Testing a Scout Application with JUnit and Jubula (Presented by BSI)

EclipseCon 2013

Alexandra Schladebeck [BREDEX GmbH]
Jérémie Bresson [BSI Business Systems Integration AG]
APPLICATION UNDER TEST
Requirements (1)

Modification of the input fields...

...will update the image and summary fields.
Requirements (2)

Only the available parts are listed in the field

Is only a part available, the field is disabled
Requirements (3)

Export button:
- validate the form
- register the minifig in the server
- reset the form
ECLIPSE SCOUT
Client / Server integration
Eclipse Scout SDK: quick results

```java
@Order(10.0)
public class QuantityColumn extends AbstractIntegerColumn{

    @Override
    protected boolean getConfiguredEditable() {
        return true;
    }

    @Override
    protected String getConfigurerHeaderText() {
        return Texts.get("Quantity");
    }
}
```
Nice user interface fields

<table>
<thead>
<tr>
<th>Legs</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Odd</td>
<td></td>
</tr>
<tr>
<td>Blue</td>
<td></td>
</tr>
<tr>
<td>Gray</td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td></td>
</tr>
<tr>
<td>Jeans</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEAD</td>
<td>Normal</td>
<td>2</td>
</tr>
<tr>
<td>HEAD</td>
<td>Pirate</td>
<td>3</td>
</tr>
<tr>
<td>HEAD</td>
<td>Talky &amp; Walky</td>
<td>2</td>
</tr>
<tr>
<td>HEAD</td>
<td>Bart</td>
<td>5</td>
</tr>
<tr>
<td>LEGS</td>
<td>Blue</td>
<td>3</td>
</tr>
<tr>
<td>LEGS</td>
<td>Gray</td>
<td>9</td>
</tr>
<tr>
<td>LEGS</td>
<td>Overall</td>
<td>8</td>
</tr>
<tr>
<td>LEGS</td>
<td>Jeans</td>
<td>5</td>
</tr>
<tr>
<td>LEGS</td>
<td>Odd</td>
<td>7</td>
</tr>
<tr>
<td>LEGS</td>
<td>Yellow</td>
<td>4</td>
</tr>
<tr>
<td>TORSO</td>
<td>Yellow</td>
<td>8</td>
</tr>
<tr>
<td>TORSO</td>
<td>Police</td>
<td>1</td>
</tr>
<tr>
<td>TORSO</td>
<td>Suit</td>
<td>1</td>
</tr>
</tbody>
</table>
Automated testing – through the GUI

As a user would work – passing through all layers
Test creation, execution, analysis

Drag and drop test creation:

- No recording
- No programming
- With the same ♥ as development code

Constant feedback about quality

- Acceptance testing
- Regression testing
BACKGROUND
The application under test
A Scout application

Client Application

- Client
- UI
- Shared

Scout Client

Eclipse Platform

Server Application

- Server
- Shared

Scout Server

Eclipse Platform

Service Tunnel
A Scout application

Client Application

Server Application

Scout Client

Scout Server

Eclipse Platform

Eclipse Platform

ui

client

shared

service

shared

Service Tunnel
A Scout application

Client Application

Client Application

Server Application

Scout Client

Eclipse Platform

Scout Server

Eclipse Platform

ui

client

shared

server

shared

Service Tunnel

Copyright (c) 2013 BSI Business Systems Integration AG, Bredex GmbH and others. Licensed under EPL v1.0
JUnit
Test for logic in the shared plugin
Where do I put my tests?

- `<app>.client`
  - `<app>.client.test`
- `<app>.server`
  - `<app>.server.test`
- `<app>.shared`
  - `<app>.shared.test`
Running the tests from Eclipse
Using maven + tycho

- Add the maven nature to the project. Maven packaging:
  - `eclipse-plugin` for plugins
  - `eclipse-test-plugin` for test fragments

- Import JUnit as imported Packages:
JUNIT IN A SCOUT CONTEXT
Test with the Scout services
@RunWith annotation

Annotate the test class with the Annotation:

```java
@RunWith(ScoutClientTestRunner.class)
public class DesktopFormTest {

    //...

}
```

It adds:

- Equinox OSGi Runtime
- Scout Context, Services, …
TestingUtility\(\cdot\)\_registerServices(\(\cdot\))

TestingUtility allows to register other implementation for services.

public class P\_DesktopProcessService
        implements IDesktopProcessService {

        //Controlled implementation
        }

The Service is

- registered before each tests @Before
- unregistered after each test @After
Running the tests with maven + tycho build
JUBULA
Test with Jubula

Black Box

Client Application

Server Application

Client Application

Server Application

Scout Client

Scout Server

Eclipse Platform

Eclipse Platform

ui

client

shared

Service Tunnel

Copyright (c) 2013 BSI Business Systems Integration AG, Bredex GmbH and others. Licensed under EPL v1.0
Using the specification to automate tests

- Replace text
- Select from smart field
- Select from smart field
- Select from smart field
- Check text
Testing an application with Jubula

1. Write test
2. Start AUT
3. Perform object mapping
4. Run test
5. Analyse
Demo
Summary

- Developer tests (unit level)
  - Plain JUnit
  - JUnit in a Scout context

- User-centric tests
  - Jubula

Sources: [http://wiki.eclipse.org/Scout/Demo#Minifig_Application](http://wiki.eclipse.org/Scout/Demo#Minifig_Application)
Give Feedback on the Sessions

1. Sign In: www.eclipsecon.org

2. Select Session Evaluate

3. Vote +1 0 -1