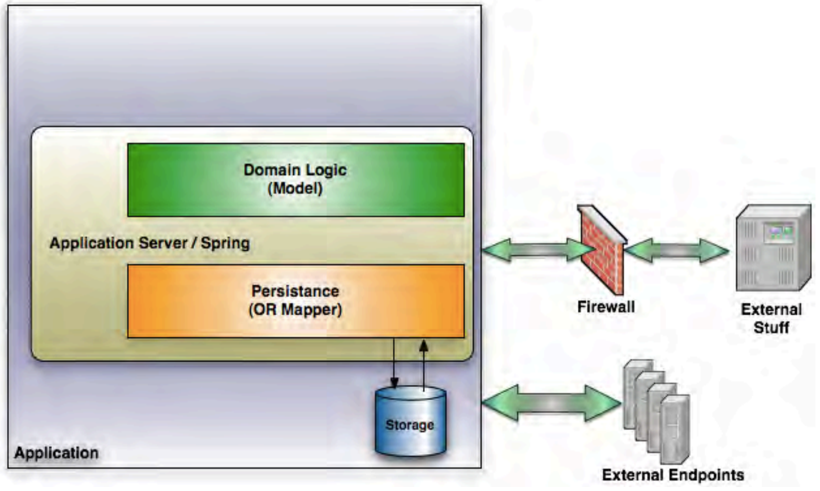
unit

single methods



functional

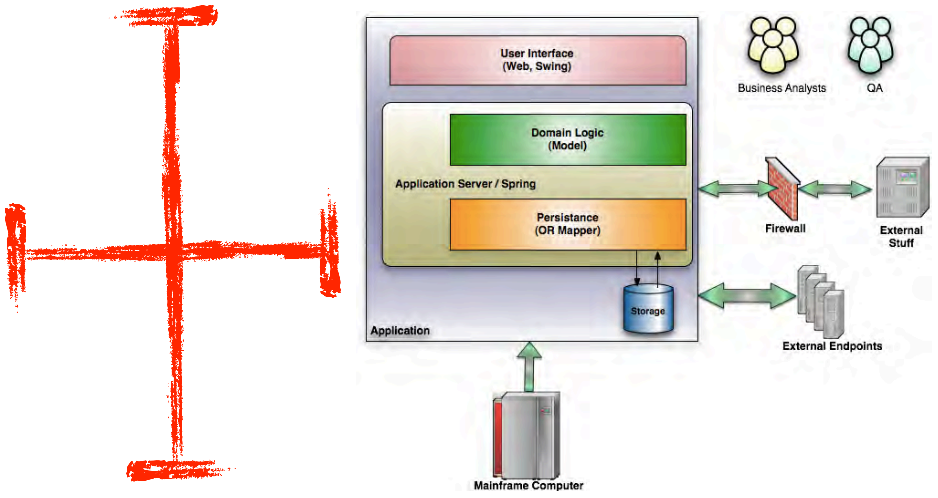
multiple methods, classes



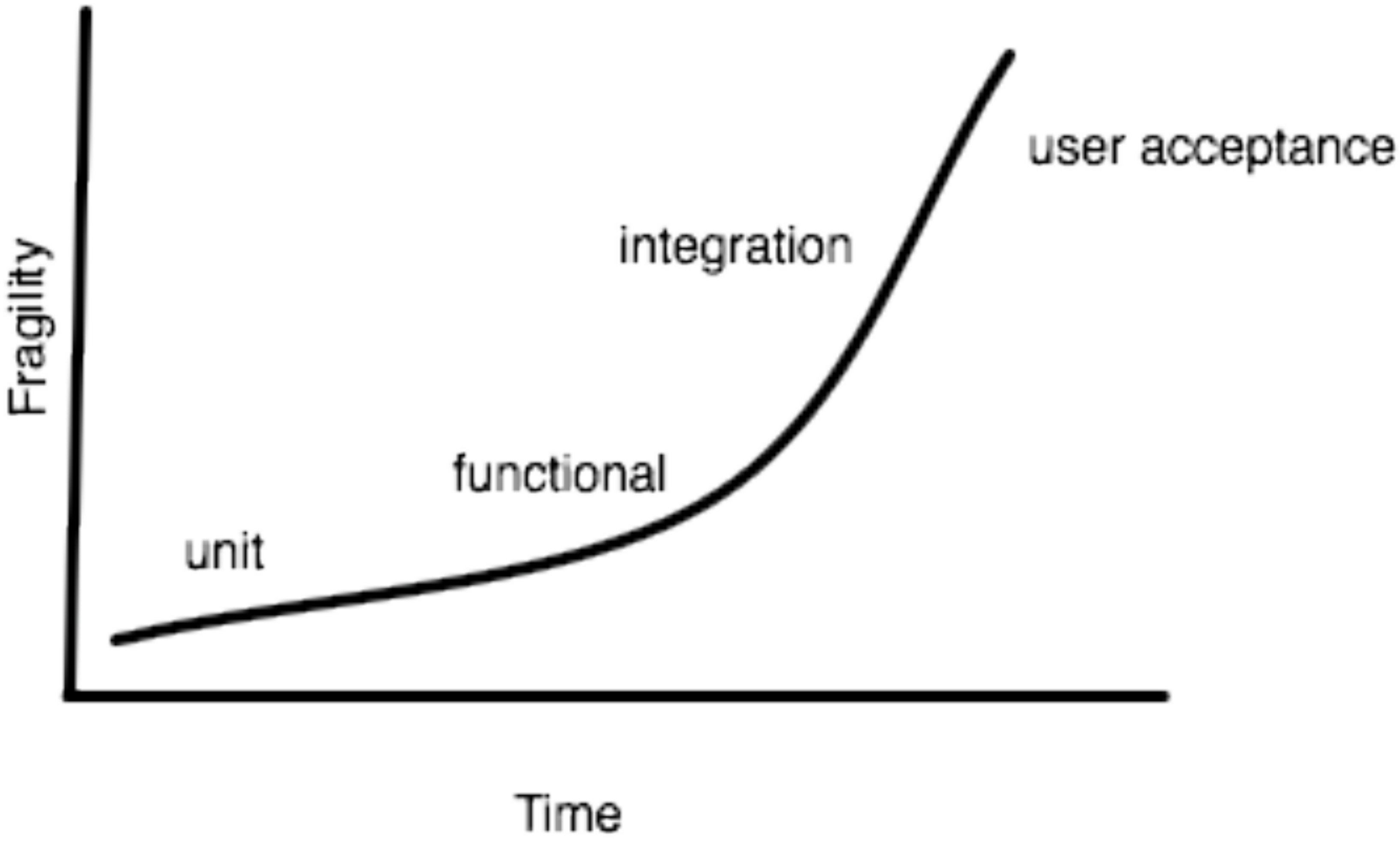
integration

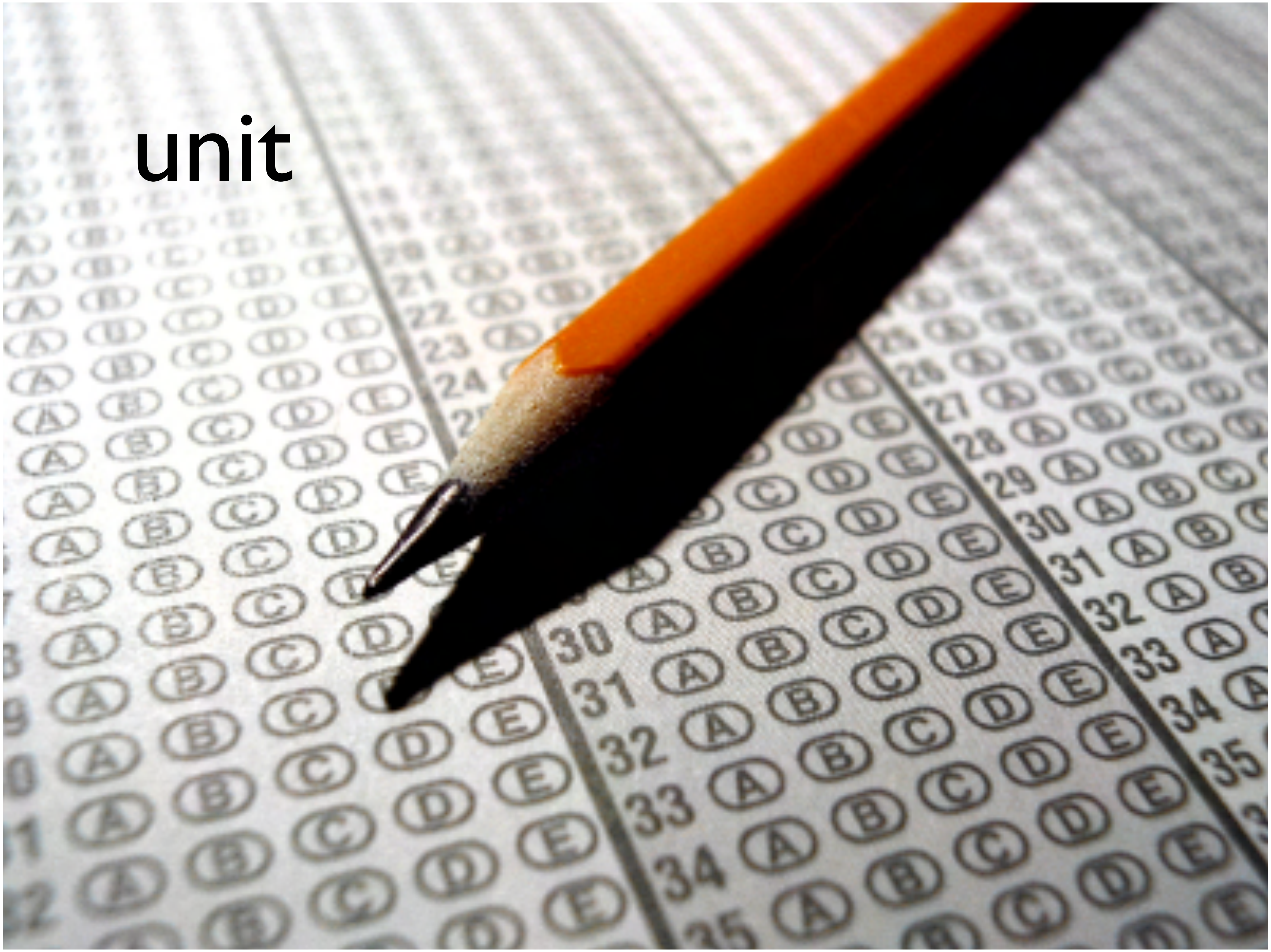
everything talks together

UAT (User Acceptance)

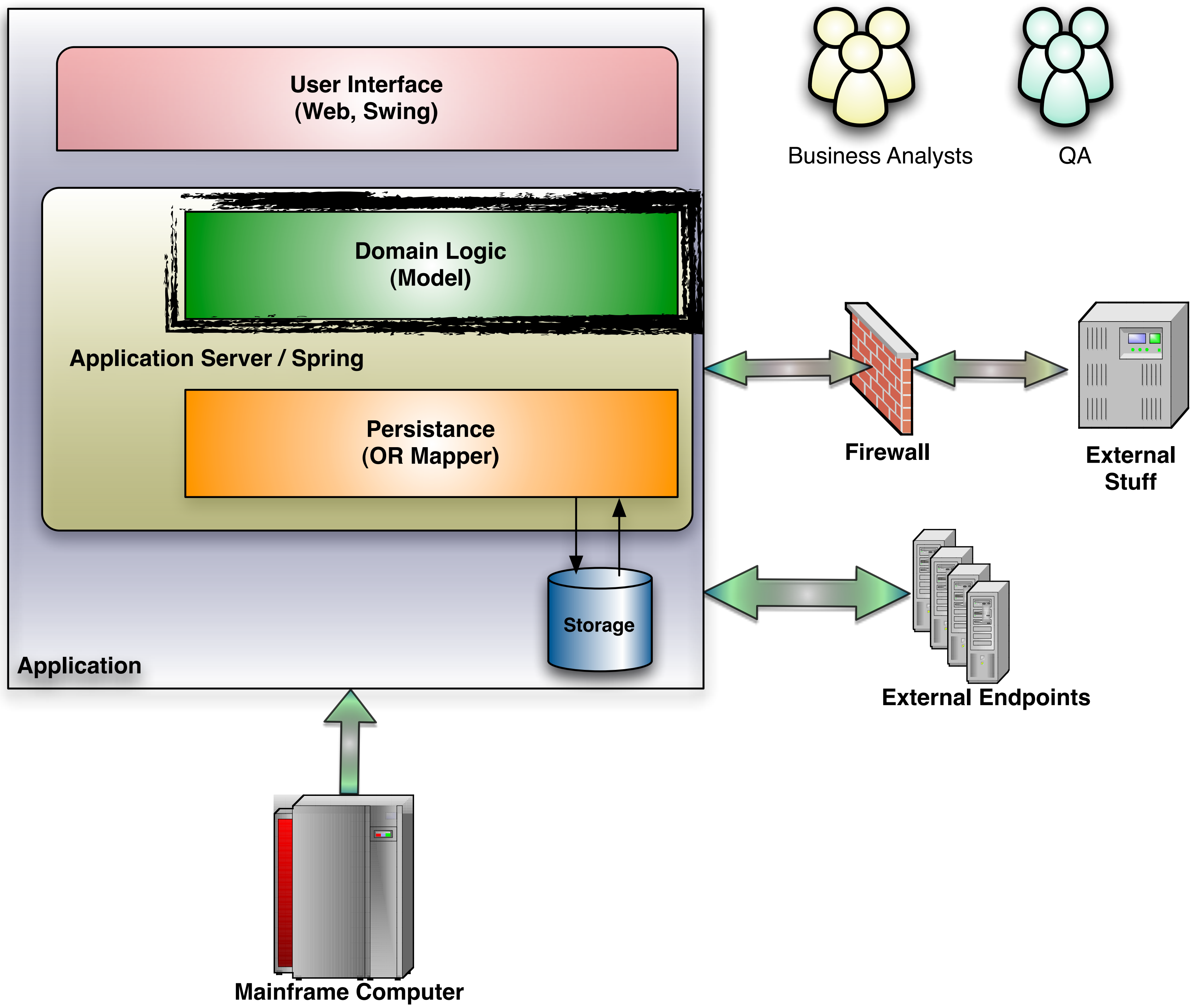


testing: fragility / time





unit


User Interface
(Web, Swing)
Domain Logic
(Model)
Application Server / Spring
Persistance
(OR Mapper)
Storage
Application
Business Analysts
QA
Firewall
External
Stuff
External Endpoints
Mainframe Computer

(mostly) a solved problem

xUnit TestNG Groovy JtestR

always test a weaker language with a stronger one

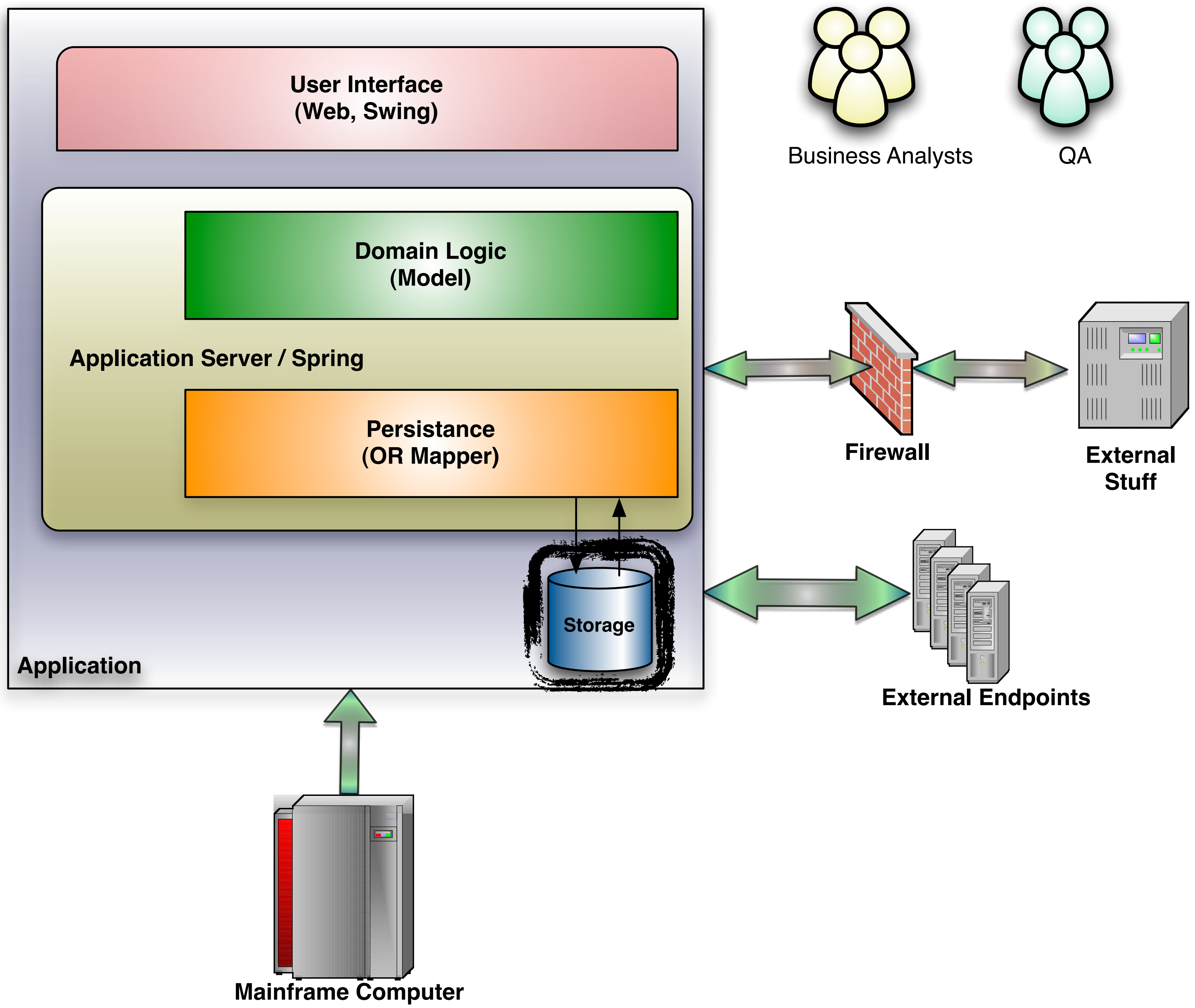
unit testing in java

groovy for “purer” java integration

JtestR for more elegant power

SPEECH
SPEECH
TEST
ESSAY
TEST
SPEECH
ESSAY
TEST
SPEECH
HOMEWORK

database


User Interface
(Web, Swing)
Business Analysts
QA
Domain Logic
(Model)
Application Server / Spring
Firewall
External
Stuff
Persistance
(OR Mapper)
Storage
External Endpoints
Application
Mainframe Computer

real data	vs.	fake data?
1001010100		0100101010
0100110010		1010010010
1001001000		0100100100
1001000100		0001001001

databases

known good state vs. "nuke & pave"?

real data

pros:

- real data!
- including invaluable years of cruft
- matches production exactly

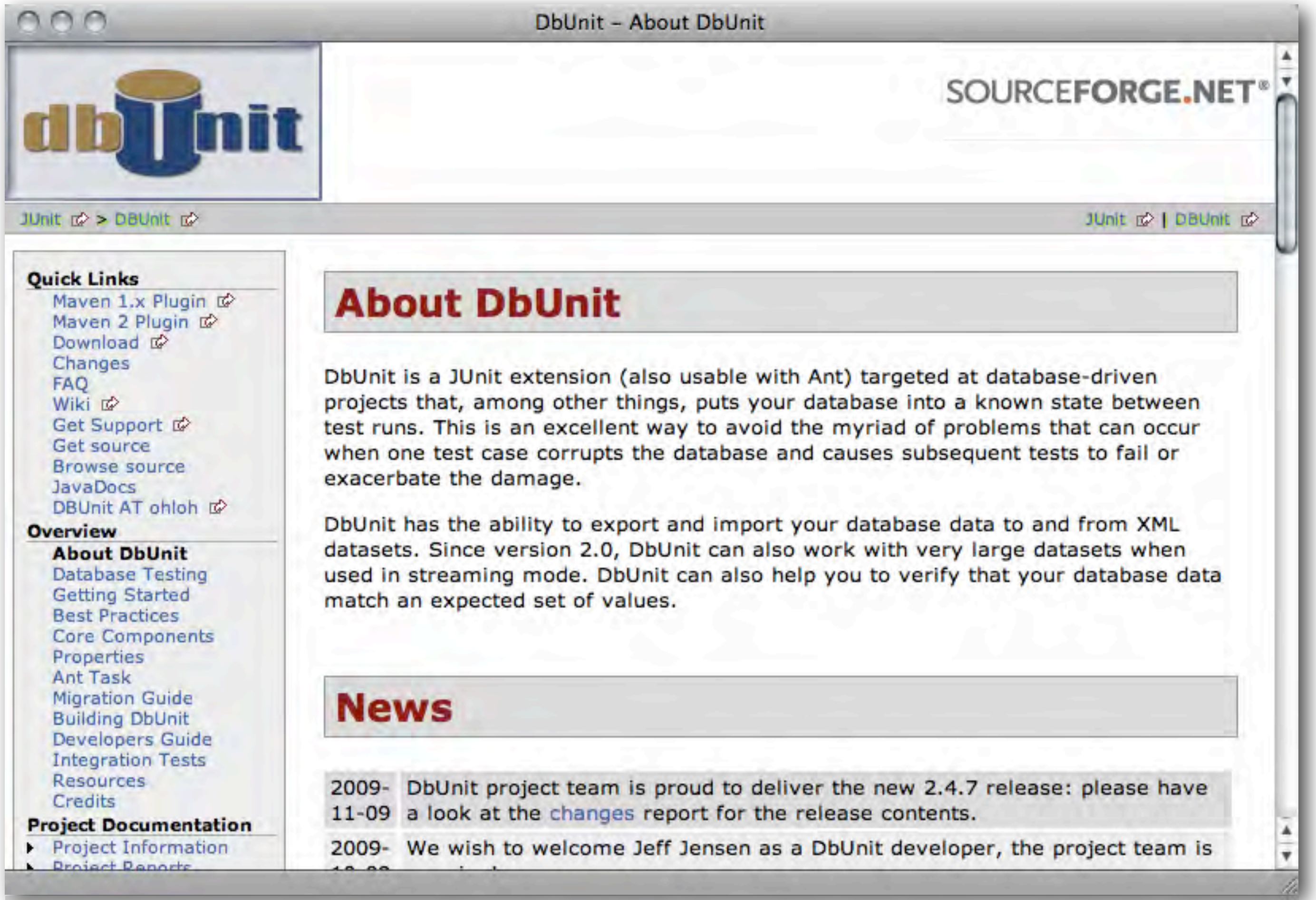
cons:

- real data!
- very hard to maintain state

tools help (dbDeploy, migrations)

linearly worse over time

known good state



About DbUnit

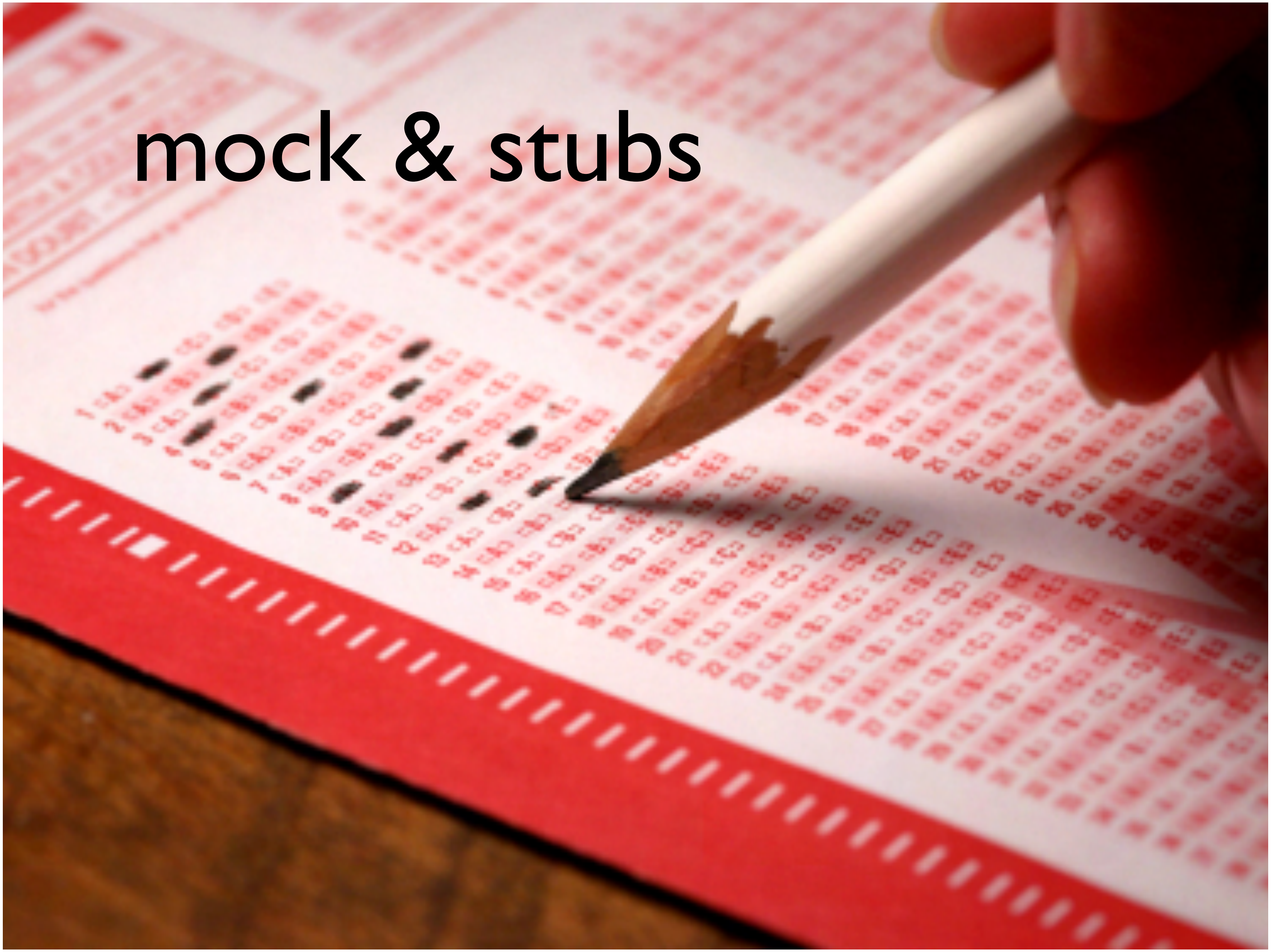
DbUnit is a JUnit extension (also usable with Ant) targeted at database-driven projects that, among other things, puts your database into a known state between test runs. This is an excellent way to avoid the myriad of problems that can occur when one test case corrupts the database and causes subsequent tests to fail or exacerbate the damage.

DbUnit has the ability to export and import your database data to and from XML datasets. Since version 2.0, DbUnit can also work with very large datasets when used in streaming mode. DbUnit can also help you to verify that your database data match an expected set of values.

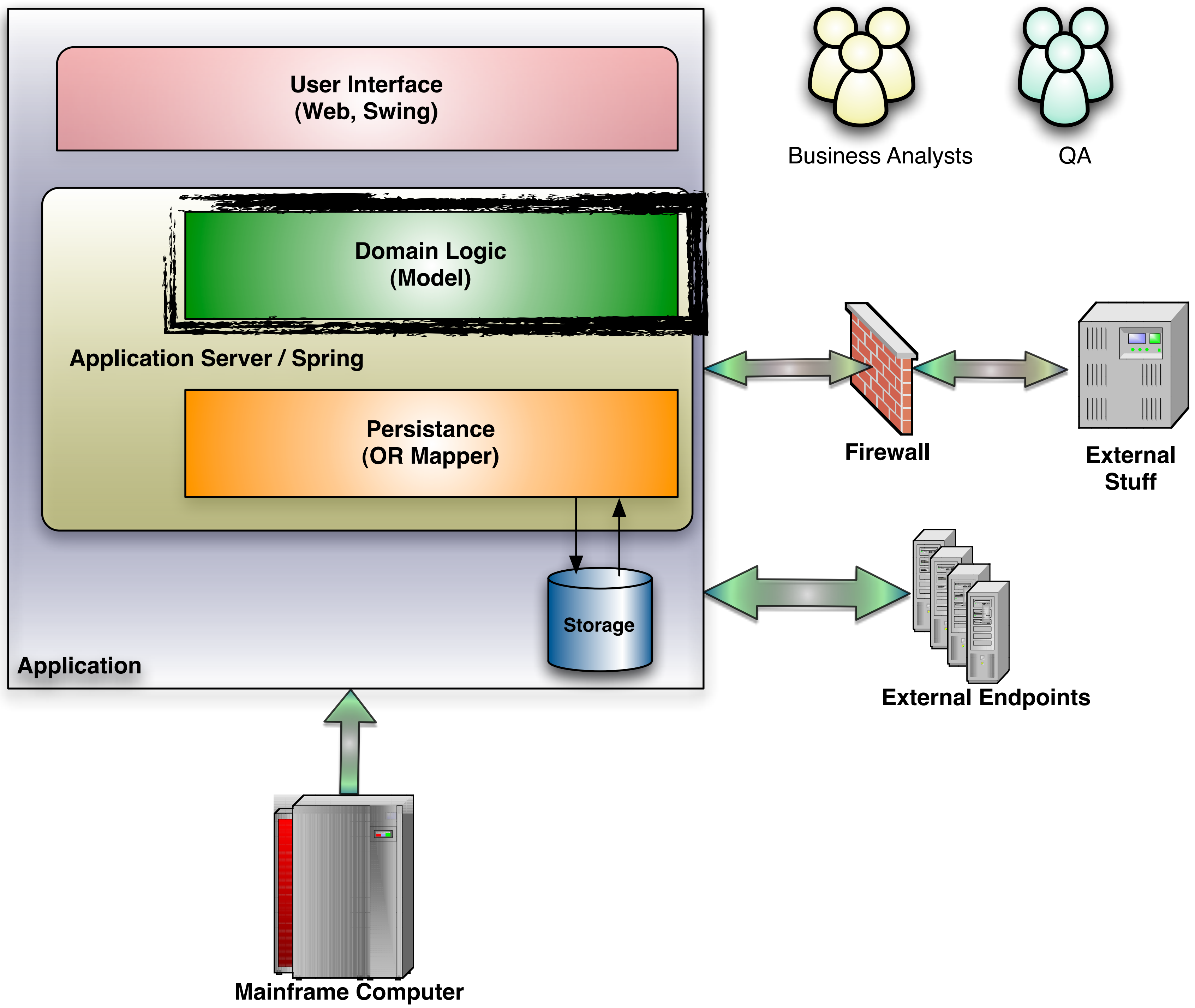
News

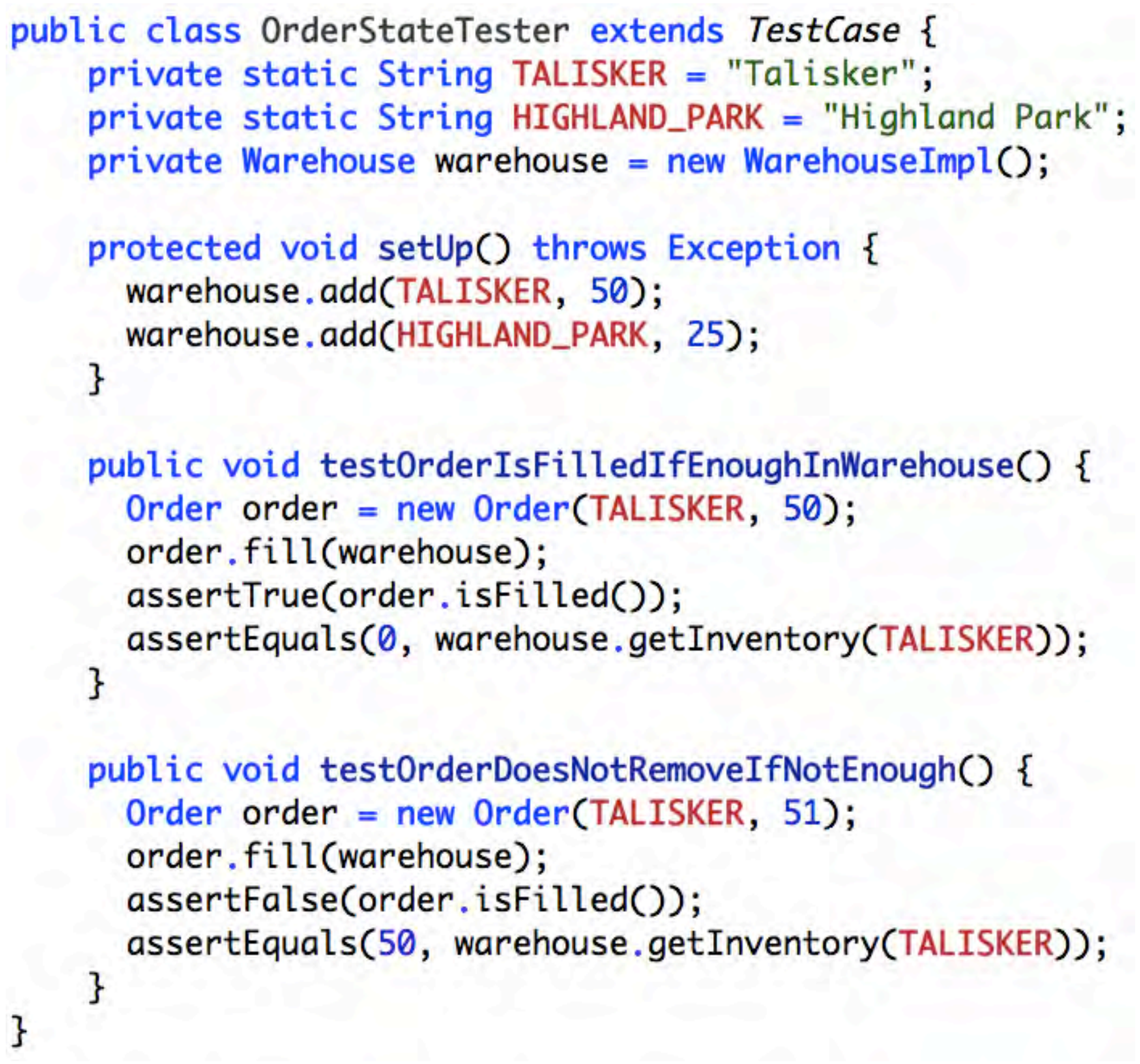
2009-11-09	DbUnit project team is proud to deliver the new 2.4.7 release: please have a look at the changes report for the release contents.
2009-	We wish to welcome Jeff Jensen as a DbUnit developer, the project team is

http://www.dbunit.org/



mock & stubs


User Interface
(Web, Swing)
Business Analysts
QA
Domain Logic
(Model)
Application Server / Spring
Firewall
External
Stuff
Persistance
(OR Mapper)
Storage
Application
External Endpoints
Mainframe Computer



```java
public class OrderStateTester extends TestCase {
    private static String TALISKER = "Talisker";
    private static String HIGHLAND_PARK = "Highland Park";
    private Warehouse warehouse = new WarehouseImpl();

    protected void setUp() throws Exception {
      warehouse.add(TALISKER, 50);
      warehouse.add(HIGHLAND_PARK, 25);
    }

    public void testOrderIsFilledIfEnoughInWarehouse() {
      Order order = new Order(TALISKER, 50);
      order.fill(warehouse);
      assertTrue(order.isFilled());
      assertEquals(0, warehouse.getInventory(TALISKER));
    }

    public void testOrderDoesNotRemoveIfNotEnough() {
      Order order = new Order(TALISKER, 51);
      order.fill(warehouse);
      assertFalse(order.isFilled());
      assertEquals(50, warehouse.getInventory(TALISKER));
    }
}
```

setup

exercise

verify

teardown

setup
(data)

setup
(expectations)

```
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();
    }
```

exercise

verification

```java
public class OrderEasyTester extends TestCase {
  private static String TALISKER = "Talisker";

  private MockControl warehouseControl;
  private Warehouse warehouseMock;

  public void setUp() {
    warehouseControl = MockControl.createControl(Warehouse.class);
    warehouseMock = (Warehouse) warehouseControl.getMock();
  }

  public void testFillingRemovesInventoryIfInStock() {
    //setup - data
    Order order = new Order(TALISKER, 50);

    //setup - expectations
    warehouseMock.hasInventory(TALISKER, 50);
    warehouseControl.setReturnValue(true);
    warehouseMock.remove(TALISKER, 50);
    warehouseControl.replay();

    //exercise
    order.fill(warehouseMock);

    //verify
    warehouseControl.verify();
    assertTrue(order.isFilled());
  }
```

terminology

test double - pretend object

dummy - object passed around but not used

fake - working implementations, but with shortcuts

stub - canned answers to calls within tests

$\left.\begin{array}{l}\text{dummy}\\\text{fake}\\\text{stub}\end{array}\right\}$ **state**

mock - objects pre-programmed with expectations

} behavior

classic TDDer

use real objects as much as possible

use doubles when real thing is awkward

mocks &| stubs

mockest TDDer

mock anything with interesting behavior

easy collaboration

use a real object

verify state directly

mock

behavior verification

awkward collaboration

case by case

take the easiest route

mock

behavior verification

edge case: hard state verification (cache)

behavior verification

mock

behavior verification

state vs behavior verification
mostly not a big deal

classic vs mock TDDer

mockist

mocks as design tool

“need driven development”

encourages thinking about collaborations

explore the outbound interfaces of the system under test

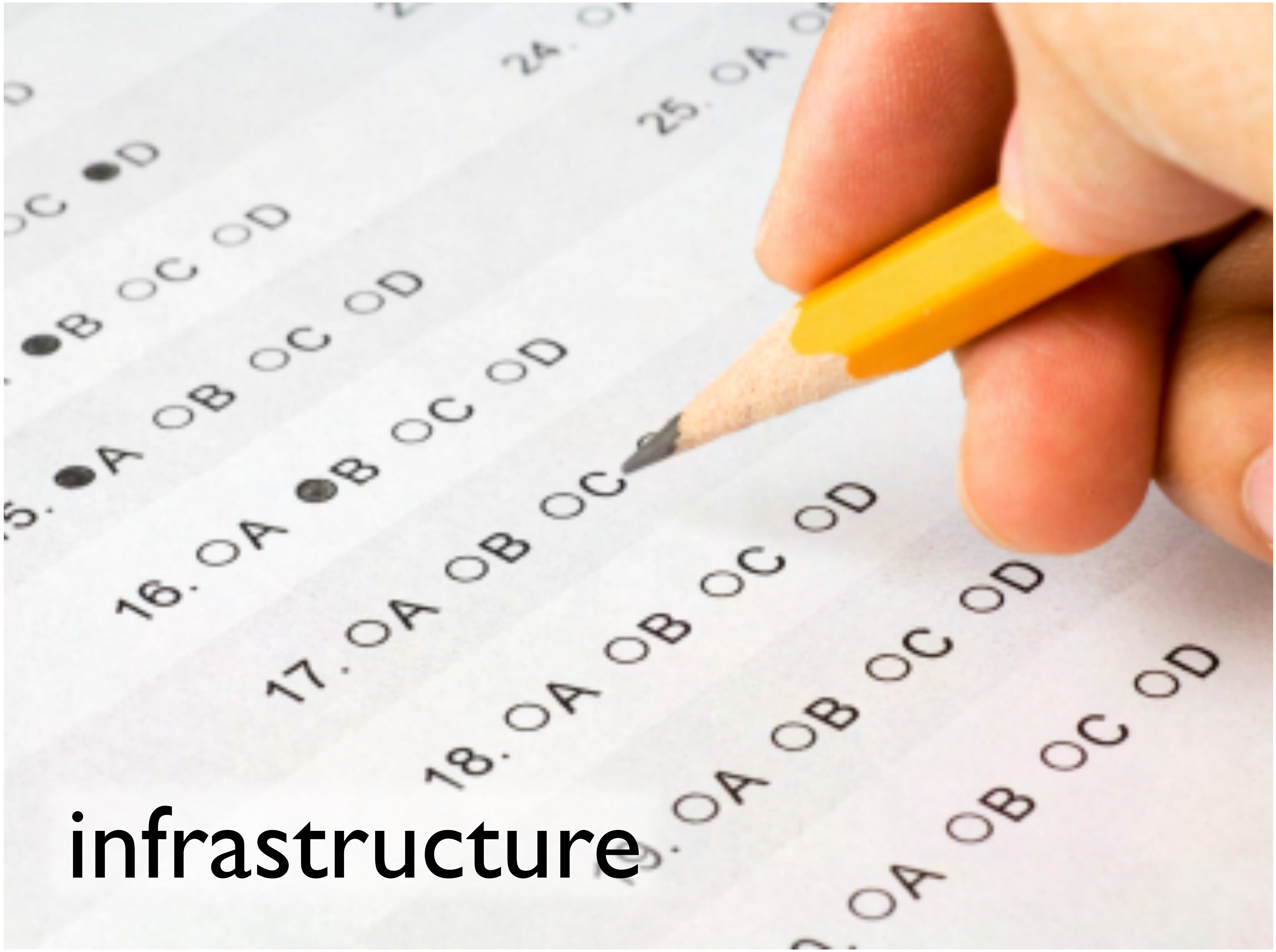
classicist

start with stubs & hard coded values

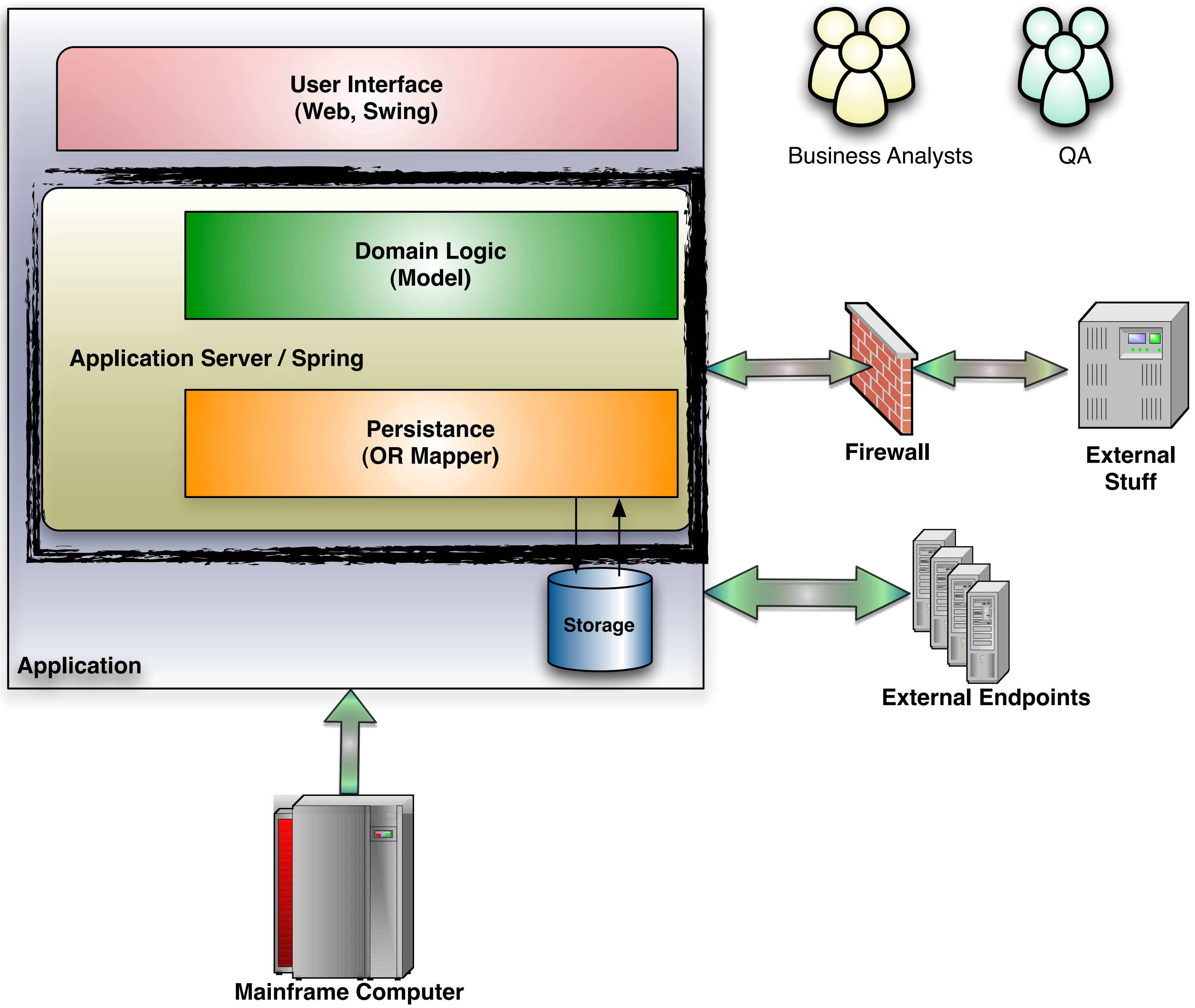
gradually build real values

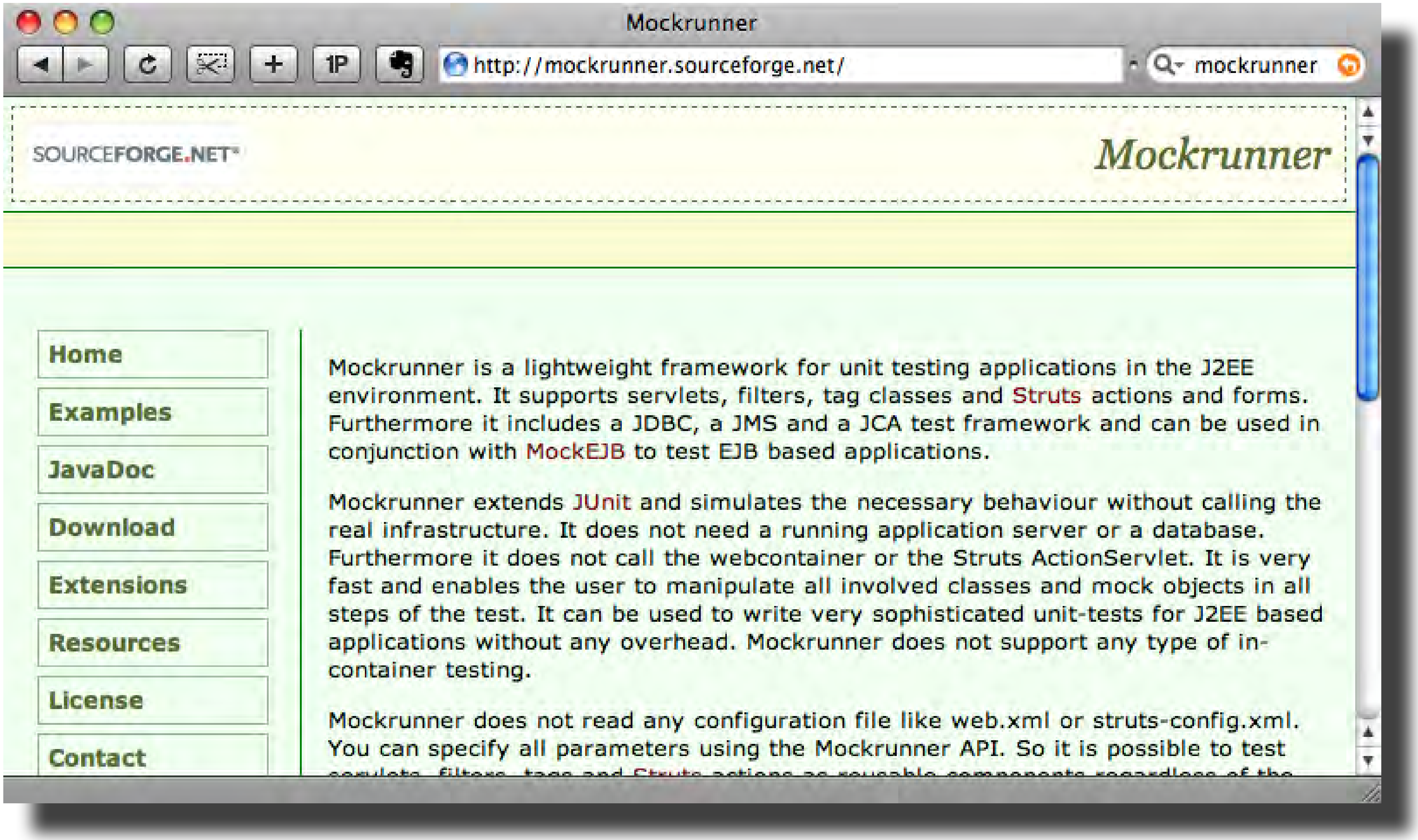
“middle out”

build domain model and gradually expand



infrastructure


User Interface
(Web, Swing)
Business Analysts
QA
Domain Logic
(Model)
Application Server / Spring
Persistance
(OR Mapper)
Firewall
External
Stuff
Storage
Application
External Endpoints
Mainframe Computer



Mockrunner is a lightweight framework for unit testing applications in the J2EE environment. It supports servlets, filters, tag classes and Struts actions and forms. Furthermore it includes a JDBC, a JMS and a JCA test framework and can be used in conjunction with MockEJB to test EJB based applications.

Mockrunner extends JUnit and simulates the necessary behaviour without calling the real infrastructure. It does not need a running application server or a database. Furthermore it does not call the webcontainer or the Struts ActionServlet. It is very fast and enables the user to manipulate all involved classes and mock objects in all steps of the test. It can be used to write very sophisticated unit-tests for J2EE based applications without any overhead. Mockrunner does not support any type of in-container testing.

Mockrunner does not read any configuration file like web.xml or struts-config.xml. You can specify all parameters using the Mockrunner API. So it is possible to test

http://mockrunner.sourceforge.net/

lightweight

J2EE

servlets

filters

Mockrunner

JDBC

JMS

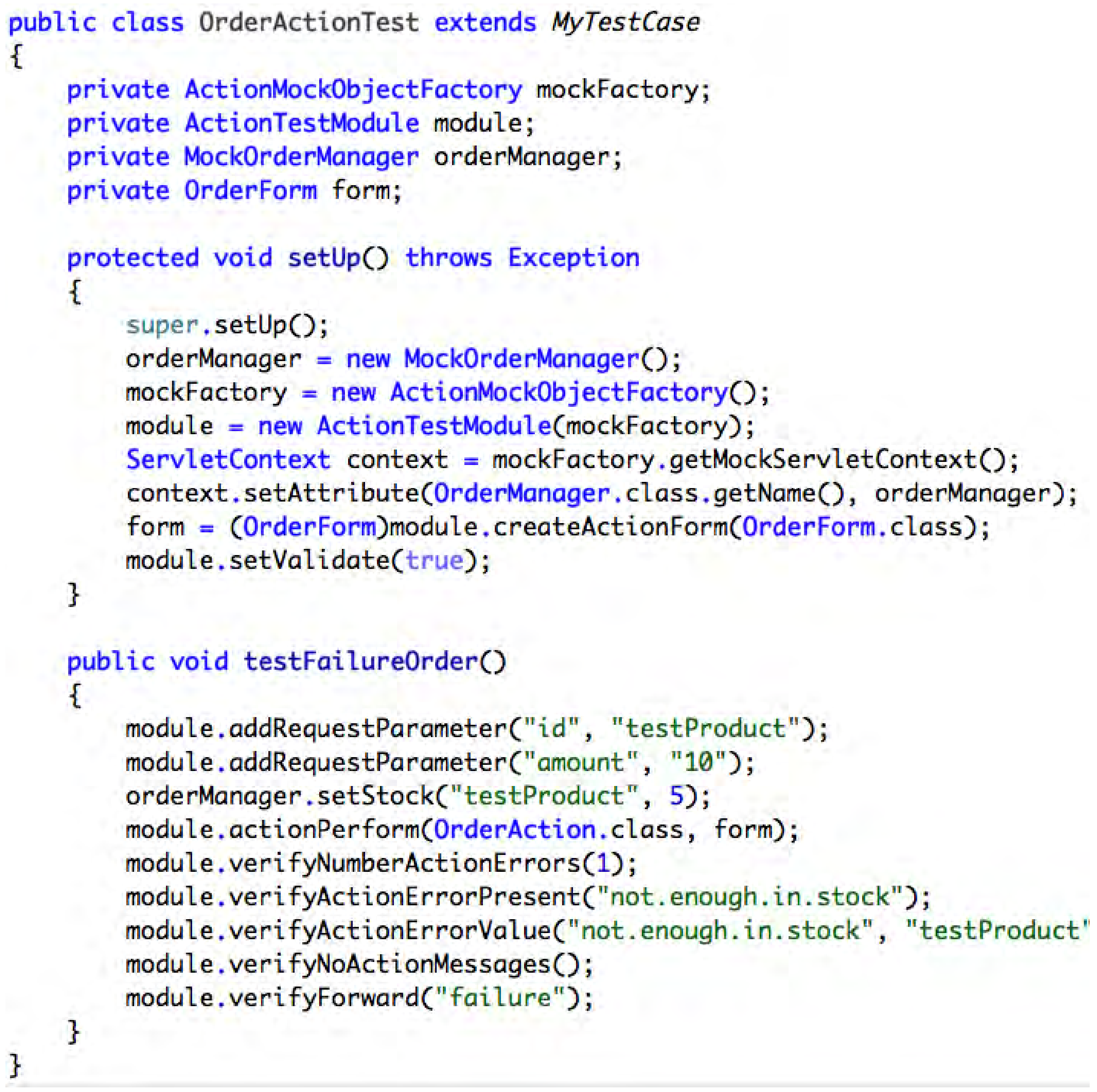
Struts actions
& forms

JCA

JUnit

```java
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");
    }
}
```

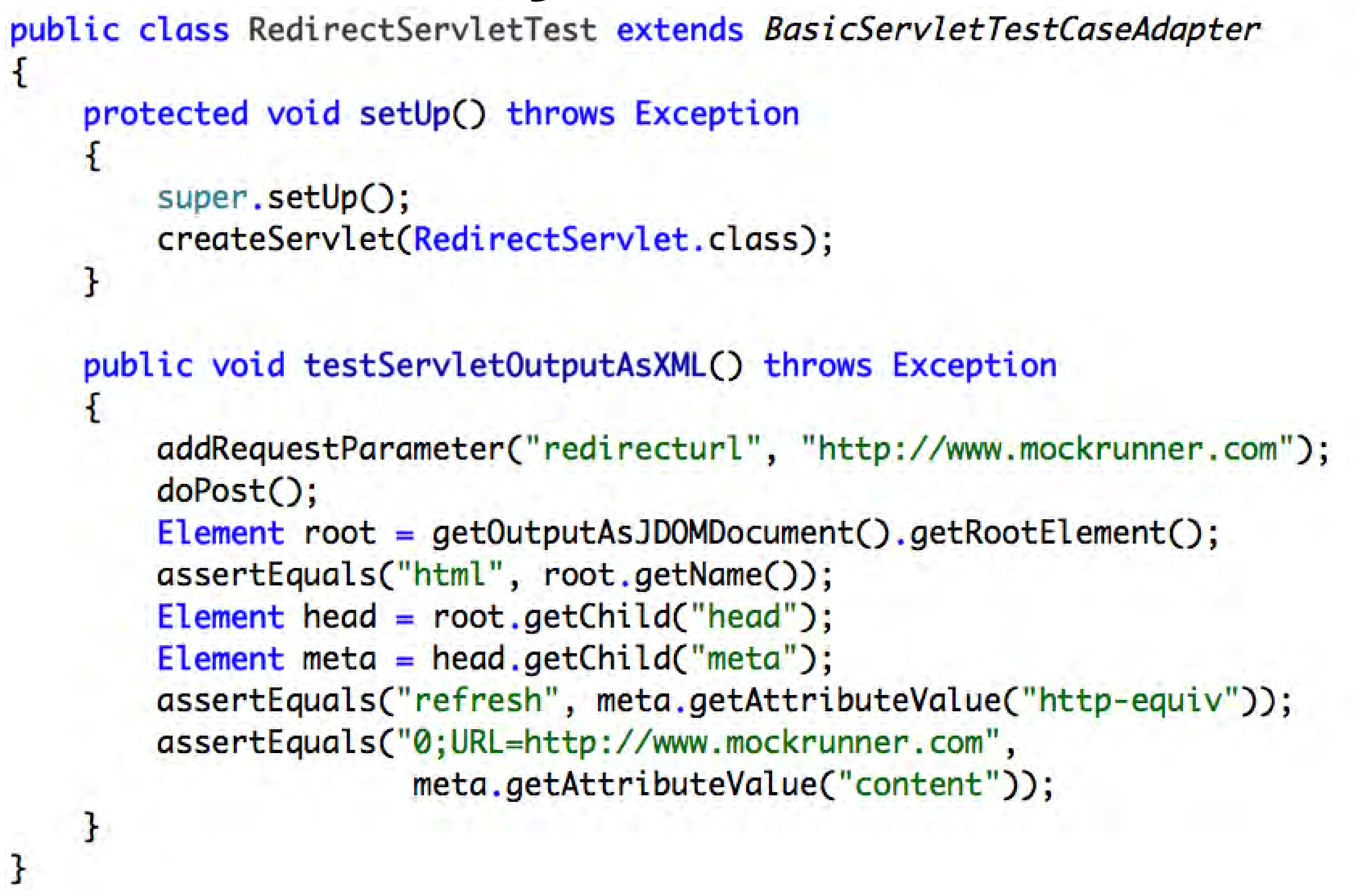


```
public class OrderActionTest extends MyTestCase
{
    private ActionMockObjectFactory mockFactory;
    private ActionTestModule module;
    private MockOrderManager orderManager;
    private OrderForm form;

    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");
    }
}
```

```java
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());
    }
}
```

```java
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

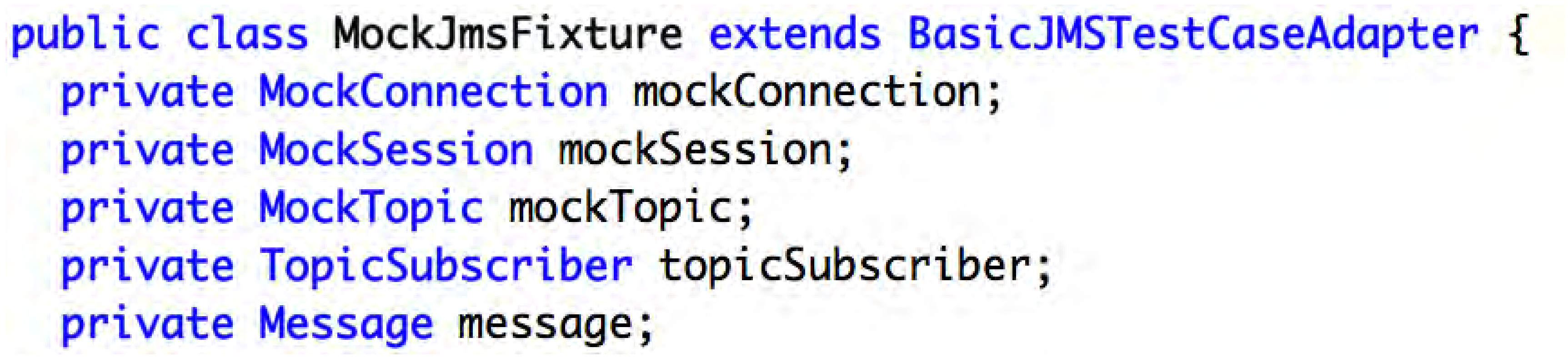


```
public class RedirectServletTest extends BasicServletTestCaseAdapter
{
    protected void setUp() throws Exception
    {
        super.setUp();
        createServlet(RedirectServlet.class);
    }

    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();
    }
}
```

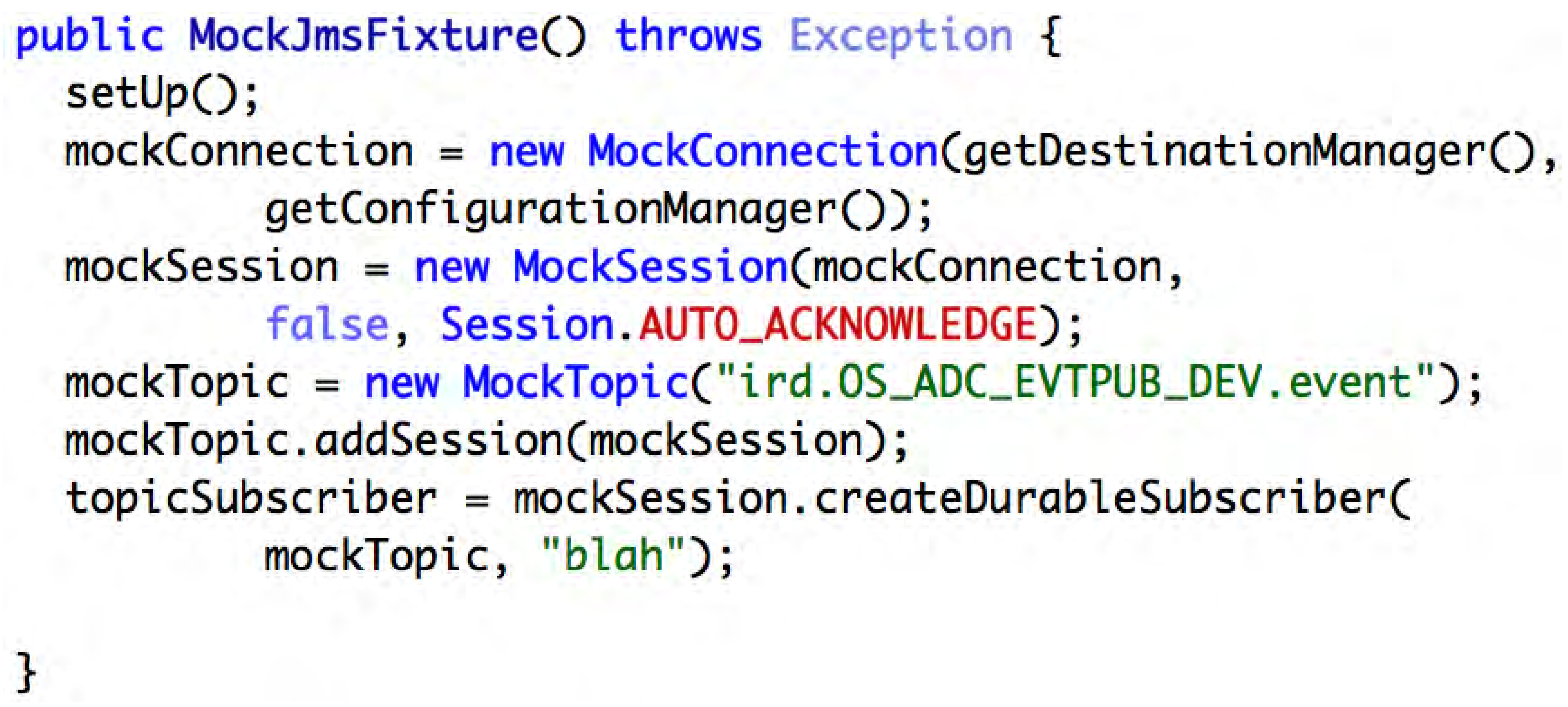
```java
public class BankTest extends BasicJDBCTestCaseAdapter
{
    private void prepareResultSet()
    {
        MockConnection connection =
            getJDBCMockObjectFactory().getMockConnection();
        StatementResultSetHandler statementHandler =
            connection.getStatementResultSetHandler();
        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



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

creating the fixture

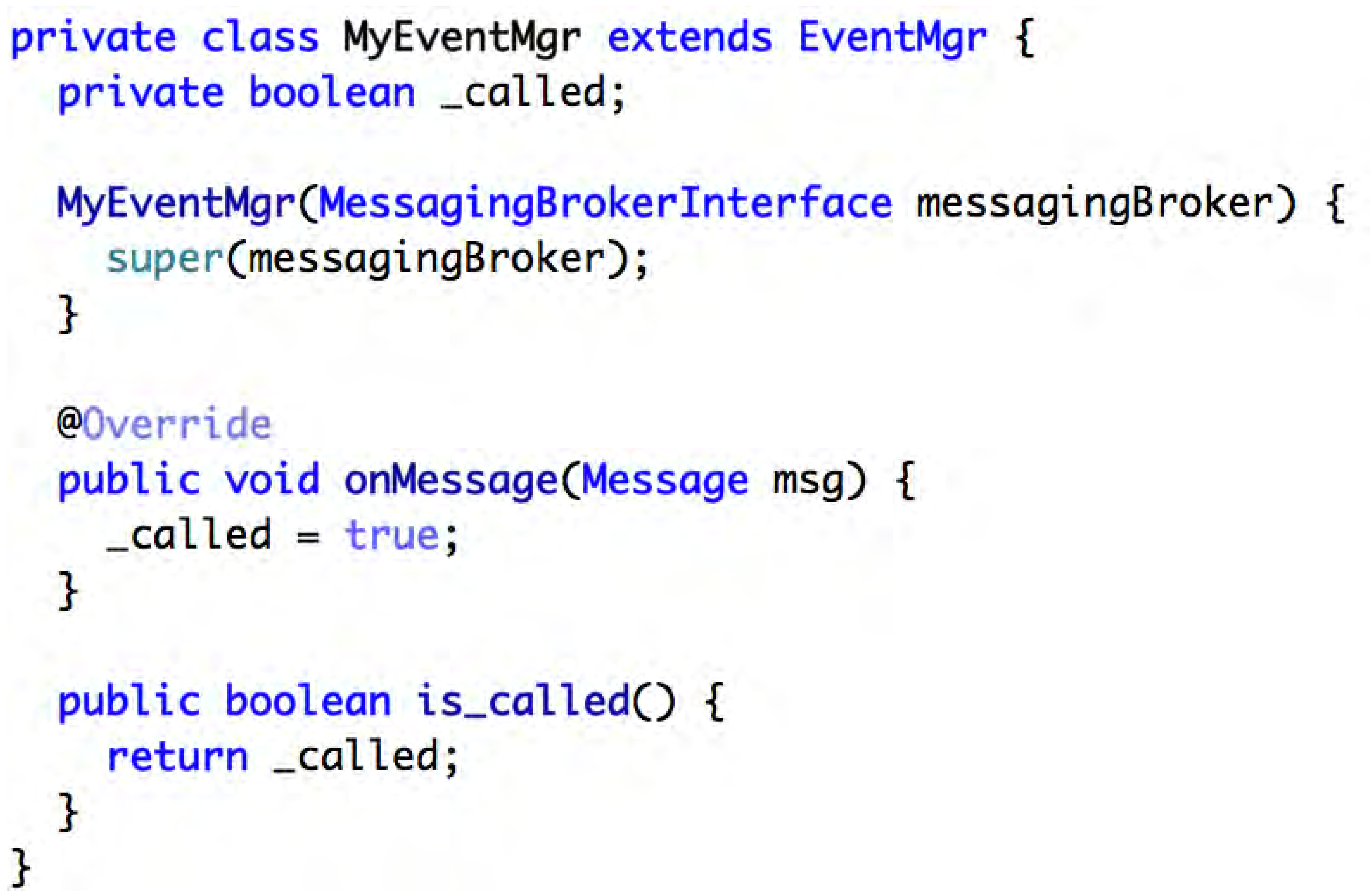


```
public MockJmsFixture() throws Exception {
  setUp();
  mockConnection = new MockConnection(getDestinationManager(),
          getConfigurationManager());
  mockSession = new MockSession(mockConnection,
          false, Session.AUTO_ACKNOWLEDGE);
  mockTopic = new MockTopic("ird.OS_ADC_EVTPUB_DEV.event");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

}
```

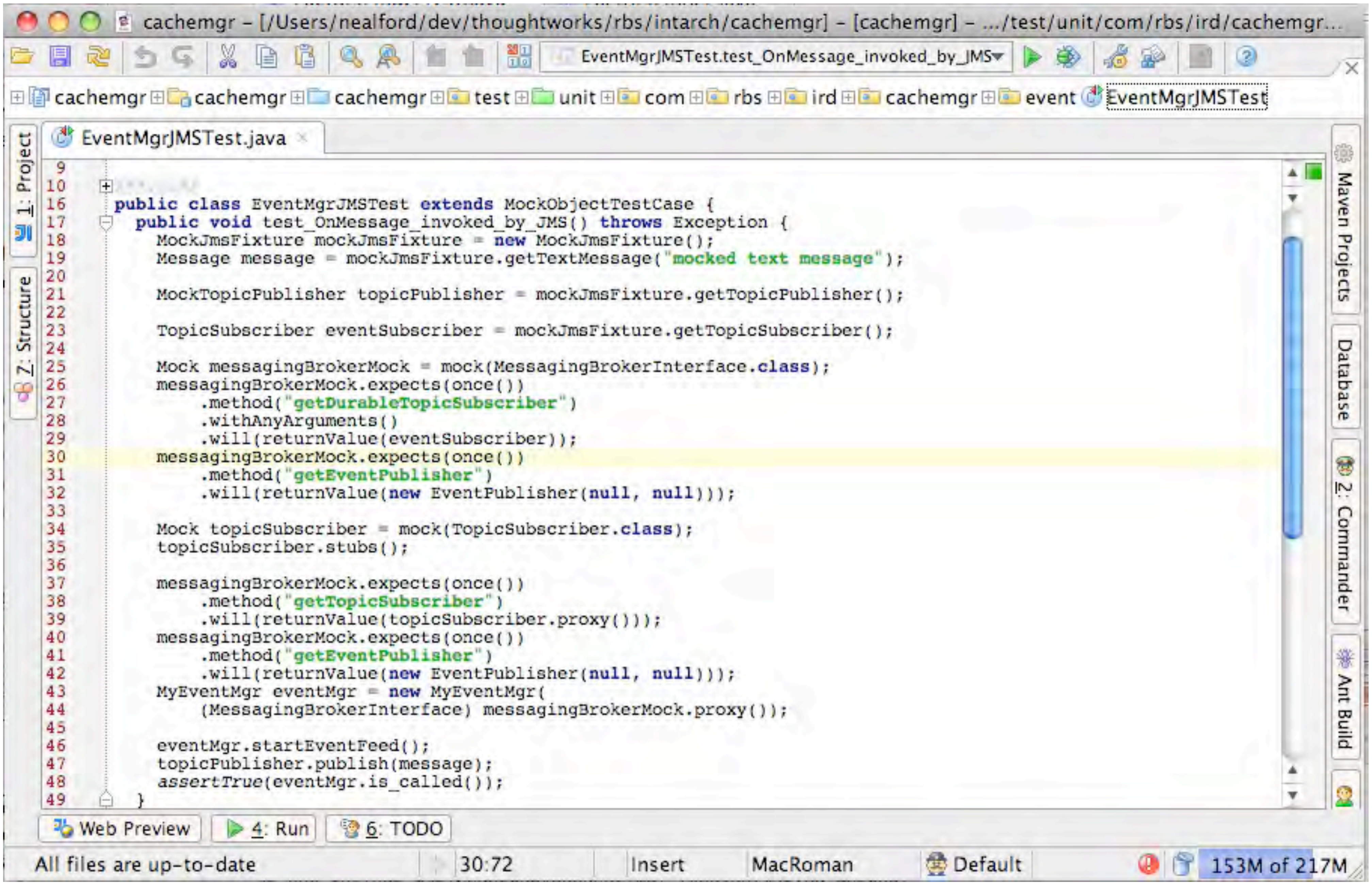
the test

```
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



```java
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;
  }
}
```

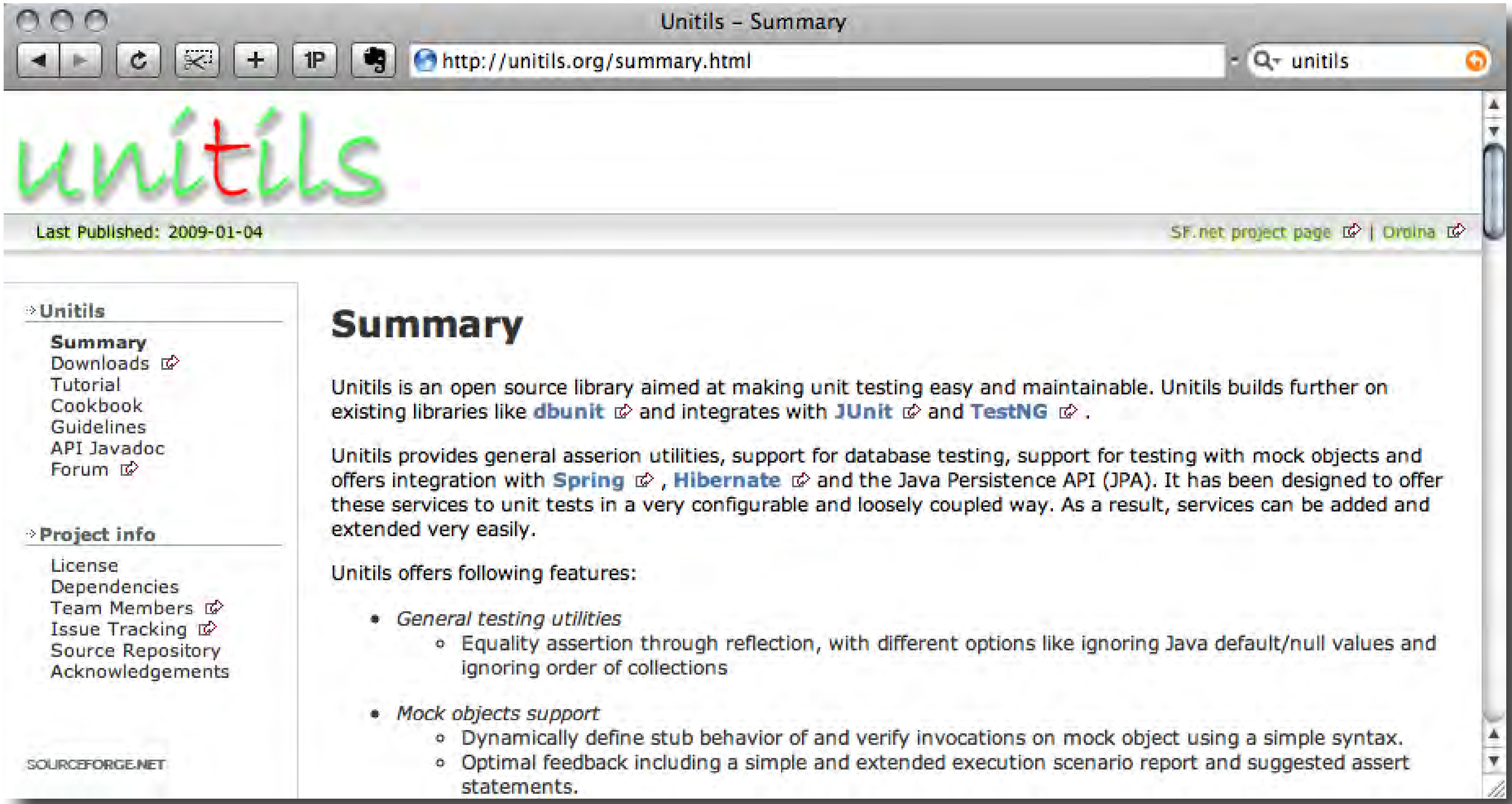

cachemgr - [/Users/nealford/dev/thoughtworks/rbs/intarch/cachemgr] - [cachemgr] - .../test/unit/com/rbs/ird/cachemgr...

```java
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());
  }
```

All files are up-to-date 30:72 Insert MacRoman Default 153M of 217M

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
Web Preview
4: Run
6: TODO

unitils



Summary

Unitils is an open source library aimed at making unit testing easy and maintainable. Unitils builds further on existing libraries like **dbunit** and integrates with **JUnit** and **TestNG** .

Unitils provides general asserion utilities, support for database testing, support for testing with mock objects and offers integration with **Spring** , **Hibernate** and the Java Persistence API (JPA). It has been designed to offer these services to unit tests in a very configurable and loosely coupled way. As a result, services can be added and extended very easily.

Unitils offers following features:

- *General testing utilities*
  - Equality assertion through reflection, with different options like ignoring Java default/null values and ignoring order of collections
- *Mock objects support*
  - Dynamically define stub behavior of and verify invocations on mock object using a simple syntax.
  - Optimal feedback including a simple and extended execution scenario report and suggested assert statements.

http://unitils.org/

unitils

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

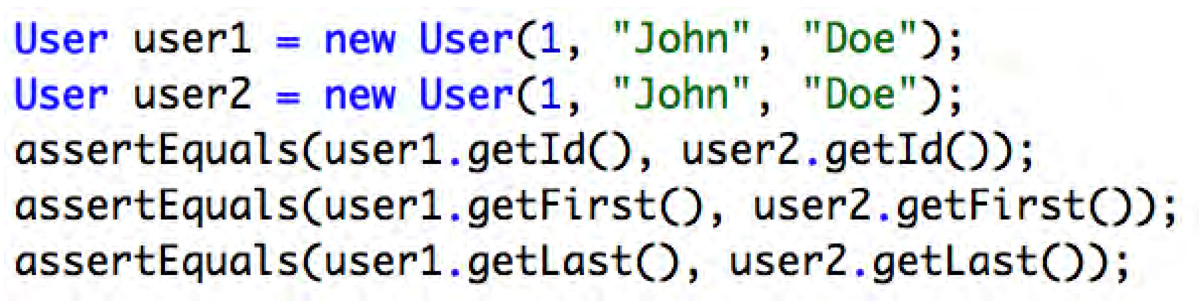
```
User user1 = new User(1, "John", "Doe");
User user2 = new User(1, "Jane", "Smith");
assertEquals(user1, user2);}
```

what is tested?

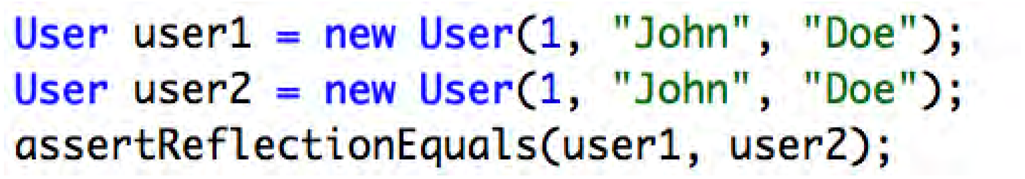
```
User user1 = new User(1, "John", "Doe");
User user2 = new User(1, "John", "Doe");
assertEquals(user1.getId(), user2.getId());
assertEquals(user1.getFirst(), user2.getFirst());
assertEquals(user1.getLast(), user2.getLast());
```

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

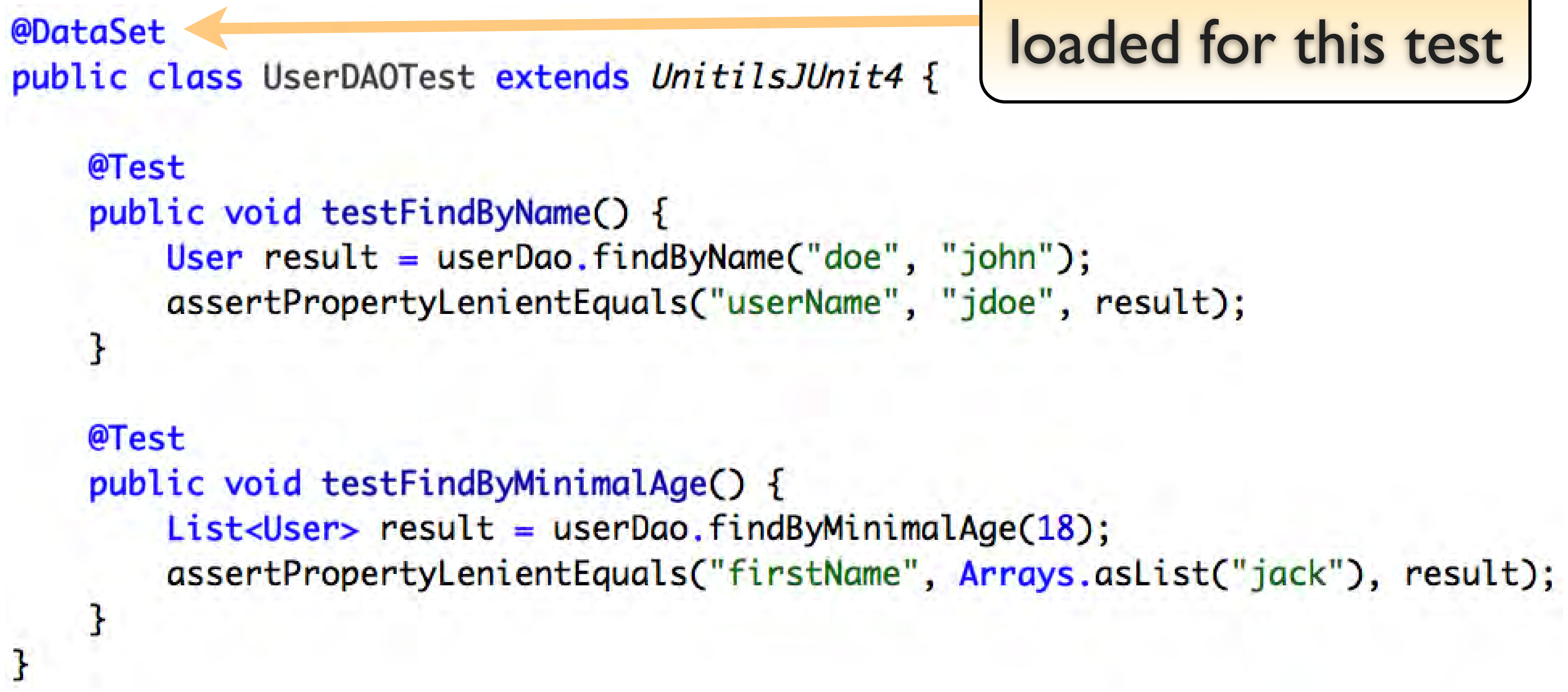
```
List<Integer> myList = Arrays.asList(3, 2, 1);
assertReflectionEquals(Arrays.asList(1, 2, 3), myList, LENIENT_ORDER);
```

```
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);
```

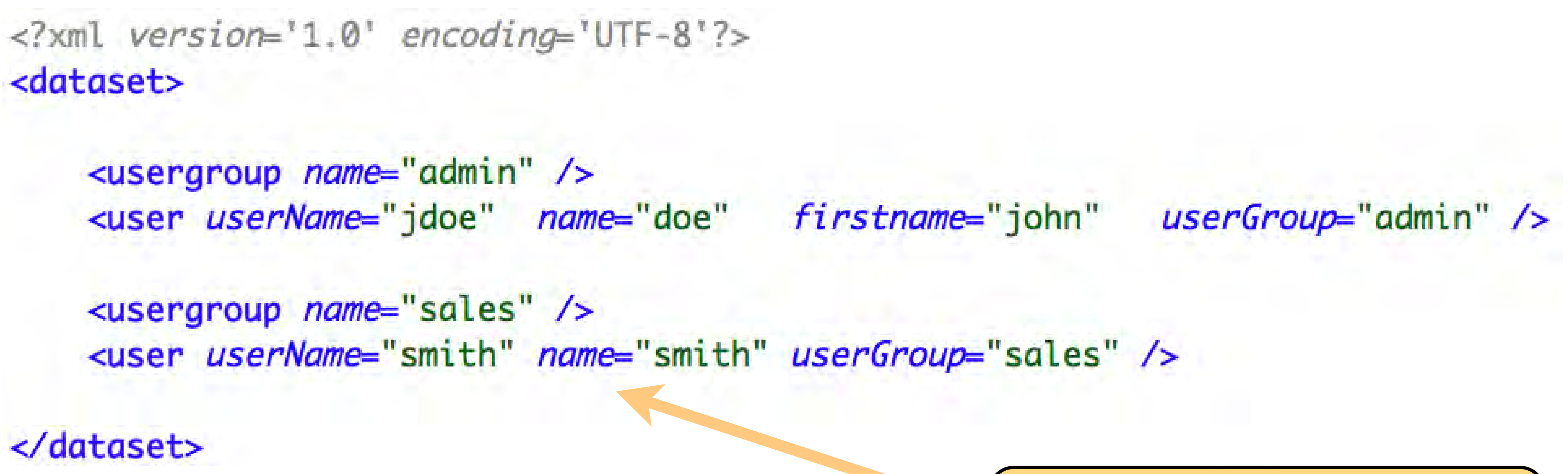
dbUnit support

dbUnit files to be loaded for this test



```java
@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

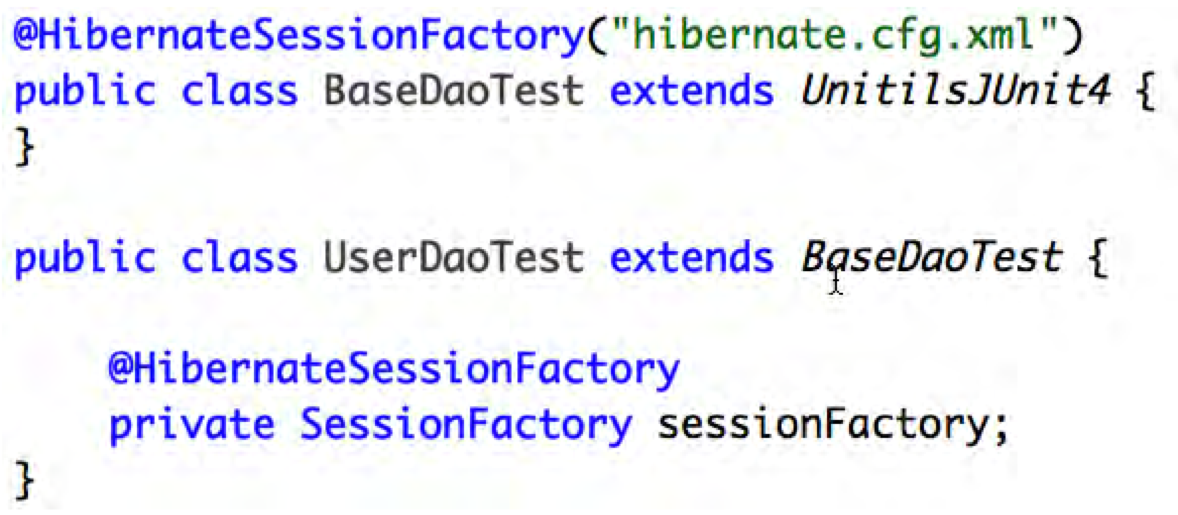


```xml
<?xml version='1.0' encoding='UTF-8'?>
<dataset>

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

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

</dataset>
```

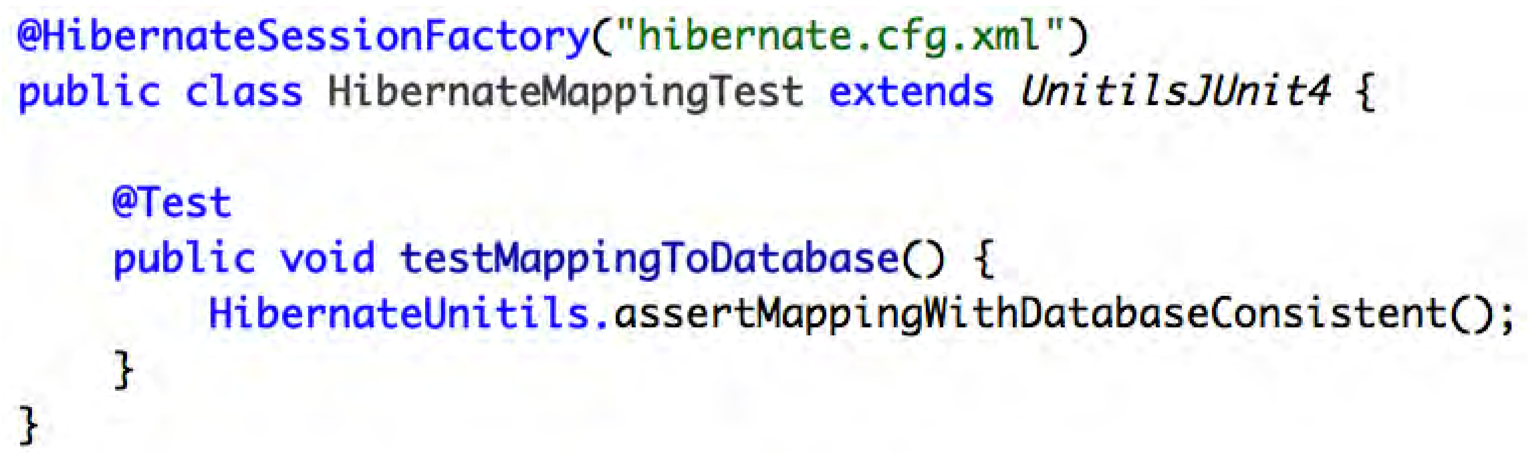
firstname == null

this data will be loaded prior to test run

hibernate support



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

public class UserDaoTest extends BaseDaoTest {

    @HibernateSessionFactory
    private SessionFactory sessionFactory;
}
```



```java
@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

```java
public class UserServiceTest extends UnitilsJUnit4 {

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

}
```

ApplicationContext

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

@SpringBeanByName
private UserService userService;

@SpringBeanByType
private UserService userService;
```

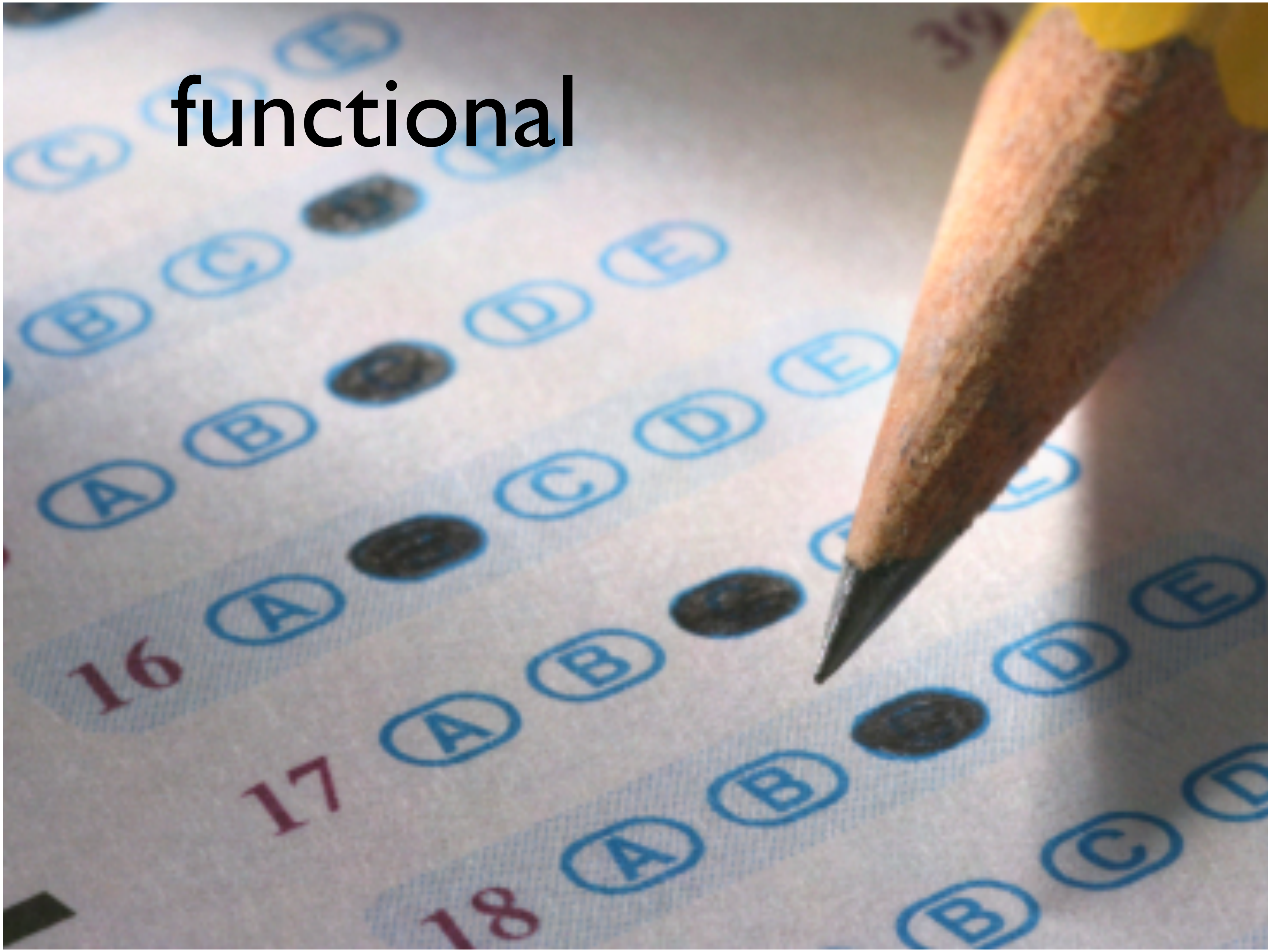
injection

```
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



functional

coarse grained state-based testing

using traditional unit testing tools or BDD

useful when retro-fitting unit tests

connected (no mocking)

collaboratively developed with analysts

connected tests:
1 strategy

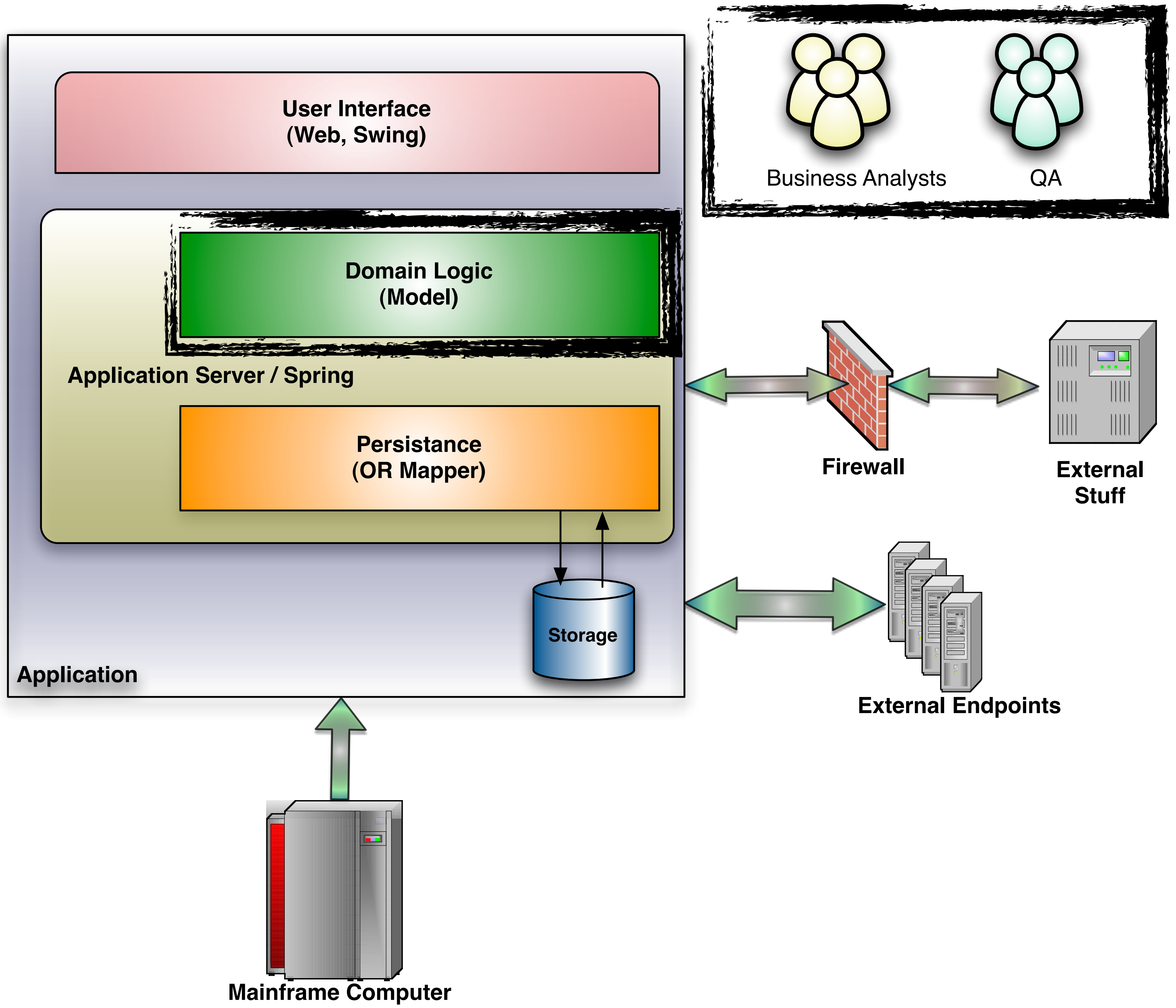
unit tests:

- all data mocked
- all external endpoints mocked

functional tests:

- everything works

behavior driven development



User Interface (Web, Swing)
Domain Logic (Model)
Application Server / Spring
Persistance (OR Mapper)
Storage
Application
Business Analysts
QA
Firewall
External Stuff
External Endpoints
Mainframe Computer

BDD

encourages collaboration between developes, BAs, testers, & other stakeholders

developed by Dan North

focuses on exposing internal logic (typically business rules) to review by stakeholders

native language + DDD’s ubiquitous language

test driven requirements gathering

JBehave

Cucumber

BDD tools

easyb -- bdd in java can't get any easier

RSpec

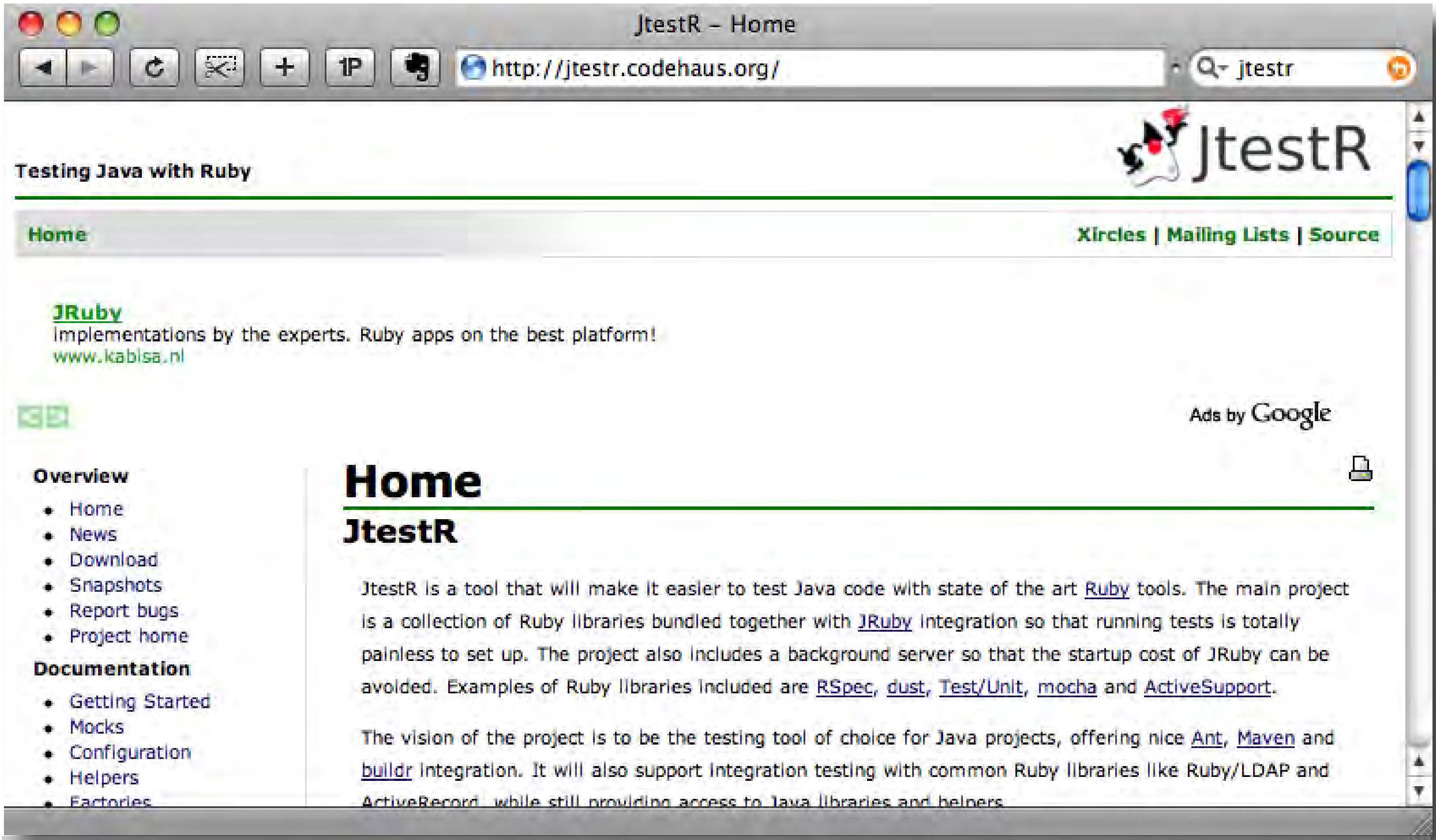
JBehave

```
Trader is alerted of status

Scenario:
In order to ensure a quick response
As a trader
I want to monitor stock prices

Given a stock of symbol STK1 and a threshold of 15.0
When the stock is traded at price 5.0
Then the alert status is OFF
When the stock is sold at price 11.0
Then the alert status is OFF
When the stock is sold at price 16.0
Then the alert status is ON

Scenario:
In order to ensure a quick response
As a trader
I want to monitor stock prices

Given a stock of <symbol> and a <threshold>
When the stock is traded with <price>
Then the trader is alerted with <status>

Examples:
|symbol|threshold|price|status|
|STK1|15.0|5.0|OFF|
|STK1|15.0|11.0|OFF|
|STK1|15.0|16.0|ON|
```

easyb -- bdd in java can't get any easier

```
given "an invalid zip code", {
  invalidzipcode = "221o1"
}

and "given the zipcodevalidator is initialized", {
  zipvalidate = new ZipCodeValidator()
}

when "validate is invoked with the invalid zip code", {
  value = zipvalidate.validate(invalidzipcode)
}

then "the validator instance should return false", {
  value.shouldBe false
}
```

rspec via jruby



JtestR – Home

http://jtestr.codehaus.org/

Testing Java with Ruby

JtestR

Home

Xircles | Mailing Lists | Source

JRuby
Implementations by the experts. Ruby apps on the best platform!
www.kabisa.nl

Ads by Google

Overview

- Home
- News
- Download
- Snapshots
- Report bugs
- Project home

Documentation

- Getting Started
- Mocks
- Configuration
- Helpers
- Factories

Home

JtestR

JtestR is a tool that will make it easier to test Java code with state of the art Ruby tools. The main project is a collection of Ruby libraries bundled together with JRuby integration so that running tests is totally painless to set up. The project also includes a background server so that the startup cost of JRuby can be avoided. Examples of Ruby libraries included are RSpec, dust, Test/Unit, mocha and ActiveSupport.

The vision of the project is to be the testing tool of choice for Java projects, offering nice Ant, Maven and buildr integration. It will also support integration testing with common Ruby libraries like Ruby/LDAP and ActiveRecord, while still providing access to Java libraries and helpers.

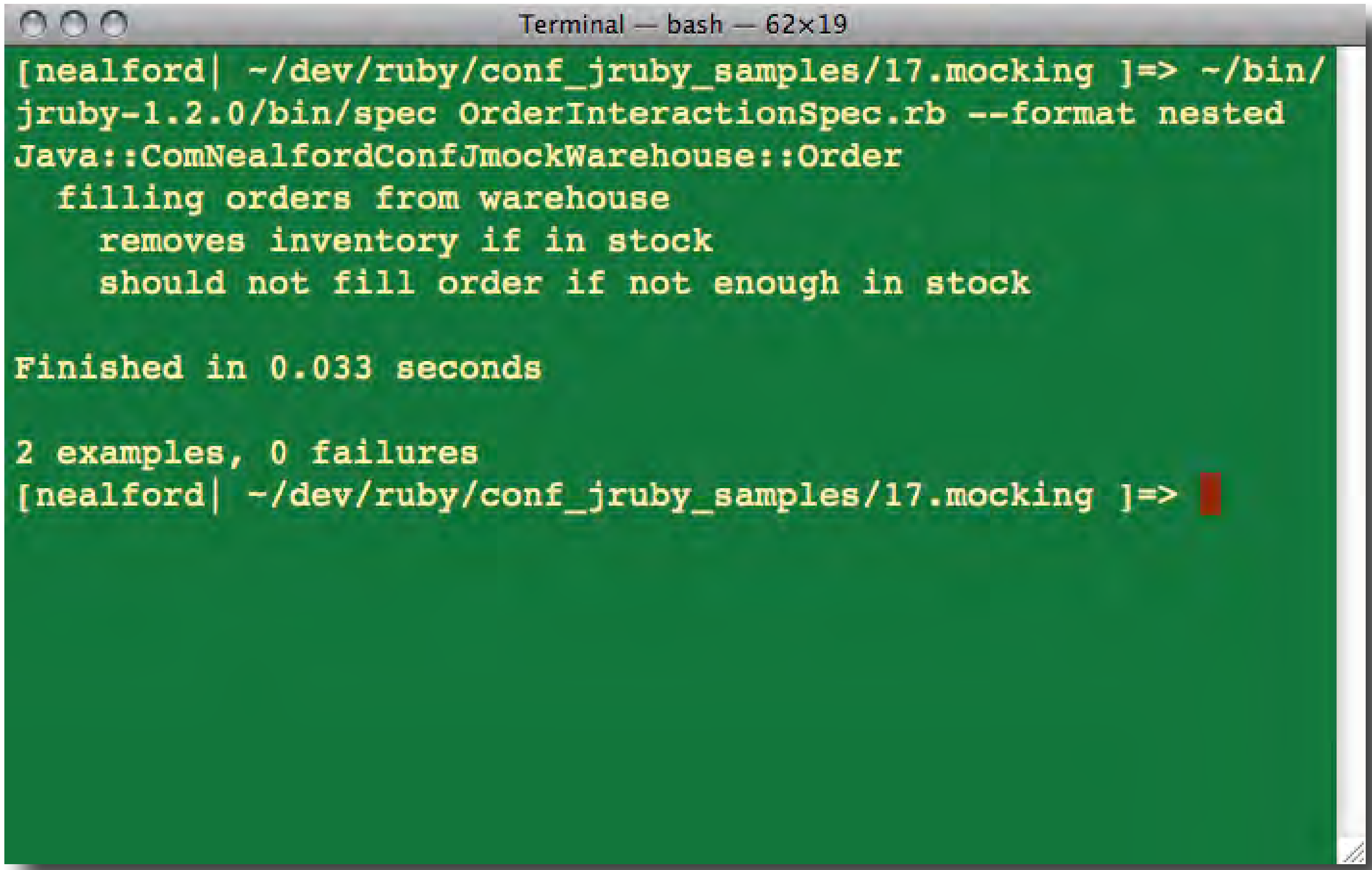
http://jtestr.codehaus.org/

specification

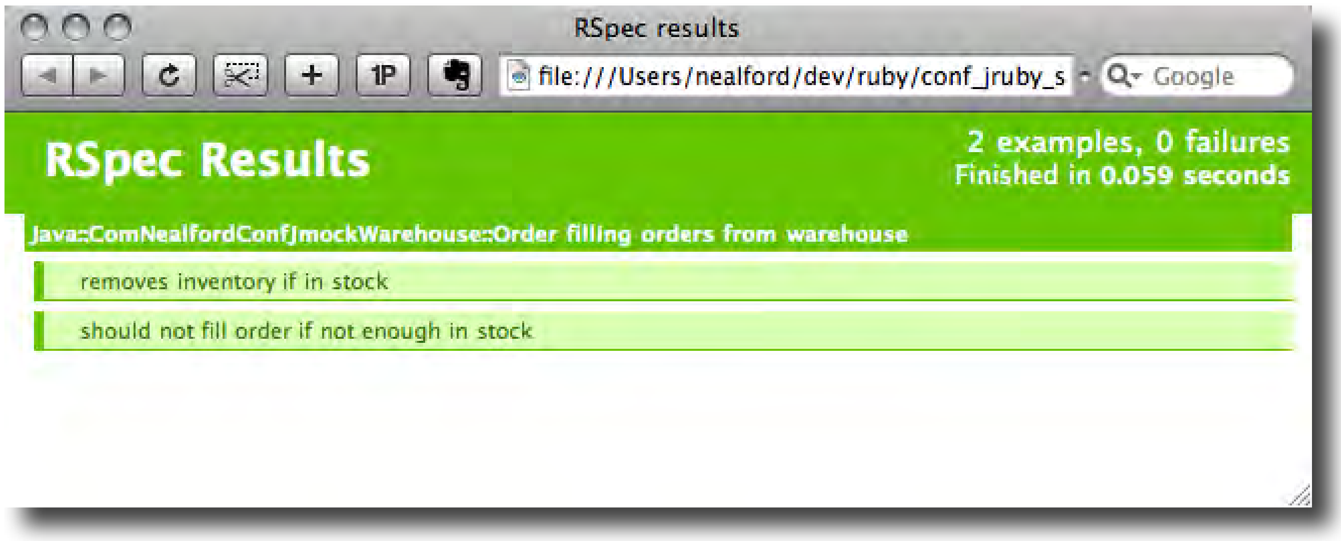
```
describe Order do
 context "filling orders from warehouse" do
   it "removes inventory if in stock" do
     order = OrderImpl.new(TALISKER, 50)
     warehouse = mock("warehouse")
     warehouse.should_receive(:hasInventory).
       with(TALISKER, 50).and_return(true)
     warehouse.should_receive(:remove).with(TALISKER,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

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

```ruby
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

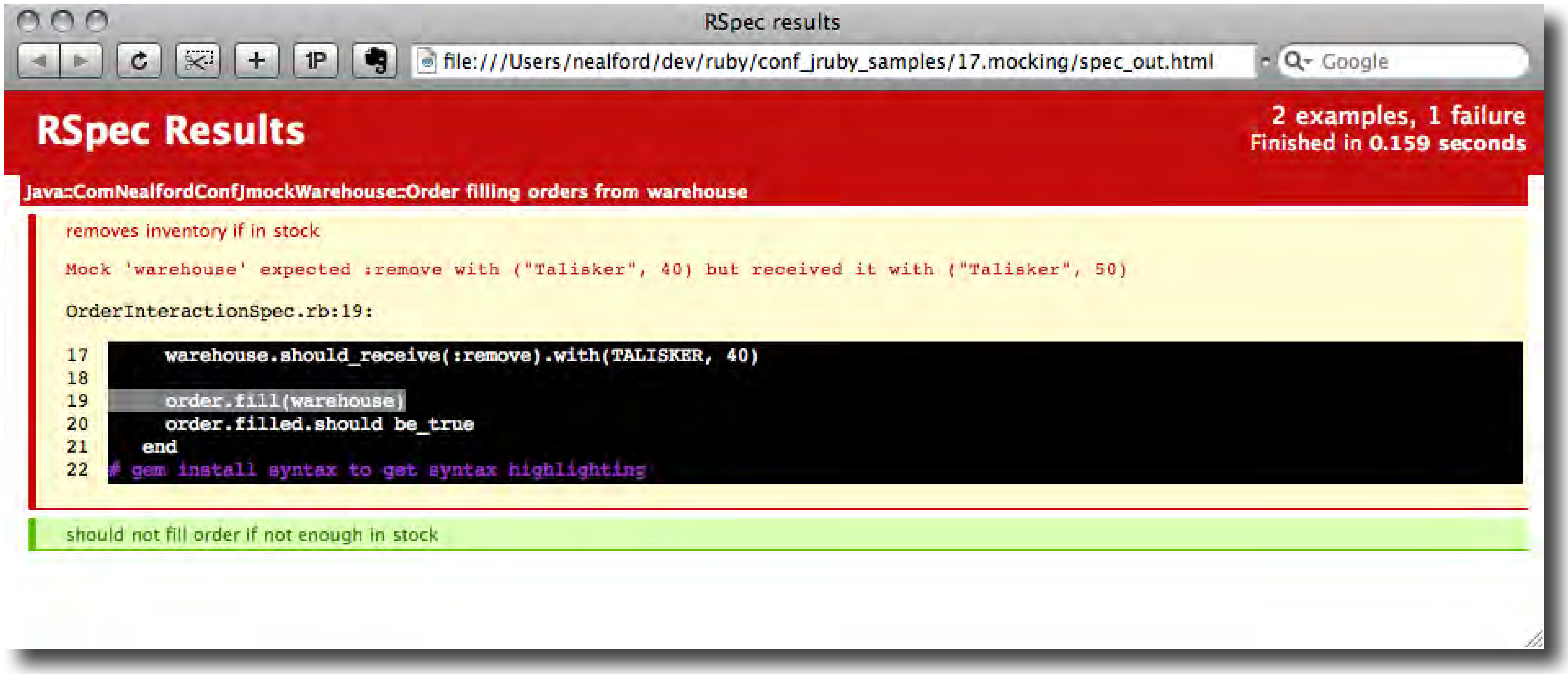


```
[nealford| ~/dev/ruby/conf_jruby_samples/17.mocking ]=> ~/bin/
jruby-1.2.0/bin/spec OrderInteractionSpec.rb --format nested
Java::ComNealfordConfJmockWarehouse::Order
  filling orders from warehouse
    removes inventory if in stock
    should not fill order if not enough in stock

Finished in 0.033 seconds

2 examples, 0 failures
[nealford| ~/dev/ruby/conf_jruby_samples/17.mocking ]=> 
```


RSpec results

file:///Users/nealford/dev/ruby/conf_jruby_s

```
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

```
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:

      warehouse.should_receive(:remove).with(TALISKER, 40)

      order.fill(warehouse)
      order.filled.should be_true
    end
  # gem install syntax to get syntax highlighting

  should not fill order if not enough in stock
```

```
Before do
  @calc = Calculator.new
end

After do
end

Given /I have entered (\d+) into the calculator/ do |n|
  @calc.push n.to_i
end

When /I press (\w+)/ do |op|
  @result = @calc.send op
end

Then /the result should be (.*) on the screen/ do |result|
  @result.should == result.to_f
end
```

Cucumber

Java

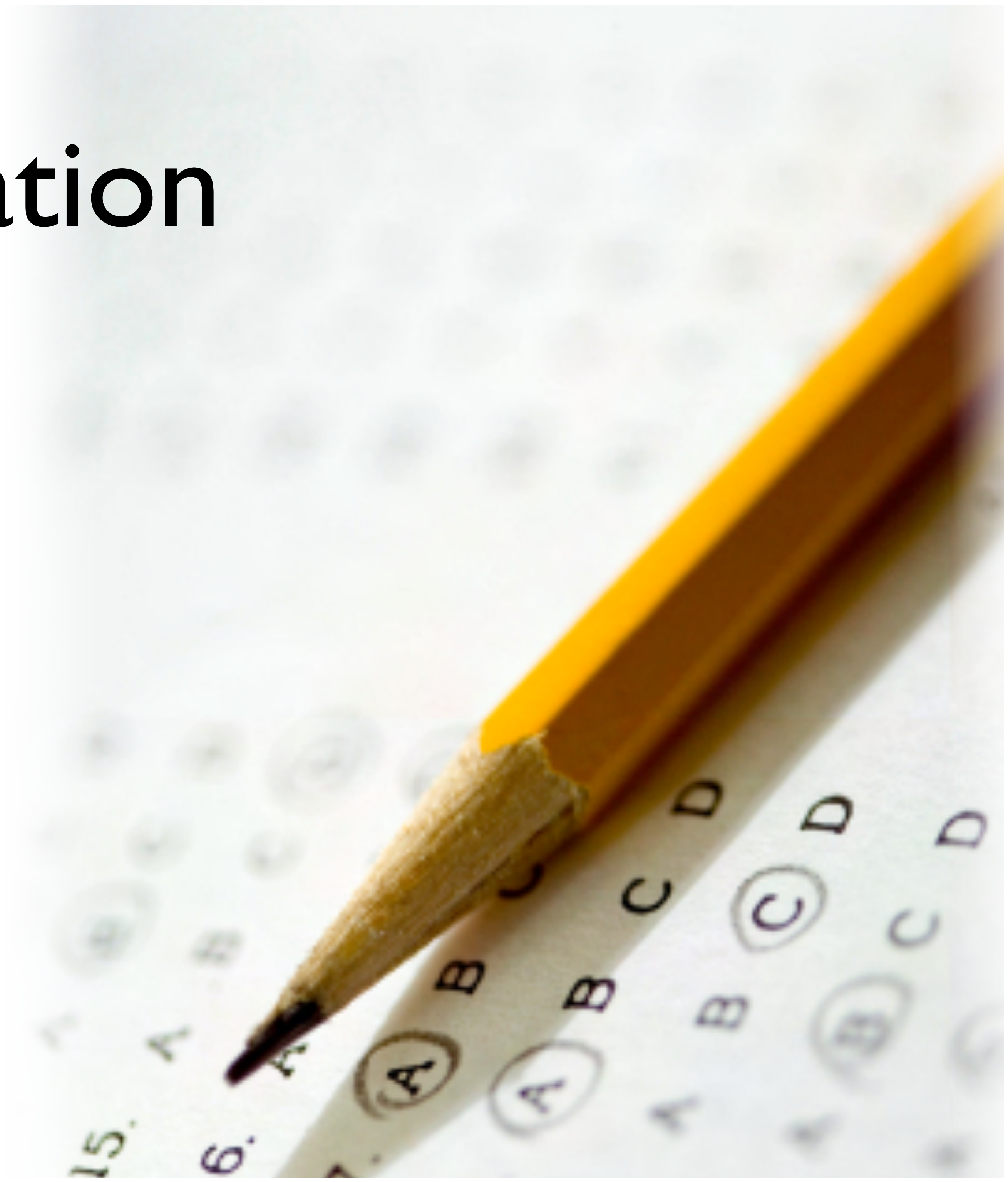
Ioke

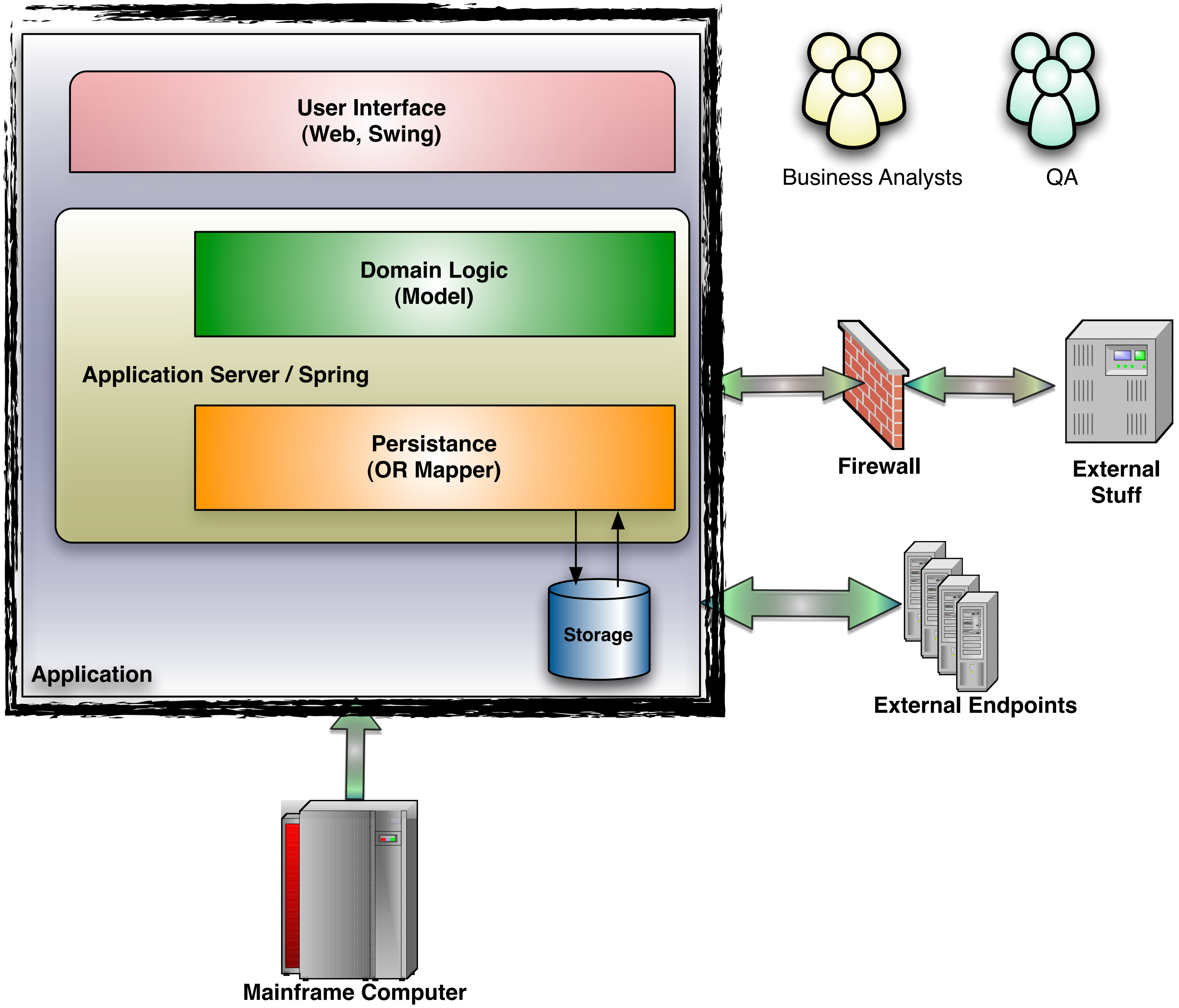
cuke4duke

Scala

JavaScript

integration




User Interface
(Web, Swing)
Domain Logic
(Model)
Application Server / Spring
Persistance
(OR Mapper)
Storage
Application
Business Analysts
QA
Firewall
External
Stuff
External Endpoints
Mainframe Computer

slow

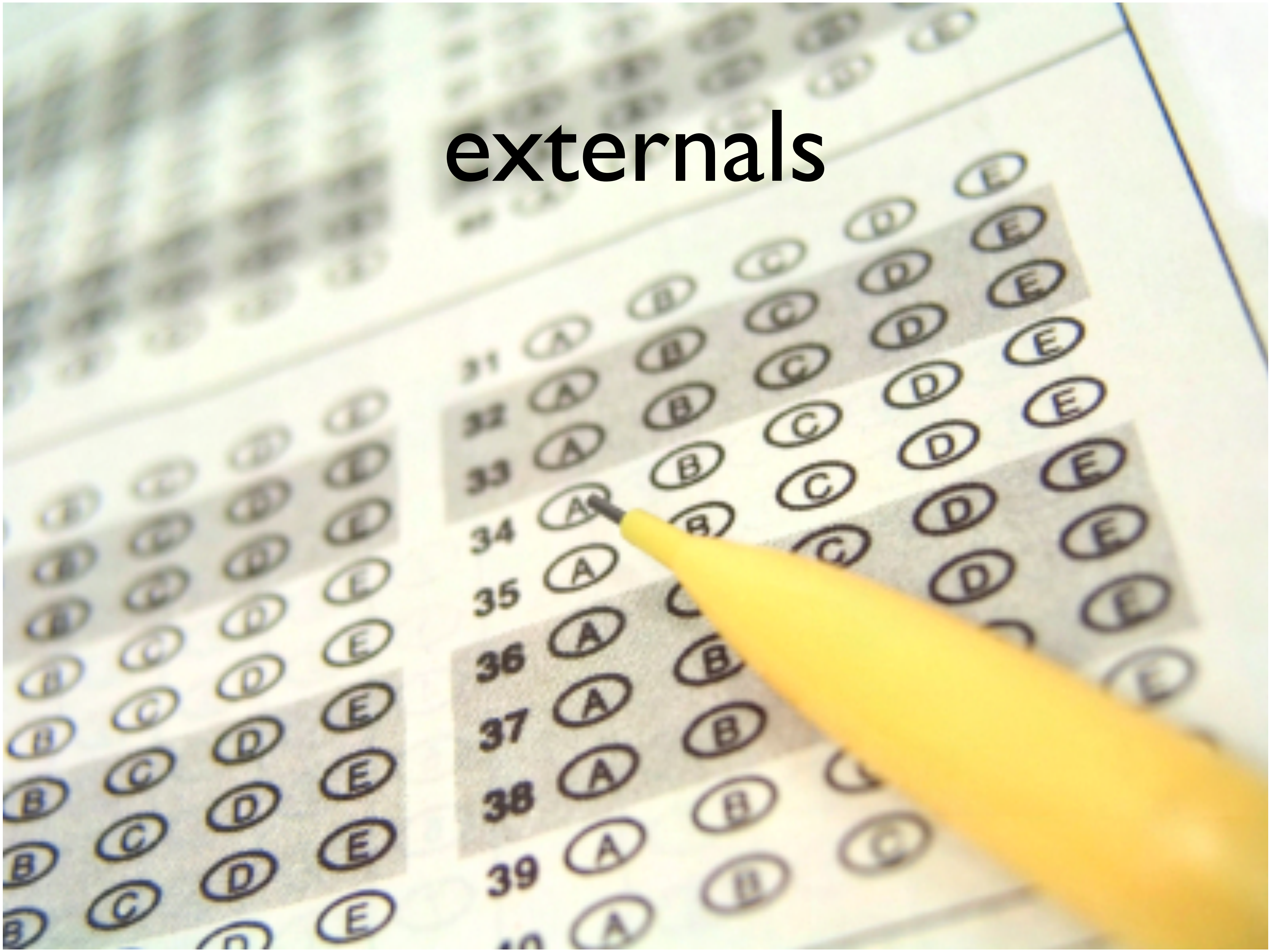
fragile

use unit testing tools (not UAT)

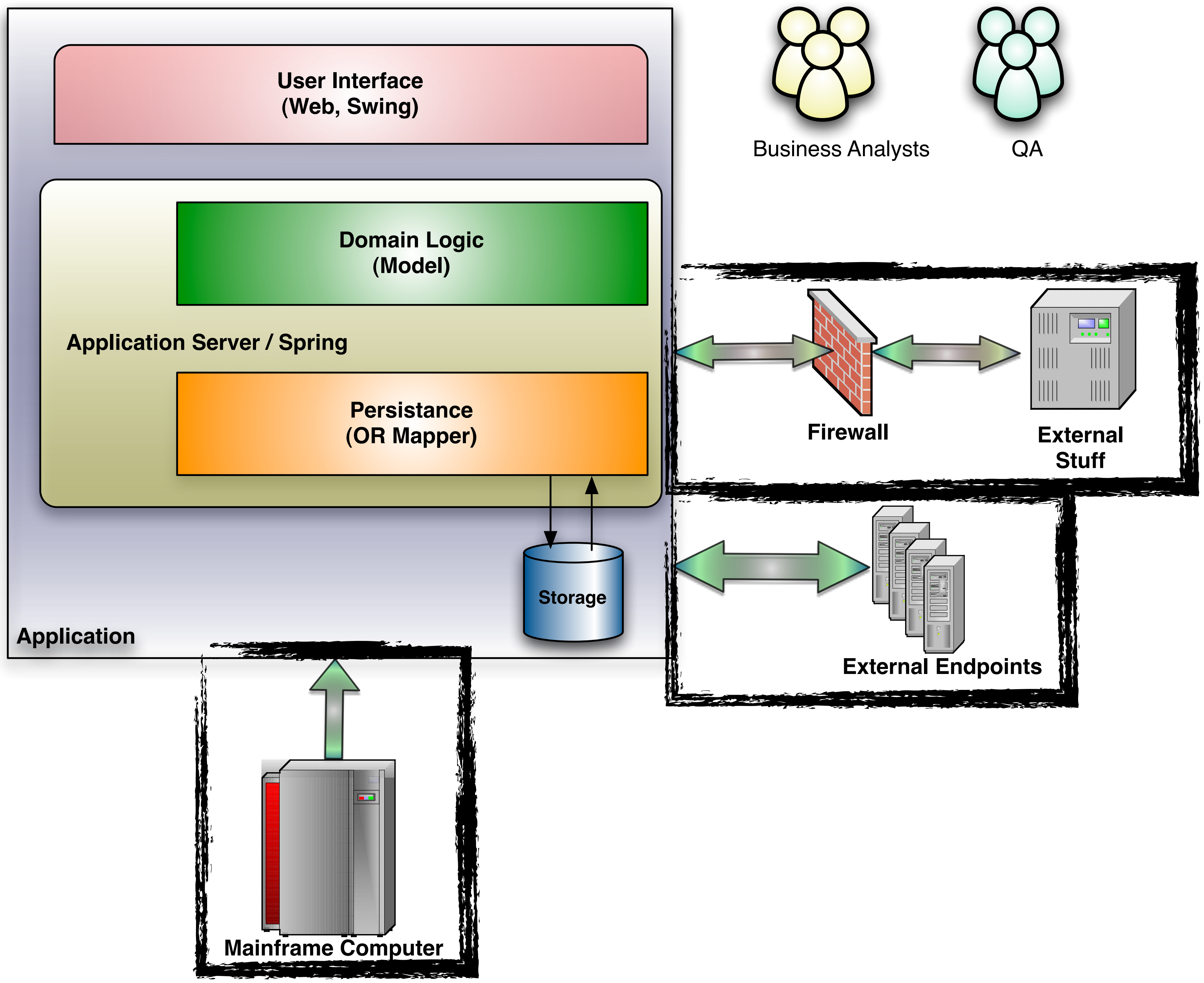
integration tests

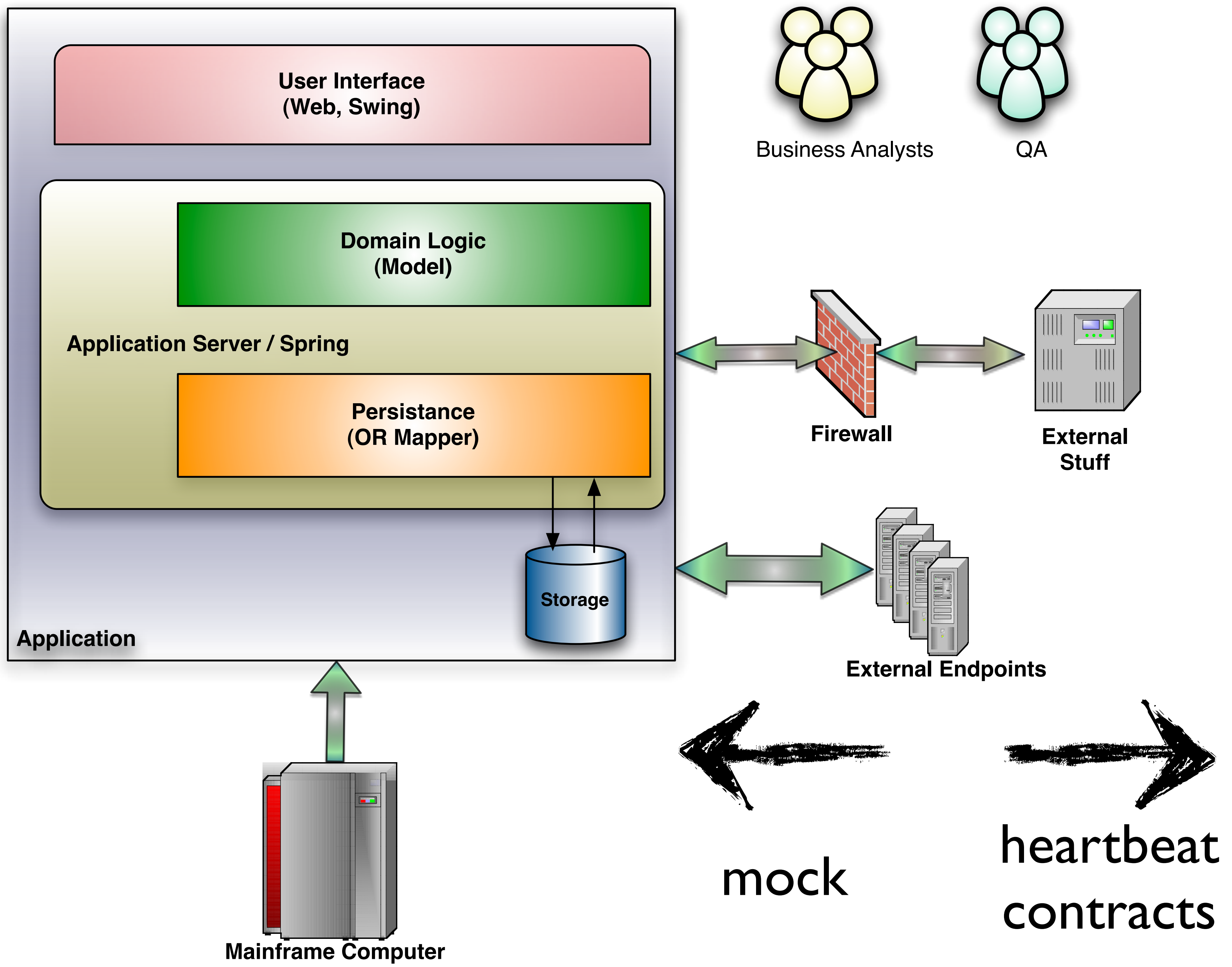
very coarse grained

as late in the process as possible



externals

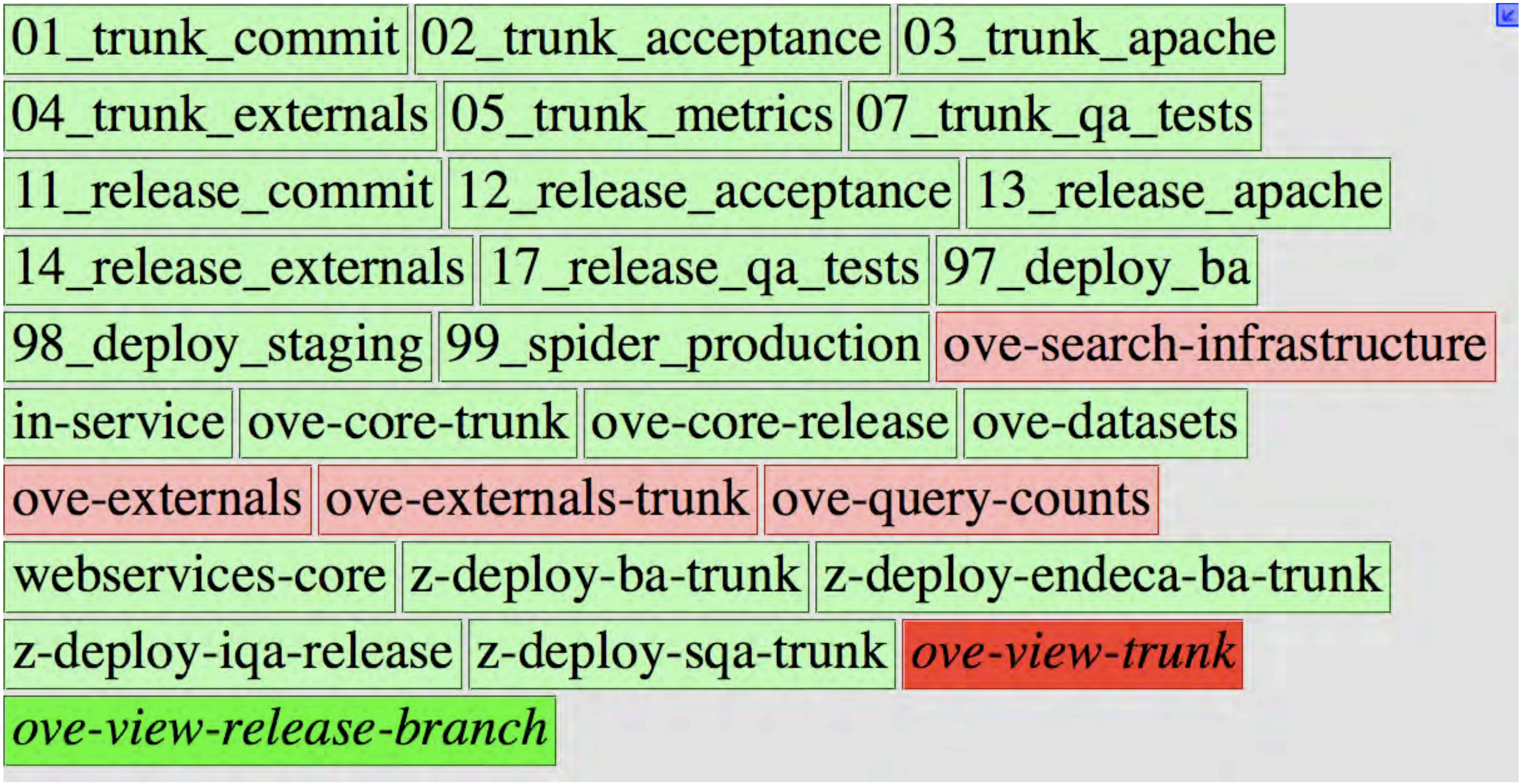

User Interface
(Web, Swing)
Business Analysts
QA
Domain Logic
(Model)
Application Server / Spring
Firewall
External
Stuff
Persistance
(OR Mapper)
Storage
External Endpoints
Application
Mainframe Computer



mock

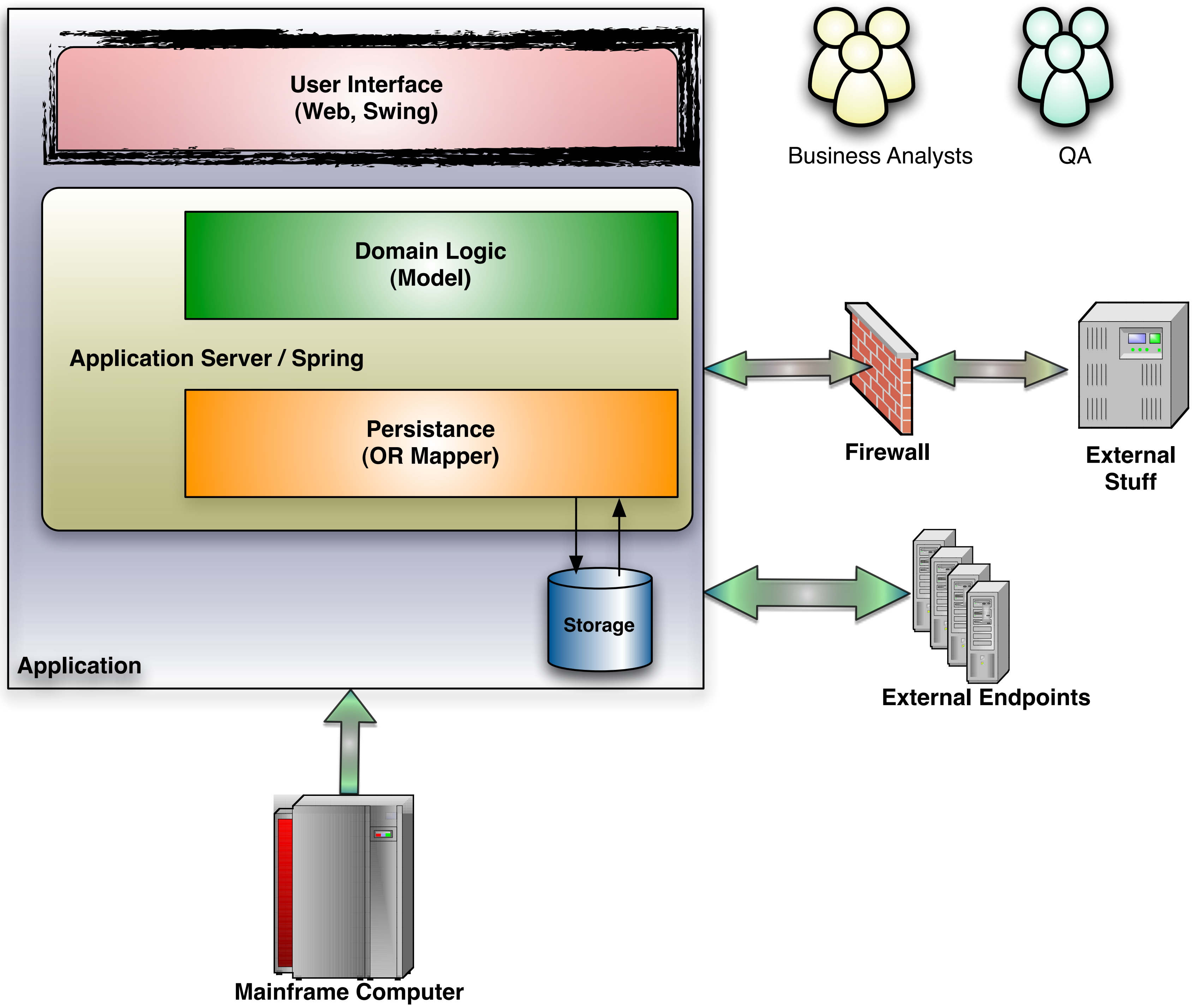
heartbeat
contracts

externals

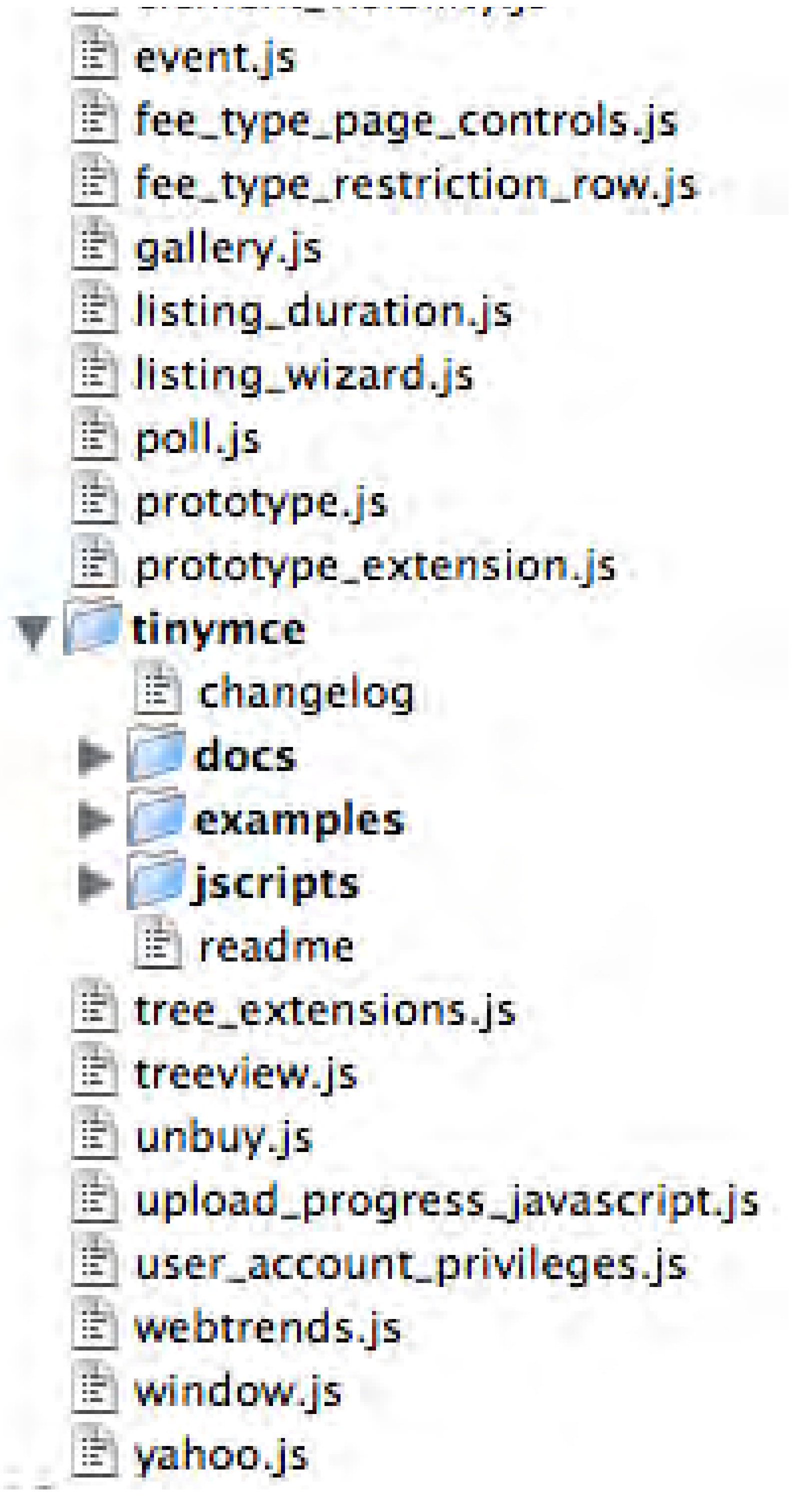

01_trunk_commit 02_trunk_acceptance 03_trunk_apache
04_trunk_externals 05_trunk_metrics 07_trunk_qa_tests
11_release_commit 12_release_acceptance 13_release_apache
14_release_externals 17_release_qa_tests 97_deploy_ba
98_deploy_staging 99_spider_production ove-search-infrastructure
in-service ove-core-trunk ove-core-release ove-datasets
ove-externals ove-externals-trunk ove-query-counts
webservices-core z-deploy-ba-trunk z-deploy-endeca-ba-trunk
z-deploy-iqa-release z-deploy-sqa-trunk *ove-view-trunk*
ove-view-release-branch

http://github.com/qxjit/cc_board/

user interface


User Interface
(Web, Swing)
Domain Logic
(Model)
Application Server / Spring
Persistance
(OR Mapper)
Storage
Application
Business Analysts
QA
Firewall
External
Stuff
External Endpoints
Mainframe Computer

JavaScript is real code!



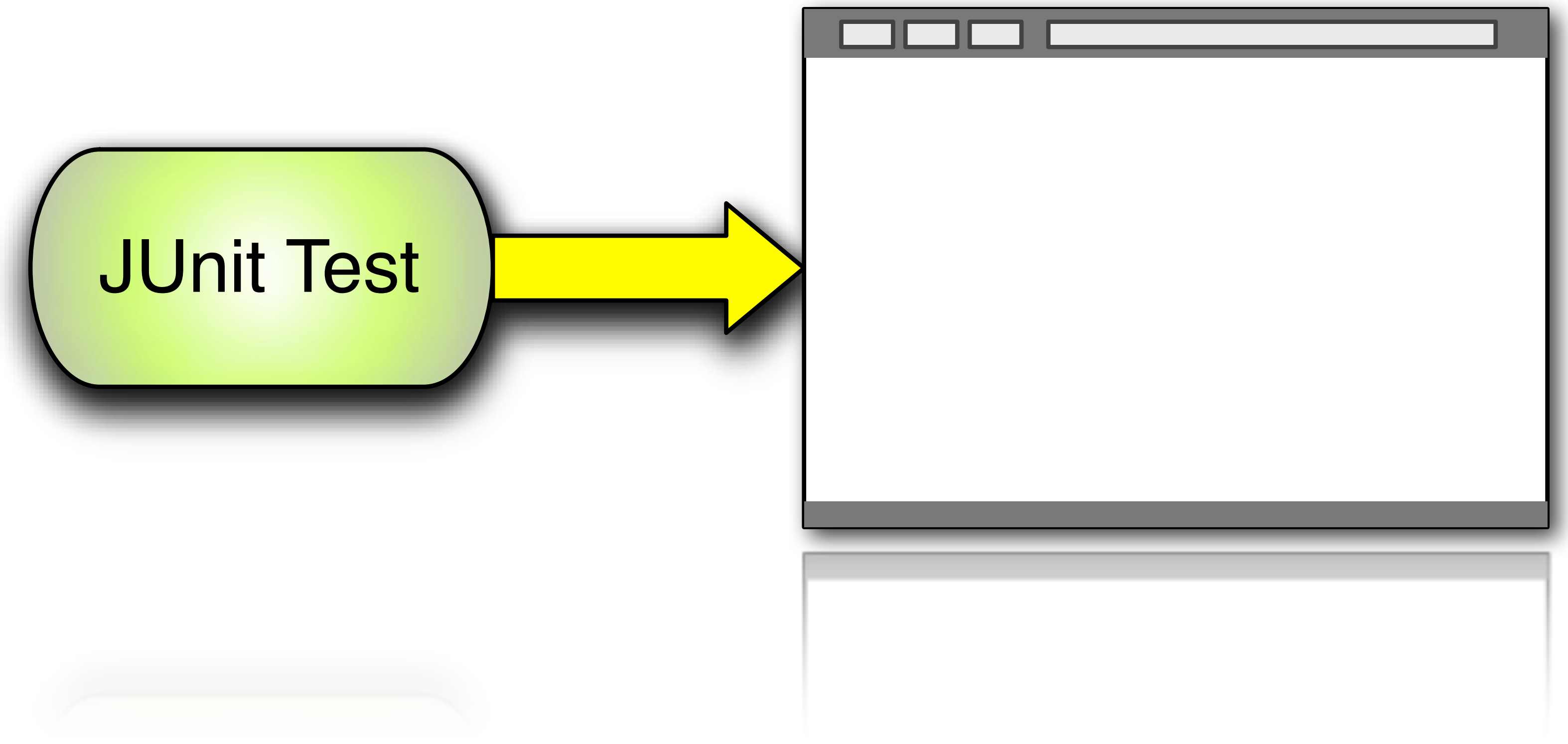
event.js
fee_type_page_controls.js
fee_type_restriction_row.js
gallery.js
listing_duration.js
listing_wizard.js
poll.js
prototype.js
prototype_extension.js
tinymce
changelog
docs
examples
jscripts
readme
tree_extensions.js
treeview.js
unbuy.js
upload_progress_javascript.js
user_account_privileges.js
webtrends.js
window.js
yahoo.js

```
function getFactorsFor(theNum) {
  if (theNum < 2)
    return 0;
  var listOfFactors = new Array();
  if (theNum == 2) {
    listOfFactors[0] = 1;
    return listOfFactors;
  }
  listOfFactors[0] = 1;
  listOfFactors[1] = theNum;
  var index = 2;
  for (i = 2; i < Math.sqrt(theNum) + 1; i++)
      if (theNum % i == 0) {
        var addIt = true;
        for (j = 0; j < listOfFactors.length; j++)
            if (listOfFactors[j] == i) {
              addIt = false;
              break;
            }
          if (addIt) {
            listOfFactors[index++] = i;
            if (i != theNum / i)
              listOfFactors[index++] = theNum / i;
          }
      }
  return listOfFactors;
}
```

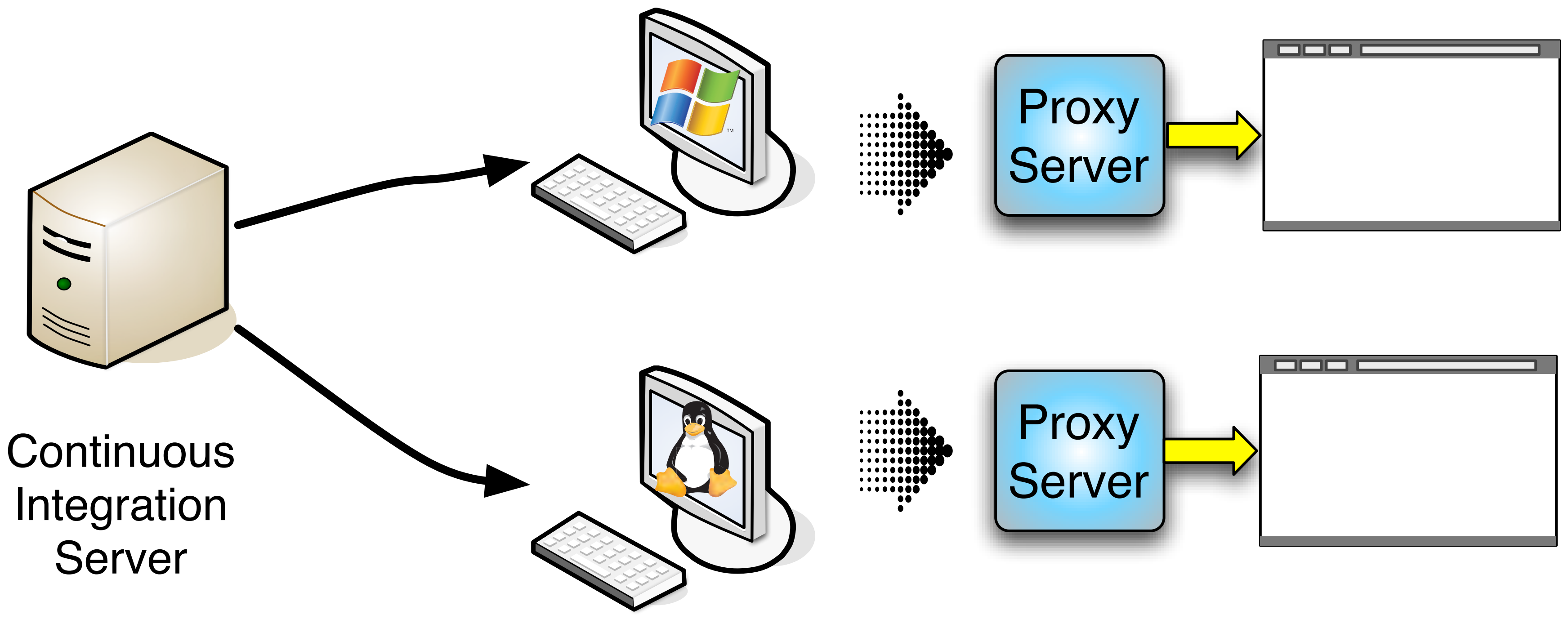
```
function sumOfFactors(num) {
  var sum = 0;
  var factorsOfNum = getFactorsFor(num);
  for (i = 0; i < factorsOfNum.length; i++) {
    sum += factorsOfNum[i];
  }
  return sum;
}

function isPerfect(number) {
  return sumOfFactors(number) - number == number;
}
```

```
function test_Proper_factors_for_abundant_number() {
  var expected = new Array(1, 12, 2, 6, 3, 4);
  var returnedFactors = getFactorsFor(12);
  assertEquals("length is correct", expected.length, returnedFactors.length);
  for (i = 0; i < expected.length; i++)
    assertEquals("array match failed", expected[i], returnedFactors[i]);
}

function test_Proper_factors_for_prime_number() {
  var expected = new Array(1, 17);
  var returnedFactors = getFactorsFor(17);
  assertEquals("length is correct", expected.length, returnedFactors.length);
  for (i = 0; i < expected.length; i++)
    assertEquals("array match failed", expected[i], returnedFactors[i]);
}

function test_Proper_factors_for_deficient_number() {
  var expected = new Array(1, 9, 3);
  var returnedFactors = getFactorsFor(9);
  assertEquals("length is correct", expected.length, returnedFactors.length);
  for (i = 0; i < expected.length; i++)
    assertEquals("array match failed", expected[i], returnedFactors[i]);
}
```

stand-alone test



distributed test



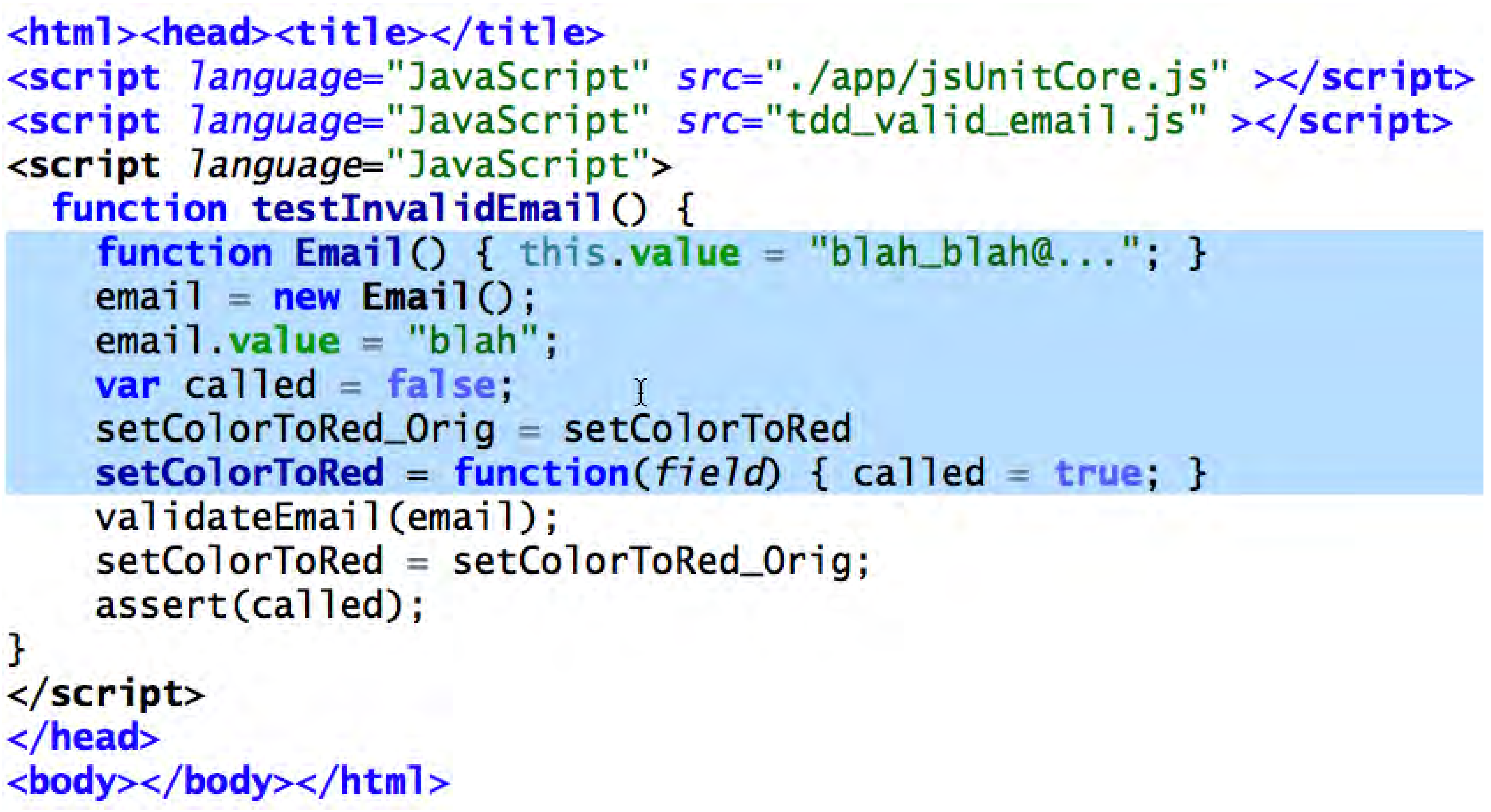
mocking javascript

```
function validateEmail(field) {
    if (field.value.match(/[A-Za-z]+_[A-Za-z]+@[A-Za-z]+\.org/) == null) {
        new Effect.Highlight(field.id, {startcolor:'#FF0000', endcolor:'#FFFFFF'});
    }
}
```

step 1: know what you are testing

```
function validateEmail(field) {
    if (field.value.match(/[A-Za-z]+_[A-Za-z]+@[A-Za-z]+\.org/) == null) {
        setColorToRed(field);
    }
}

function setColorToRed(field) {
    new Effect.Highlight(field.id, {startcolor:'#FF0000', endcolor:'#FFFFFF'});
}
```

step 2: don’t test what you don’t have to



```
<html><head><title></title>
<script language="JavaScript" src="./app/jsUnitCore.js" ></script>
<script language="JavaScript" src="tdd_valid_email.js" ></script>
<script language="JavaScript">
  function testInvalidEmail() {
    function Email() { this.value = "blah_blah@..."; }
    email = new Email();
    email.value = "blah";
    var called = false;
    setColorToRed_Orig = setColorToRed
    setColorToRed = function(field) { called = true; }
    validateEmail(email);
    setColorToRed = setColorToRed_Orig;
    assert(called);
}
</script>
</head>
<body></body></html>
```

headless JavaScript testing

blue-ridge

http://github.com/relevance/blue-ridge

```
require("spec_helper.js");
require("../../public/javascripts/application.js");

Screw.Unit(function() {
  describe("Your application javascript", function() {
    it("does something", function() {
      expect("hello").to(equal, "hello");
    });

    it("accesses the DOM from fixtures/application.html", function() {
      expect($('.select_me').length).to(equal, 2);
    });
  });
});
```

pros:

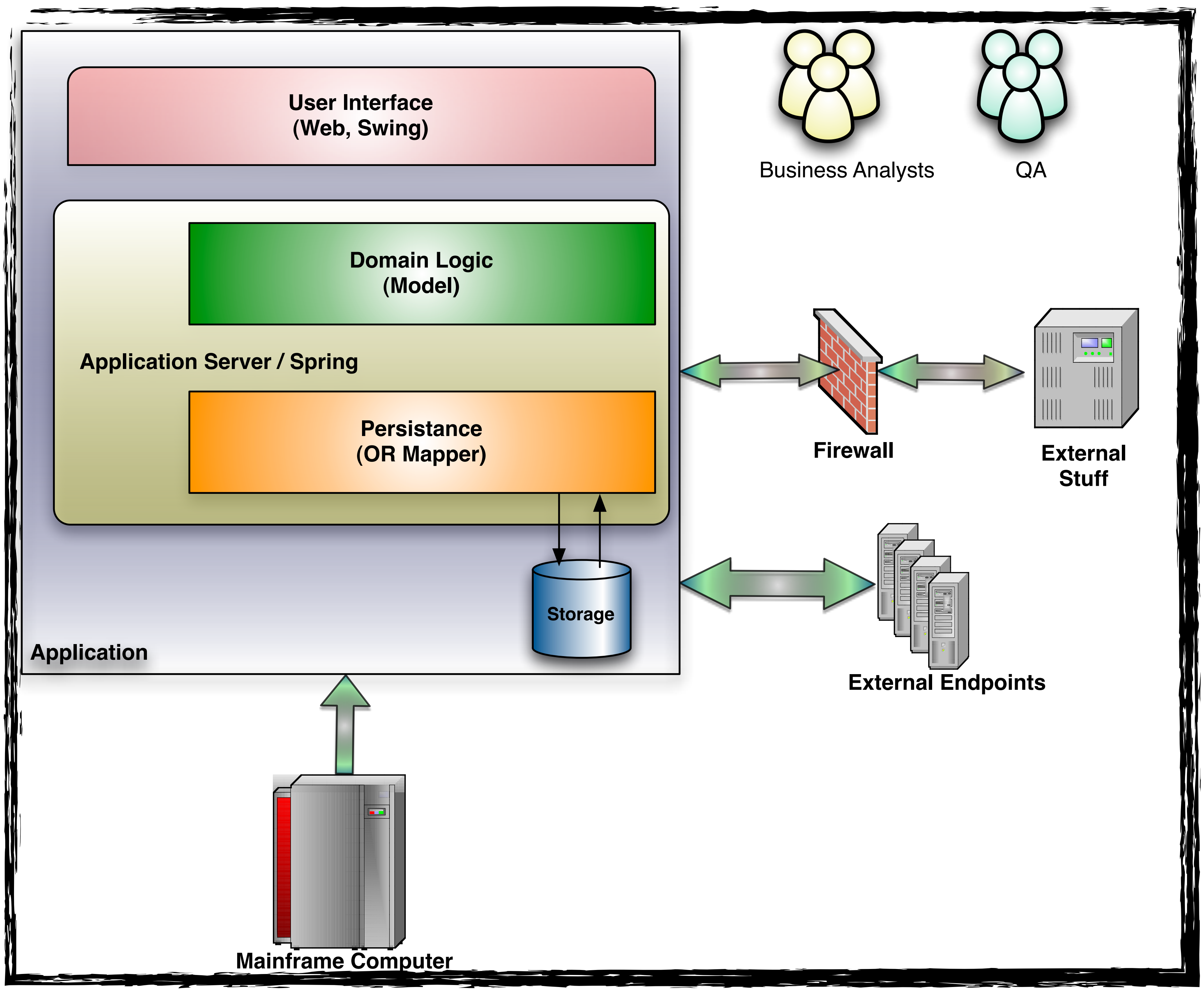
- fast!
- easier to continually integrate

headless?

cons:

- not running in a browser
- only as good as your mocks

user acceptance


User Interface
(Web, Swing)
Business Analysts
QA
Domain Logic
(Model)
Application Server / Spring
Firewall
External
Stuff
Persistance
(OR Mapper)
Storage
External Endpoints
Application
Mainframe Computer

user acceptance tests

top to bottom, left to right: *everything!*

as late as possible in the development process http://seleniumhq.org/

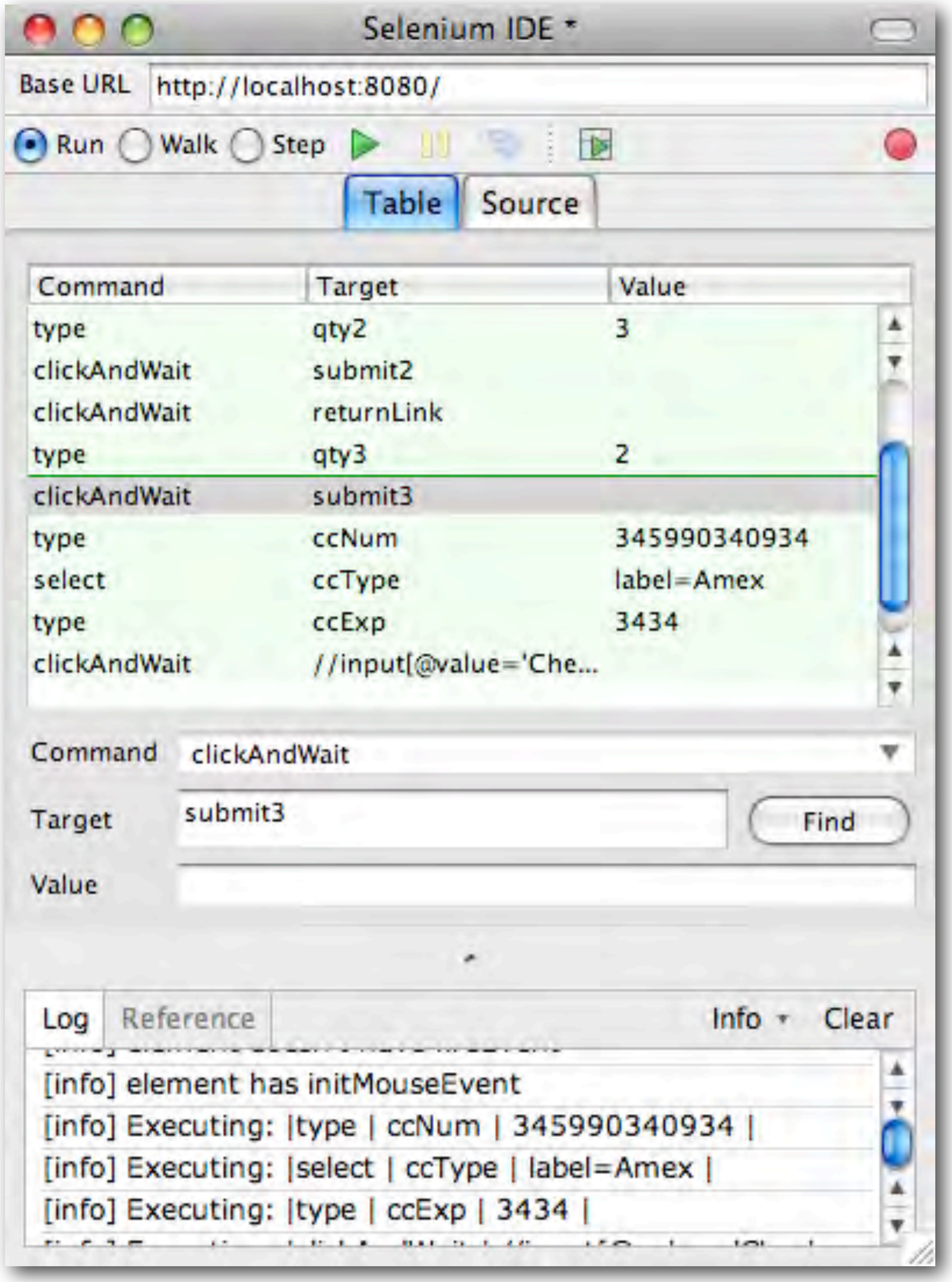
open source UAT tool for web applications

works in all browsers

for all types of web applications

side project Selenium IDE provides recorder

state-of-the-art UAT testing



Base URL: http://localhost:8080/

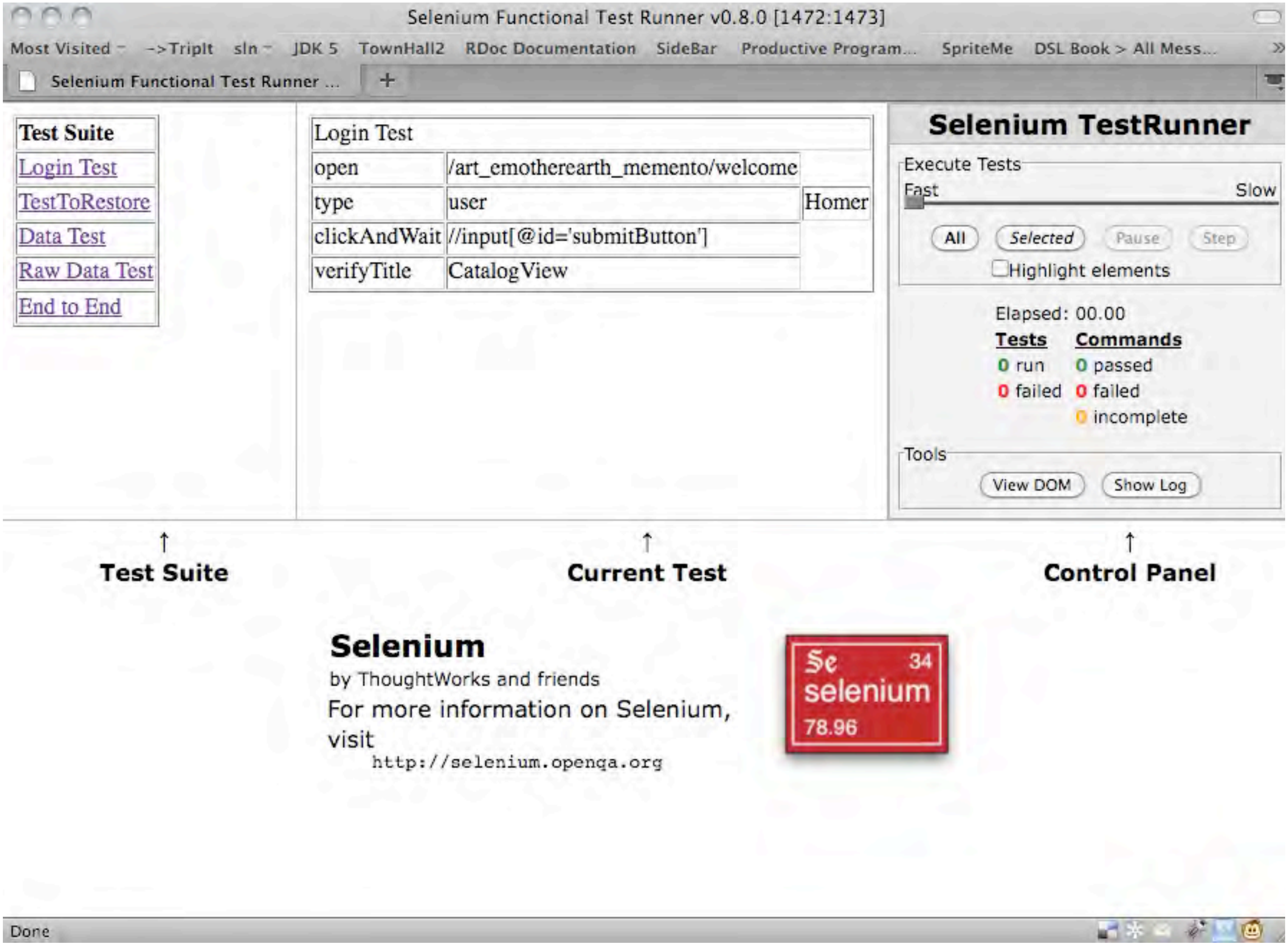
Command	Target	Value
type	qty2	3
clickAndWait	submit2	
clickAndWait	returnLink	
type	qty3	2
clickAndWait	submit3	
type	ccNum	345990340934
select	ccType	label=Amex
type	ccExp	3434
clickAndWait	//input[@value='Che...	

Command: clickAndWait

Target: submit3

Value:

```
[info] element has initMouseEvent
[info] Executing: |type | ccNum | 345990340934 |
[info] Executing: |select | ccType | label=Amex |
[info] Executing: |type | ccExp | 3434 |
```



Selenium Functional Test Runner v0.8.0 [1472:1473]

Test Suite
Login Test
TestToRestore
Data Test
Raw Data Test
End to End

Login Test		
open	/art_emotherearth_memento/welcome	
type	user	Homer
clickAndWait	//input[@id='submitButton']	
verifyTitle	CatalogView	

Selenium TestRunner

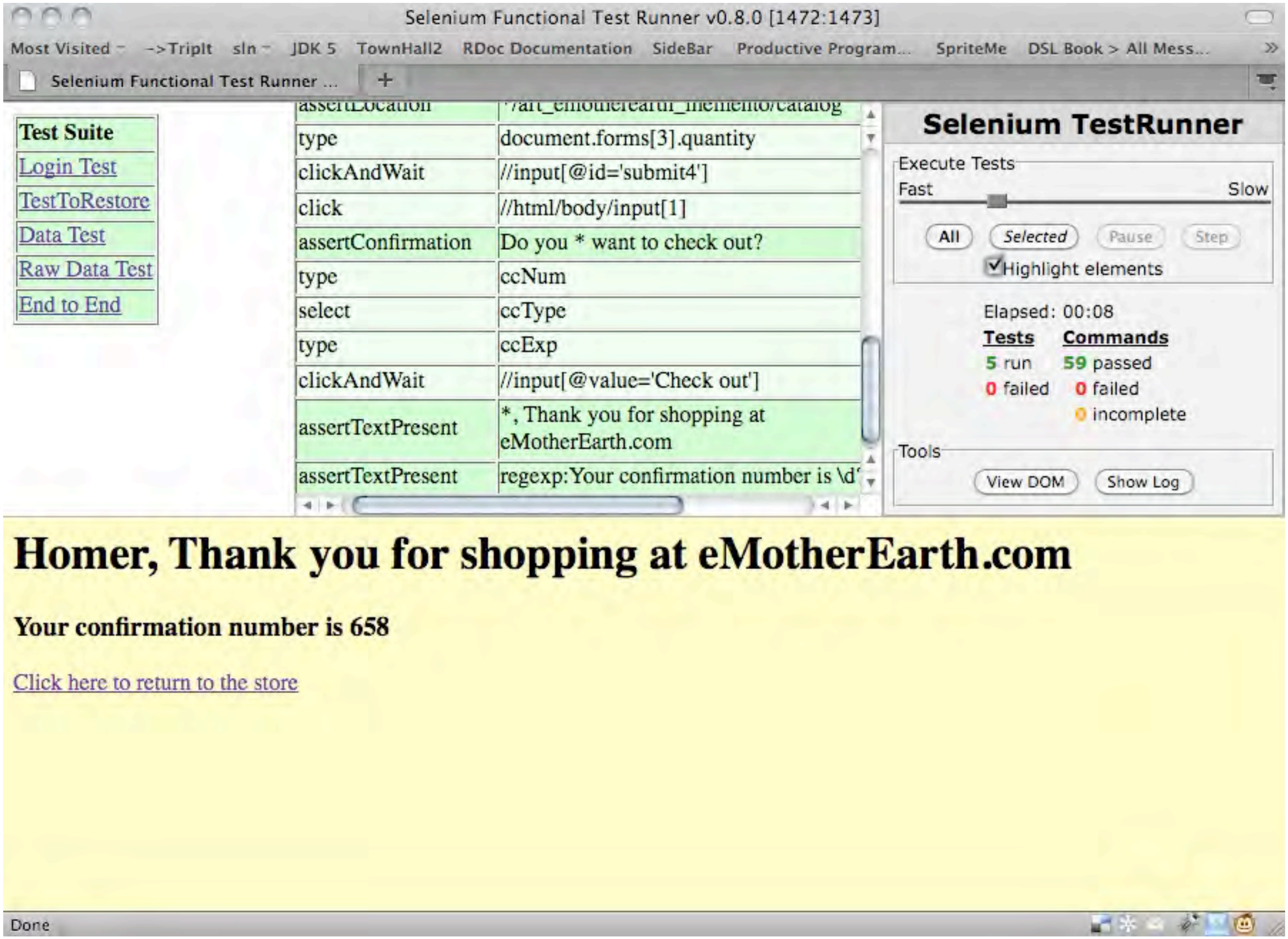
Elapsed: 00.00

Tests	Commands
0 run	0 passed
0 failed	0 failed
	0 incomplete

Test Suite

Current Test

Control Panel



Test Suite
Login Test
TestToRestore
Data Test
Raw Data Test
End to End

assertLocation	*/art_emotherearth_memento/catalog
type	document.forms[3].quantity
clickAndWait	//input[@id='submit4']
click	//html/body/input[1]
assertConfirmation	Do you * want to check out?
type	ccNum
select	ccType
type	ccExp
clickAndWait	//input[@value='Check out']
assertTextPresent	*, Thank you for shopping at eMotherEarth.com
assertTextPresent	regexp:Your confirmation number is \d

Selenium TestRunner

✓ Highlight elements

Elapsed: 00:08

Tests	Commands
5 run	59 passed
0 failed	0 failed
	0 incomplete

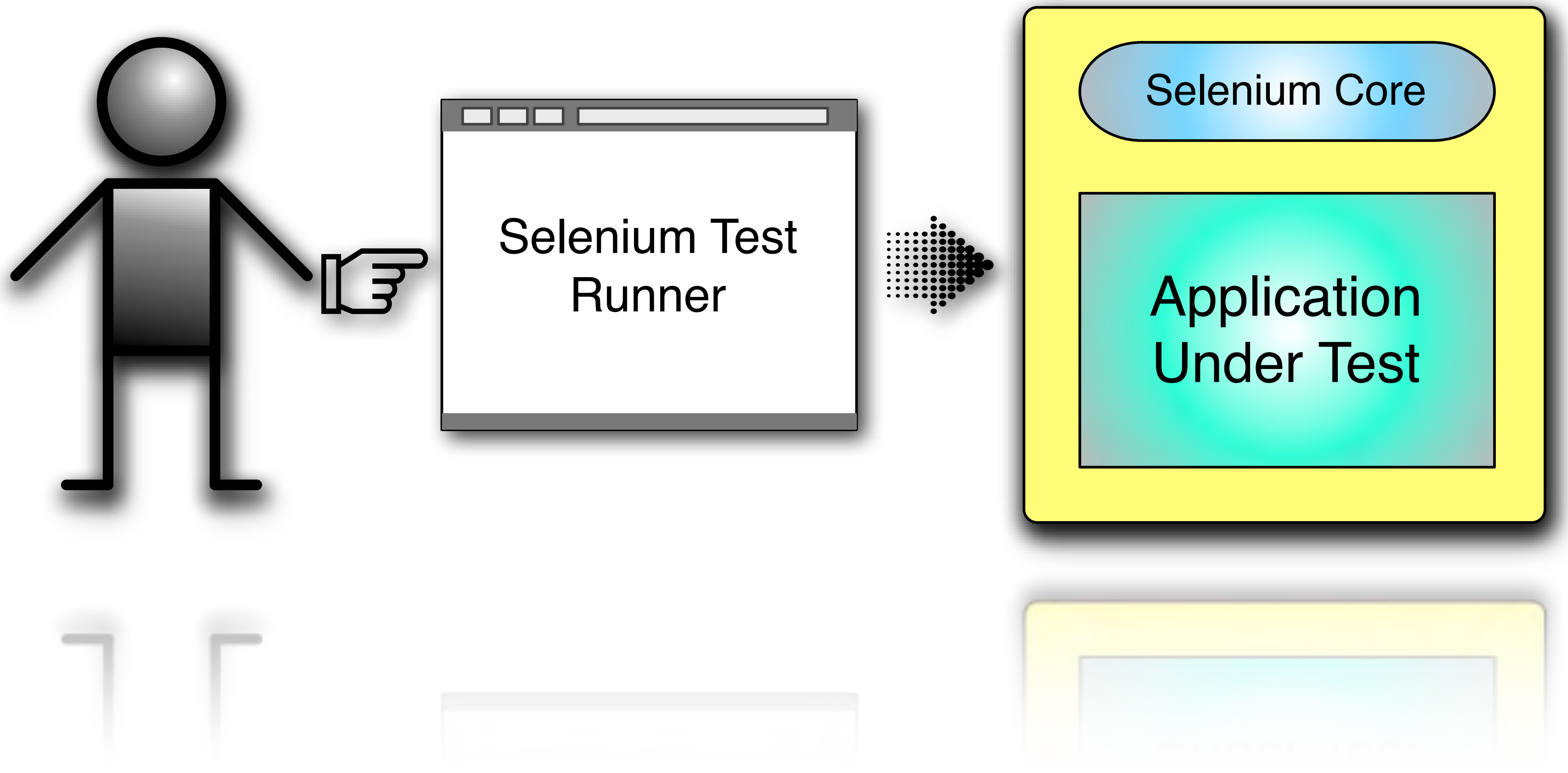
Homer, Thank you for shopping at eMotherEarth.com

Your confirmation number is 658

Click here to return to the store

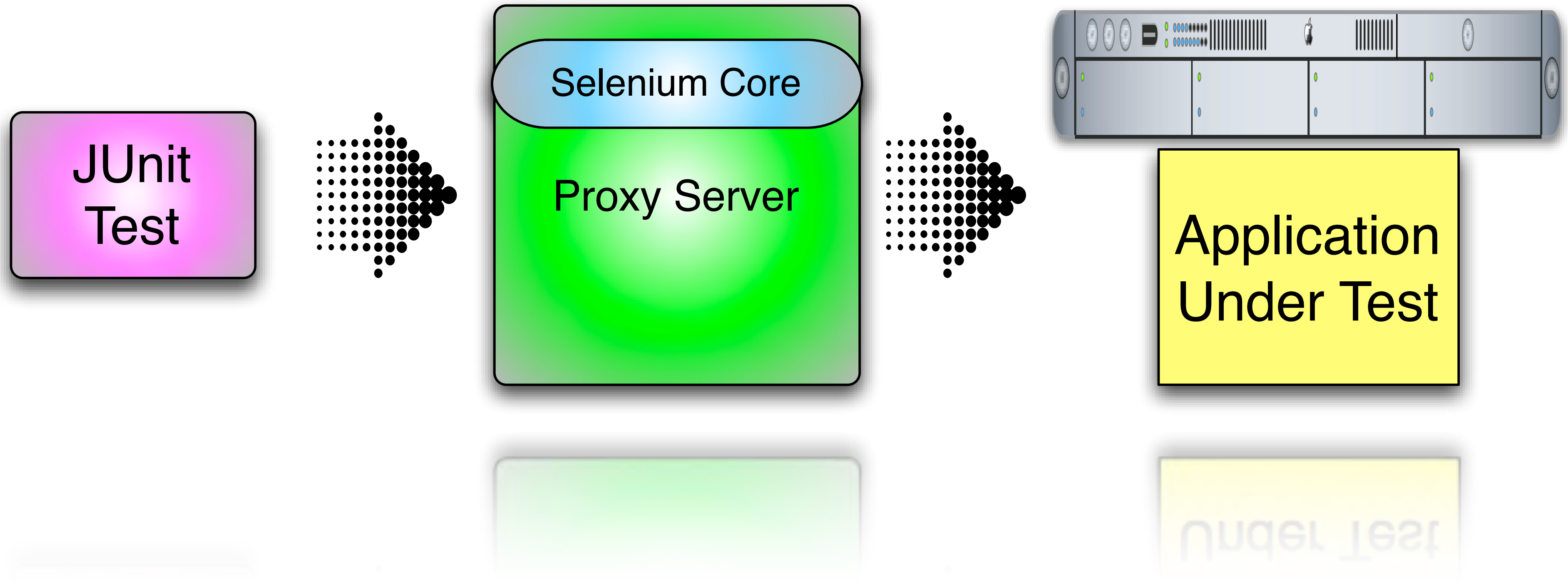
Done

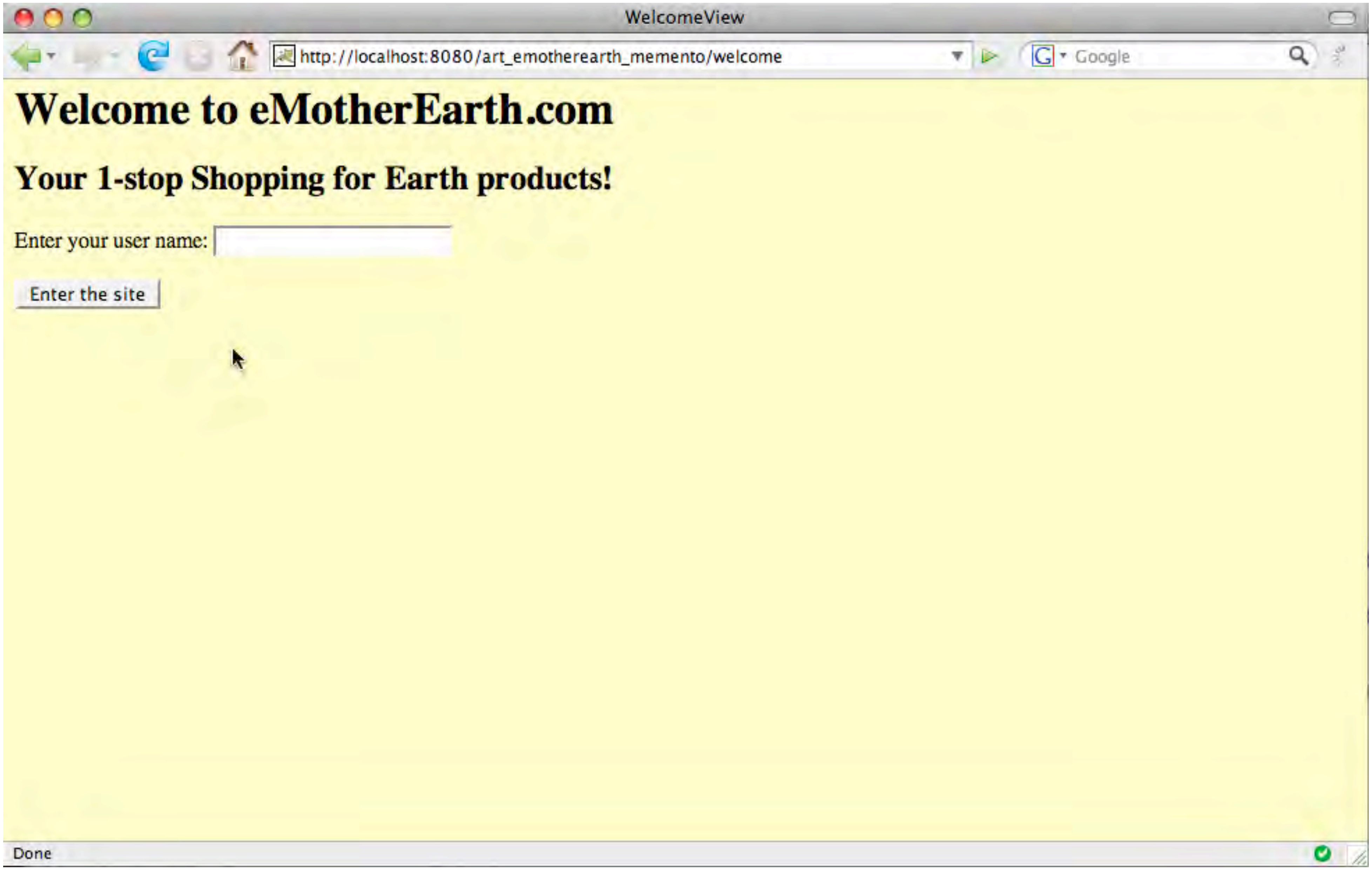
test runner mode



New Test		
open	/art_emotherearth_memento/welcome	
type	userName	Homer
clickAndWait	submitButton	
type	qty2	3
clickAndWait	submit2	
clickAndWait	returnLink	
type	qty6	4
clickAndWait	submit6	
type	ccNum	234234234234
select	ccType	label=MC
type	ccExp	2323
clickAndWait	//input[@value='Check out']	

remote control





Welcome to eMotherEarth.com

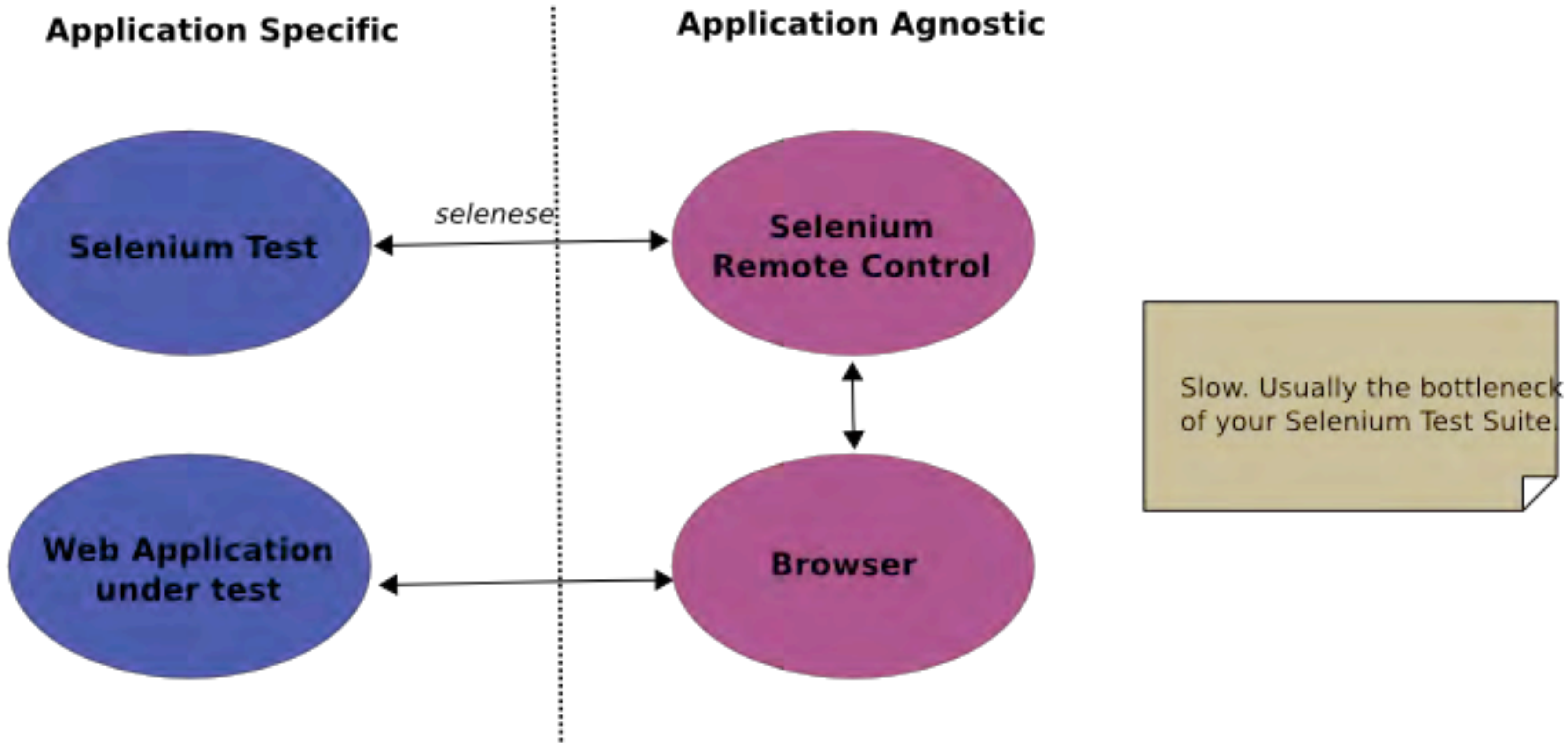
Your 1-stop Shopping for Earth products!

Enter your user name:

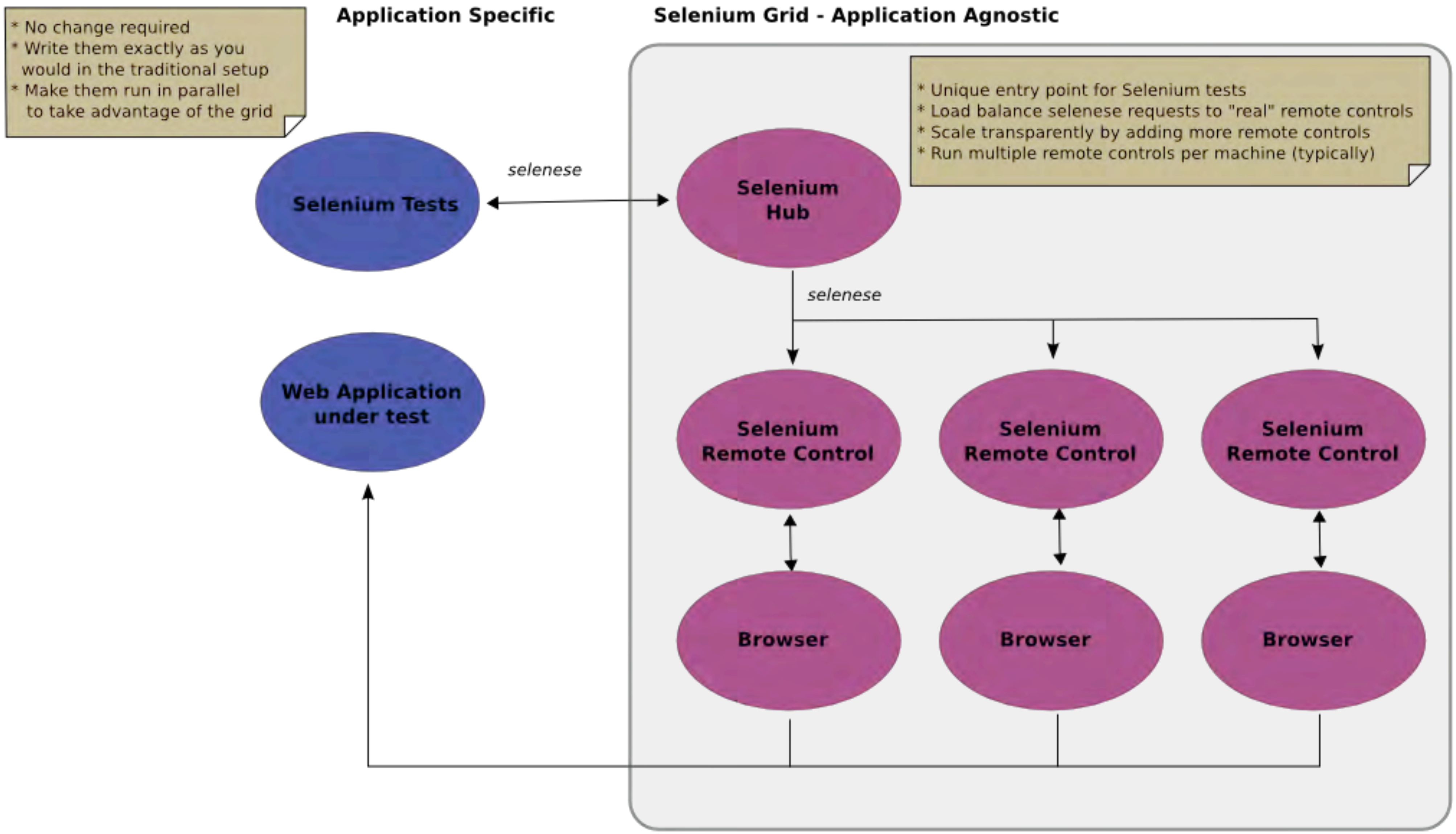
```java
public class NewTest extends SeleneseTestCase {
    public void testNew() throws Exception {
        selenium.open("/art_emotherearth_memento/welcome");
        selenium.type("userName", "Homer");
        selenium.click("submitButton");
        selenium.waitForPageToLoad("30000");
        selenium.type("qty2", "3");
        selenium.click("submit2");
        selenium.waitForPageToLoad("30000");
        selenium.click("returnLink");
        selenium.waitForPageToLoad("30000");
        selenium.type("qty6", "4");
        selenium.click("submit6");
        selenium.waitForPageToLoad("30000");
        selenium.type("ccNum", "234234234234");
        selenium.select("ccType", "label=MC");
        selenium.type("ccExp", "2323");
        selenium.click("//input[@value='Check out']");
        selenium.waitForPageToLoad("30000");
    }
}
```

```ruby
class NewTest < Test::Unit::TestCase
  def setup
    @verification_errors = []
    if $selenium
      @selenium = $selenium
    else
      @selenium = Selenium::SeleneseInterpreter.new(
          "localhost", 4444, "*firefox", "http://localhost:4444", 10000);
      @selenium.start
    end
    @selenium.set_context("test_new", "info")
  end

  def teardown
    @selenium.stop unless $selenium
    assert_equal [], @verification_errors
  end

  def test_new
    @selenium.open "/art_emotherearth_memento/welcome"
    @selenium.type "userName", "Homer"
    @selenium.click "submitButton"
    @selenium.wait_for_page_to_load "30000"
    @selenium.type "qty2", "3"
    @selenium.click "submit2"
    @selenium.wait_for_page_to_load "30000"
    @selenium.click "returnLink"
    @selenium.wait_for_page_to_load "30000"
    @selenium.type "qty6", "4"
    @selenium.click "submit6"
    @selenium.wait_for_page_to_load "30000"
    @selenium.type "ccNum", "234234234234"
    @selenium.select "ccType", "label=MC"
    @selenium.type "ccExp", "2323"
    @selenium.click "//input[@value='Check out']"
    @selenium.wait_for_page_to_load "30000"
  end
end
```

Traditional Selenium Setup



Selenium Grid Setup



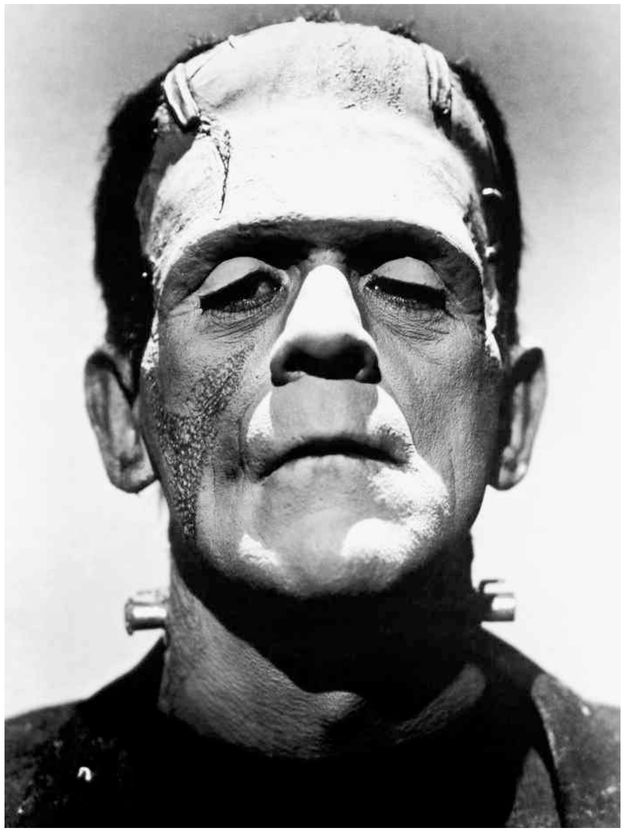
* No change required
* Write them exactly as you would in the traditional setup
* Make them run in parallel to take advantage of the grid
Application Specific
Selenium Grid - Application Agnostic
* Unique entry point for Selenium tests
* Load balance selenese requests to "real" remote controls
* Scale transparently by adding more remote controls
* Run multiple remote controls per machine (typically)
Selenium Tests
selenese
Selenium Hub
selenese
Web Application under test
Selenium Remote Control
Selenium Remote Control
Selenium Remote Control
Browser
Browser
Browser

why? (just kidding — it isn't your fault)

record / playback

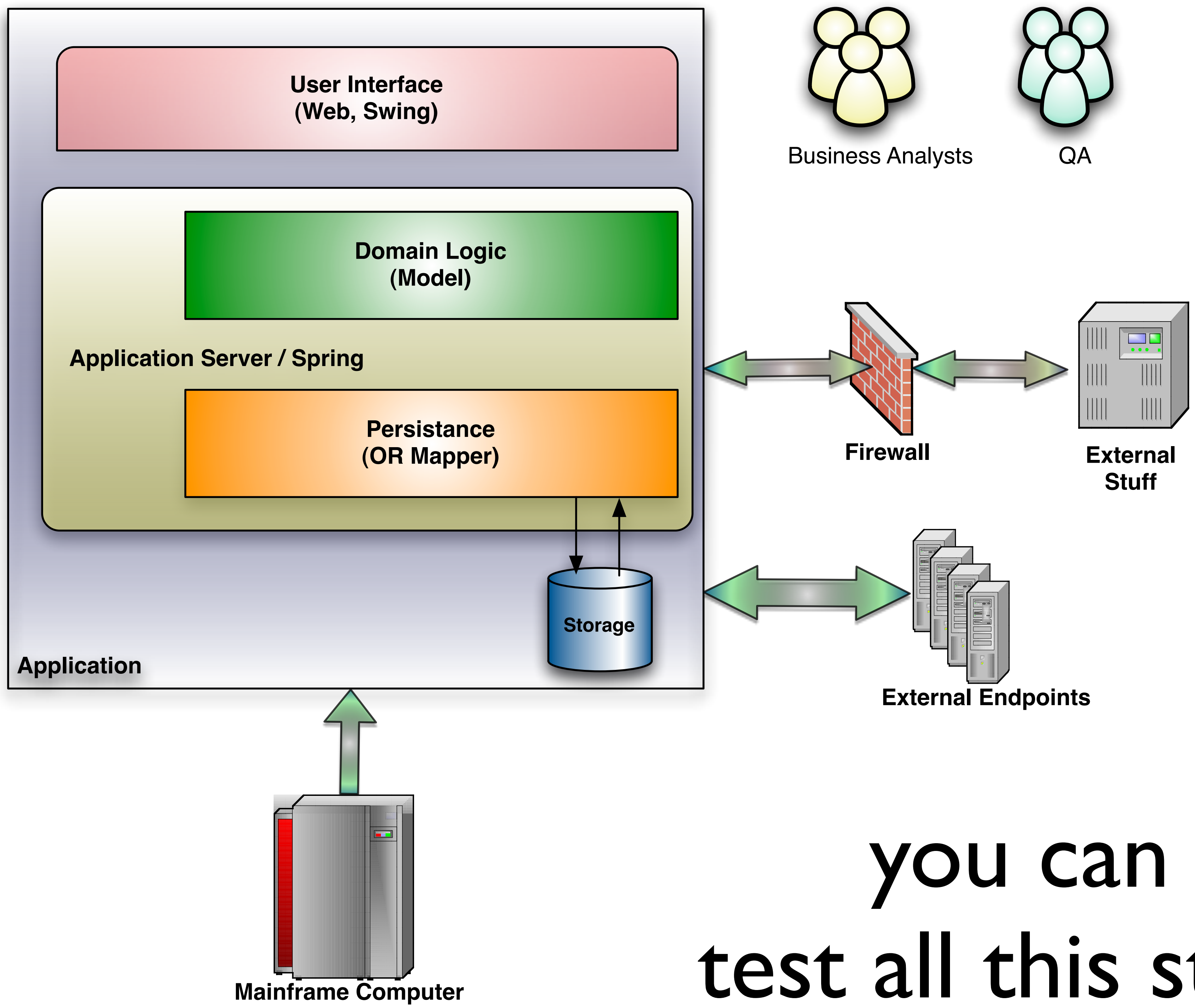
supports most swing controls

Swing?



location independence

http://frankenstein.openqa.org/



you can
test all this stuff!

ThoughtWorks®

?'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

NF