Testing Java EE with Arquillian
The star shining in the universe of Java testing

JBoss QE Associate, Red Hat
Karel Piwko
February 11, 2011
Agenda

1. What is Arquillian?
2. How to use Arquillian?
3. Arquillian on the edge
Section 1
What is Arquillian?
What is Arquillian?

Pick the correct answer

(a) A part of comprehensive tool set for application developers
(b) Another testing framework reinventing the wheel
(c) A fancy name for a beer bottle opener
What is Arquillian?

Pick the correct answer

(a) A part of comprehensive tool set for application developers
(b) Another testing framework reinventing the wheel
(c) A fancy name for a beer bottle opener
Java EE application testing

Problems

Java EE applications are complex, thus it is difficult to isolate components

- communication (JMS, HornetQ, ...)
- UI (web based - JSF, JSP, RichFaces, GWT)
- database layer (JPA, Hibernate, ...)
- application server (JBoss AS, GlassFish, WebSphere, ...)

Testing is highly time consuming, not enjoyable and hard to be done properly!
Goals of Arquillian

- Provide a simple way how to write integration test
  - Manage container’s lifecycle
  - Build and deploy test archive
  - Enrich test classes
  - Capture test results
  - Keep configuration externally
  - Isolate classpath
- Can be easily extended to support tool of your choice

*Arquillian makes integration testing a breeze*
Parts of Arquillian

Not a complete list, but what you might find useful:

**Arquillian**
One framework to rule them all, still your tests are basically JUnit or TestNG

**ShrinkWrap**
The crucial component to pack your testing archive

**Descriptors**
A DSL language in Java to create mock XML configuration, for instance:
- Java EE descriptors, Arquillian configuration, etc.

**Extensions and supported frameworks**

**Dependencies**
Use Maven to fetch dependencies

**Byteman**
Coming soon!

**JSFUnit**
Test JSF pages

**Jacoco**
Measure test coverage

**Selenium**
Run functional tests

**Ajocado**
Type safe AJAX tests

**Other remarks**
Support for JBoss AS 5, 6; Jetty 7; Glassfish 3; Tomcat 6; JSR-299 impls; OSGi; ...
What is Arquillian?

Execution scheme

- JUnit / TestNG
- ShrinkWrap
- Any test framework
- Containers

JVM
Section 2

How to use Arquillian?
Arquillian modes

Set Arquillian mode for your test with `@Run(RunModeType)`. You can mix them as they can be specified per test method.

**IN_CONTAINER**
- The default way
- Test is deployed along side with deployment
- Test is run inside of container

**AS_CLIENT**
- Use Arquillian to build `@Deployment`
- Test is not run inside of container
ShrinkWrap

What it does?

- Builds JAR, WAR or EAR archive directly in Java code
- Let you pick up only the required parts of application
- Allows you to reuse Maven bits if desired
- Import from / export to external archives

```java
ShrinkWrap.create(JarArchive.class)
  .addClasses(Foo.class, Bar.class)
  .addPackages(Z.class.getPackage());
```
ShrinkWrap Descriptors

- Build an XML from Java using type-safe DSL
  - Specify only required bits
  - Modify existing files
- Descriptors can be deployed by Arquillian aside archives
Enabling Arquillian for your test

```java
@RunWith ( Arquillian.class )
public class Test {

    @Deployment
    public static Archive<?> war() {
        return ShrinkWrap.create ( WebArchive.class , "test.war" )
            .addClasses ( TheBean.class )
            .setWebXML ( new File ( "src/test/web.xml" ) );
    }

    @Test
    public void testExtraFeature() {
        ...
    }
}
```
How to use Arquillian?

TestNG

Enabling Arquillian for your test

```java
public class Test extends Arquillian {

    @Deployment
    public static Archive<?> jar() {
        return ShrinkWrap.create(JavaArchive.class)
            .addPackage(MyBean.class.getPackage());
    }

    @Test
    public void testExtraFeature() {
        ...
    }
}
```
How to make your test units isolated?

**Injection, EJB**

- Package necessary classes and configuration files
- Use `@Inject` or `@EJB` in test class to get instance from container

```java
@Deployment
public static JavaArchive createDeployment() {
    return ShrinkWrap.create(Ja Va Archive.class, "test.jar")
        .addClasses(GreetingManager.class,
                    GreetingManagerBean.class);
}

@EJB GreetingManager greetingManager;

@Test
public void shouldGreetUser() throws Exception {
    ...}
```
How to use persistence context?

**Persistence**

- Package necessary classes and configuration files
- Use `@PersistenceContext` and `@Produces` to create `EntityManager` which is bound automatically

```java
@PersistenceContext  @Produces  @Default
EntityManager em;

@EJB  AuctionManager auctionManager;

@Test
public void testLogin() {
    auctionManager.findAll();
    ...
}
```
Dive into extensions

Selenium/Ajocado

- Verify your application via functional test
- Let Arquillian manage:
  - Browser object - @Selenium
  - Deployed URL - @ContextPath
- Ajocado is Selenium on steroids

```java
@Selenium AjaxSelenium driver;
@ContextPath URL contextPath;

@Test @Run(AS_CLIENT)
public void testLogin() {
    driver.open(contextPath);
    driver.type(LOGIN_INPUT, "kpiwko");
    waitHttp(driver).click(LOGIN_BUTTON);
}
```
Dive into extensions cont’d

ShrinkWrap dependencies

- Include Maven artifacts in your ShrinkWrap archives
- Highly customizable
- Reuse Maven POM and settings files

@Deployment
public static Archive<?> war() {
  return ShrinkWrap.create(WebArchive.class)
    .addLibraries(
      Dependencies.artifact("foo:bar:1.0.0")
        .exclusions("foo:no", "foo:never")
        .artifact("foo:yes:pom:1.0.0")
          .scope("import")
          .resolve()
    );
}
Section 3

Arquillian on the edge
Coming soon in your tests!

- Multiple target (@Target) containers for an archive
- Inject Arquillian bits into your test classes - @ArquillianResource
  - Parallelization, HA and cluster testing
- Support cloud targets
- Multiple browser for Selenium extension
- More extensions (Byteman, RushEye support)
Where to continue?

Questions, feature requests, bug reports

- #jbosstesting on irc.freenode.net
- JIRAs (ARQ, ARQAJO, SHRINKWRAP, SHRINKDESC)
- jboss.org blogs and RSS
- JBUG in the future (Coming to Brno!)

Track current progress

- http://github.com/aslakknutsen/arquillian/tree/the_bigger_picture
- http://github.com/ALRubinger/shrinkwrap/tree/SHRINKWRAP-140
The end.

Thanks for listening.
<table>
<thead>
<tr>
<th>Lecture room</th>
<th>D2 (80)</th>
<th>D3 (150)</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00-9:45</td>
<td>ABRT 2.0 – Karel Klíč, Jiří Moskovčák</td>
<td>Matahari &amp; FMCI – Jaroslav Řezník</td>
</tr>
<tr>
<td>9:50-10:35</td>
<td>Beyond Myths: Revealing JSF 2 &amp; RichFaces 4 - Lukas Fryc</td>
<td>coreutils - tips &amp; common mistakes – Ondřej Vašík</td>
</tr>
<tr>
<td>10:40-11:25</td>
<td>The truth about Seam – Jozef Hartinger</td>
<td>OpenLDAP, Kerberos, SSSD, FreeIPA - Jan Vcelak, Zbysek Mraz, Jan Zeleny, Pavel Zuna</td>
</tr>
<tr>
<td>11:30-12:30</td>
<td>lunch</td>
<td>lunch</td>
</tr>
<tr>
<td>13:20-14:05</td>
<td>New features in OpenJDK 7 – Pavel Tišnovský</td>
<td>Debugging Tools Intro – Jan Kratochvíl</td>
</tr>
<tr>
<td>14:10-14:55</td>
<td>Byteman – Martin Večeřa</td>
<td>OpenSCAP – Peter Vrabec</td>
</tr>
<tr>
<td>15:00-15:45</td>
<td>Spacewalk on PostgreSQL – Jan Pazdziora</td>
<td>TeX Live – Jindřich Nový</td>
</tr>
<tr>
<td>15:50-16:35</td>
<td>Confining Spacewalk with SELinux – Jan Pazdziora</td>
<td>Amateur radio in Fedora – Jaroslav Škarvada</td>
</tr>
<tr>
<td>Time</td>
<td>Lab1 (B007)</td>
<td>Lab2 (B011) – Laptops needed</td>
</tr>
<tr>
<td>----------</td>
<td>---------------------------------</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>9:00-10:10</td>
<td></td>
<td>Infinispan 4 - Data Grids Hands-On lab - Radoslav Husar, Michal Linhard</td>
</tr>
<tr>
<td>10:15-11:25</td>
<td>Django for beginners - Dan Mach</td>
<td>Infinispan 4 - Data Grids Hands-On lab (continued) - Radoslav Husar, Michal Linhard</td>
</tr>
<tr>
<td>11:30-12:30</td>
<td>lunch</td>
<td>lunch</td>
</tr>
<tr>
<td>12:30-13:40</td>
<td>Working with Tito - Miroslav Suchý</td>
<td>Firewalld - Thomas Woerner (presentation and discussion)</td>
</tr>
<tr>
<td>13:45-14:55</td>
<td>SSSD setup - Jan Zeleny, Jakub Hurozek</td>
<td></td>
</tr>
<tr>
<td>15:00-16:10</td>
<td></td>
<td>Modern Enterprise Java Development and Testing – Karel Piwko, Lukas Fryc</td>
</tr>
<tr>
<td>16:10-17:25</td>
<td></td>
<td>Modern Enterprise Java Development and Testing (continued) – Karel Piwko, Lukas Fryc</td>
</tr>
</tbody>
</table>