Arquillian - a different kind of testing

Nils Heyer
Christian Neumann

Fontys Hogeschool voor Techniek en Logistiek

November 30, 2015

What is Arquillian

Advantages of testing with Arquillian

- First

Disadvantages of testing with Arquillian

- First

Arquillian unit test

Simple Greeter test

```
@Inject
Greeter greeter;

@Test
public void should_create_greeting() {
    Assert.assertEquals("Hello, Earthling!",
        greeter.createGreeting("Earthling"));
}
```

- **@Injected** Dependency inject objects that are used during the tests, equals as using jUnits *@before* to set up tests
- **@Test** defines which methods are tests and should be executed
- **Asserts** are the same as in jUnit

Shrinkwrapper example

Greeter test Shrinkwrapper

```
@Deployment
public static JavaArchive createDeployment() {
    JavaArchive jar = ShrinkWrap.create(↩
        JavaArchive.class)
        .addClasses(Greeter.class, PhraseBuilder↩
            .class)
        .addAsManifestResource(EmptyAsset.↩
            INSTANCE, "beans.xml");

    return jar;
}
```

- **@Deployment** defines which method is run before a test is executed. It need to return the minimal archive which is needed to execute the tests of this class
- **JavaArchive** a class representing the structure of a jar file
- **ShrinkWrap.create** allows to create an archive of the specified class, in case of a unit test JavaArchive.class

Shrinkwrapper example

Greeter test Shrinkwrapper

```
@Deployment
public static JavaArchive createDeployment() {
    JavaArchive jar = ShrinkWrap.create(↩
        JavaArchive.class)
        .addClasses(Greeter.class, PhraseBuilder↩
            .class)
        .addAsManifestResource(EmptyAsset.↩
            INSTANCE, "beans.xml");

    return jar;
}
```

- **addClasses** allows to add classes,that are needed during the tests, to the jar
- **addAsManifestResource** allows to add a resource,that are needed for the execution e.g. database configuration file, to the jar
- **EmptyAsset.INSTANCE** creates an empty file

Complete test

Greeter test

```
@RunWith(Arquillian.class)
public class GreeterTest {

    @Deployment
    public static JavaArchive createDeployment() {
        JavaArchive jar = ShrinkWrap.create(JavaArchive.class)
            .addClasses(Greeter.class, PhraseBuilder.class)
            .addAsManifestResource(EmptyAsset.INSTANCE, "beans.xml");

        return jar;
    }

    @Inject
    Greeter greeter;

    @Test
    public void should_create_greeting() {
        Assert.assertEquals("Hello, Earthling!",
            greeter.createGreeting("Earthling"));
    }
}
```

- **@RunWith(Arquillian.class)** is need for every test class to indicate that this test should be executed with Arquillian

Arquillian persistence test

Arquillian functional test

Any open issues?

Not all understood?

- see `http://arquillian.org/guides/`

Questions?

Questions or remarks?