concepts | development | integration

CONCEPTS | DEVELOPMENT | INTEGRATION

Automatisiertes Testen von Java EE-Applikationen mit Arquillian

Sebastian Lammering
CDI AG

Firmenkurzportrait

Die CDI AG

ist ein IT-Beratungsunternehmen mit Standorten Rhein/Ruhr und Rhein/Main.

Unsere Leistungsschwerpunkte

liegen auf den Themen Projektmanagement und Technologieberatung.

Unsere Kunden

finden sich vorrangig im Konzernumfeld und im gehobenen Mittelstand.

Unsere Mitarbeiter

sind zertifizierte Spezialisten und unterstützen Sie in komplexen IT Projekten.

Firmenkurzportrait

Zahlen, Daten, Fakten

Unsere Experten

50 Mitarbeiter arbeiten in unseren Projekten.

Unsere Erfahrungen

mehr als

50.000

Beratertage haben wir seit unserer Firmengründung im Jahr 2000 bei unseren Kunden geleistet.

Unsere Kunden

In jedem

3.

DAX30 Unternehmen arbeiten wir in Projekten.

Unsere Erfolge

über

200

Projekte haben wir bei unseren Kunden bisher durchgeführt und erfolgreich beendet.

Agenda

1. Problembeschreibung
2. Einführung in Arquillian anhand eines EJB Tests
3. CDI & Servlet Test
4. Container Types in Arquillian
5. WebService Test
6. Arquillian Extensions
7. JSF Test

Wo ist das Problem?

- **Remote-Server** muss für den Testfall explizit zur Verfügung gestellt werden.
- **Datenbank** muss für den Testfall explizit zur Verfügung gestellt werden.
- **EJBs** sind bei Web-Applikationen **nicht „von außen“ aufrufbar**.
  - Testfälle können nicht in der gleichen VM ausgeführt werden.
  - Debugging von Tests ist erschwert.

Wo ist das Problem?

- **Remote-Server** muss für den Testfall explizit zur Verfügung gestellt werden.

  ⇒ **Java EE 6 definiert Embedded EJB Container**

- **Datenbank** muss für den Testfall explizit zur Verfügung gestellt werden.

- **EJBs** sind bei Web-Applikationen **nicht „von außen“ aufrufbar**.
  - Testfälle können nicht in der gleichen VM ausgeführt werden.
  - Debugging von Tests ist erschwert.

Wo ist das Problem?

- **Remote-Server** muss für den Testfall explizit zur Verfügung gestellt werden.
  ⇒ **Java EE 6 definiert Embedded EJB Container**
- **Datenbank** muss für den Testfall explizit zur Verfügung gestellt werden.
  ⇒ **In-Memory Datenbanken**
- **EJBs** sind bei Web-Applikationen **nicht „von außen“ aufrufbar**.
  - Testfälle können nicht in der gleichen VM ausgeführt werden.
  - Debugging von Tests ist erschwert.

Wo ist das Problem?

- **Remote-Server** muss für den Testfall explizit zur Verfügung gestellt werden.

  ⇒ **Java EE 6 definiert Embedded EJB Container**

- **Datenbank** muss für den Testfall explizit zur Verfügung gestellt werden.

  ⇒ **In-Memory Datenbanken**

- **EJBs** sind bei Web-Applikationen **nicht „von außen“ aufrufbar**.
  - Testfälle können nicht in der gleichen VM ausgeführt werden.
  - Debugging von Tests ist erschwert.

  ⇒ **???**

Was sollte ein Testframework können?

- Container starten und stoppen
- Für die Testfälle die benötigten Klassen und Ressourcen in Archive zusammenstellen
- Diese Archive in einem Container deployen
- Möglichkeiten bereitstellen, um in Testklassen auf EJBs und andere Ressourcen zuzugreifen
- Ausführung der Testfälle im Container
- Aufnahme und Weitergabe der Resultate an Entwicklungsumgebung und Build-System

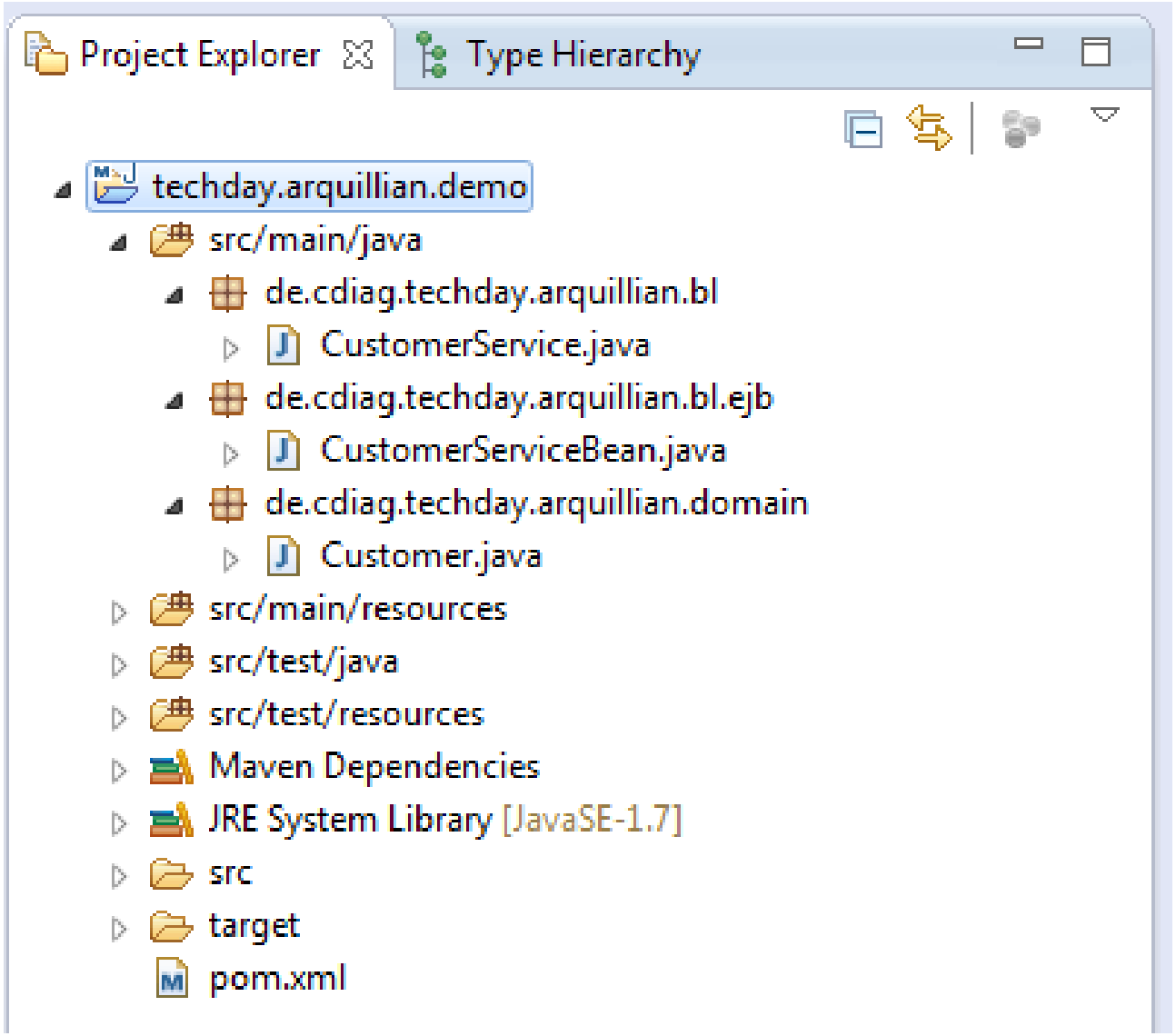
Welche Produkte werden eingesetzt?

- Embedded EJB Container

- In-Memory Database

- Testframework

Wie sieht das Projekt-Setup aus?



Was soll getestet werden?

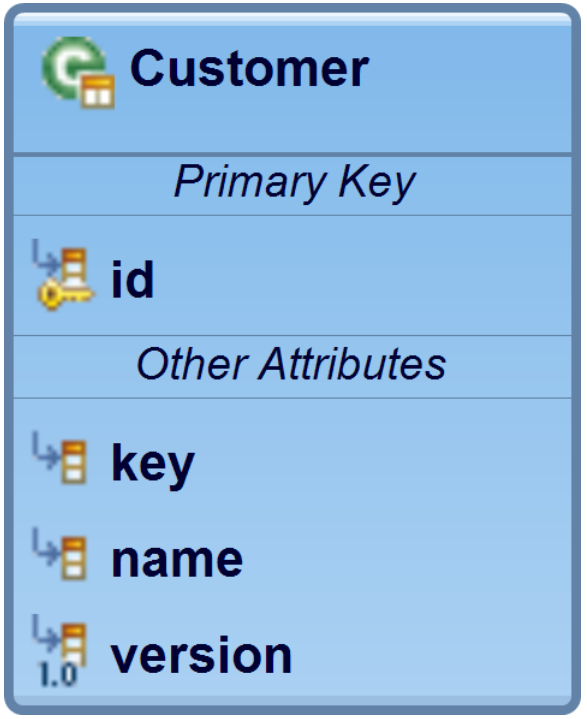
```
@Local
public interface CustomerService {

        List<Customer> getAll();

        Customer getByKey(String key);

        Customer save(Customer customer);

}
```



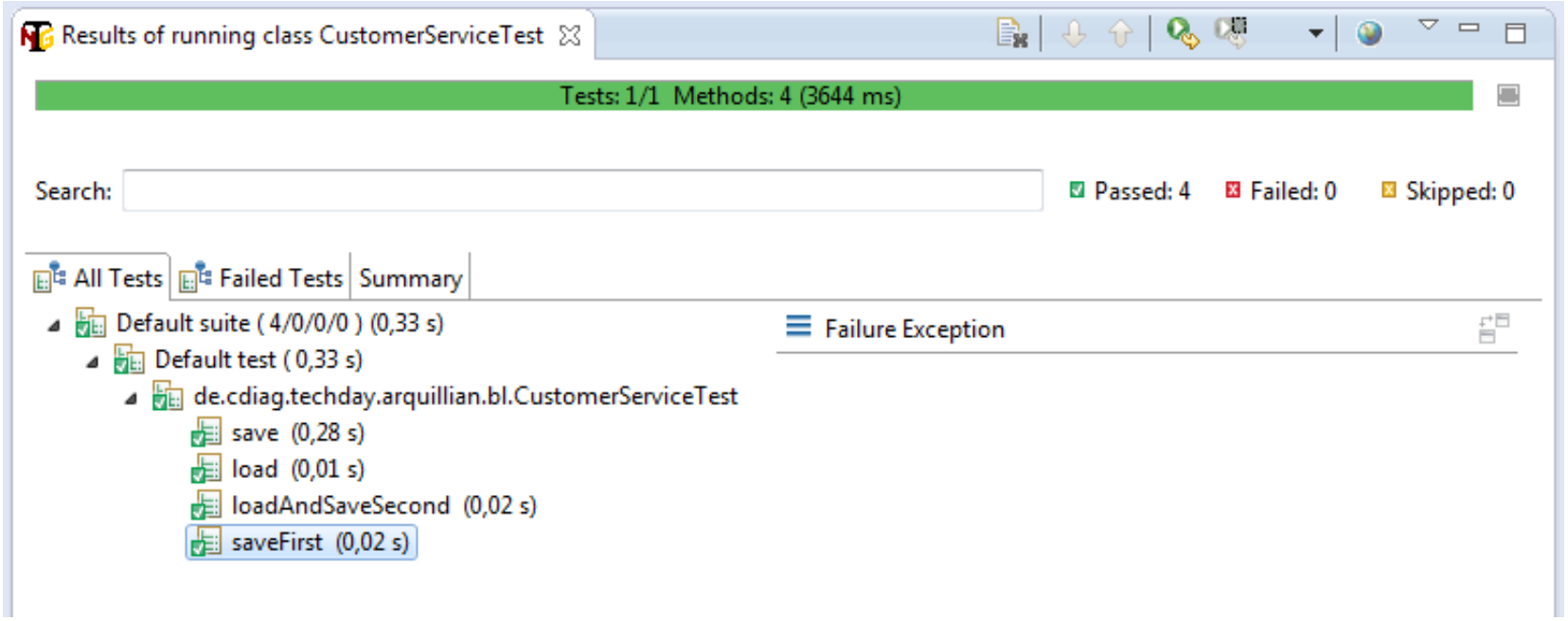
Wie wird es gemacht?

Dependencies in der POM erweitern

- JUnit
- Arquillian für JUnit
- OpenEJB Container

Testklasse schreiben

- Entity speichern
- Entity lesen
- Optimistic Locking



CDI Test

Container

- Weld EE 1.1

Beispiel:

```
public class HelloWorld {

    private static final String HELLO_TEMPLATE = "Hello %s!";

    public String sayHello(String name) {
        return String.format(HELLO_TEMPLATE, name);
    }

}
```

Servlet Test

Container

- Tomcat 7.0

Beispiel:

```
@WebServlet(name="HelloServlet", urlPatterns={"/hello"})
public class HelloServlet extends HttpServlet{

    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
            throws ServletException, IOException {
        PrintWriter out = resp.getWriter();
        out.println("Hello JugDo!");
    }
}
```

Unterstützte EJB, CDI & Servlet Container

Container	Embedded
Apache OpenEJB 3.1	
Apache OpenWebBeans 1.0	
Jetty 6.1	
Jetty 7.0	
Tomcat 5.5	
Tomcat 6	
Tomcat 7	
Weld SE 1.0	
Weld SE 1.1	
Weld EE 1.1	

Unterstützte EJB, CDI & Servlet Container

Container	Embedded	Managed	Remote
Apache OpenEJB 3.1			
Apache OpenWebBeans 1.0			
Jetty 6.1			
Jetty 7.0			
Tomcat 5.5			
Tomcat 6			
Tomcat 7			
Weld SE 1.0			
Weld SE 1.1			
Weld EE 1.1			

Unterstützte Java EE Container

Container	Embedded	Managed	Remote
GlassFish 3.1			
JBoss AS 5			
JBoss AS 5.1			
JBoss AS 6			
JBoss AS 7			
JBoss AS 7.1/EAP 6			
WebLogic 10.3			
WebLogic 12.1			
IBM WebSphere 7			
IBM WebSphere 8			

Web Service Test

- Container
  - JBoss AS 7.1.1

Beispiel:

```
@RequestScoped
@Path("/user")
public class UserResource {
    @EJB
    private UserService userService;

    @Produces("application/json")
    @Path("/list")
    @BadgerFish
    @GET
    public List<User> list() {
        return userService.findAllUser();
    }

    @Produces("application/json")
    @Path("/find/{userId}")
    @BadgerFish
    @GET
    public User find(@PathParam("userId") Integer userId) {
        return userService.find(userId);
    }

    @Consumes("application/json")
    @Produces("application/json")
    @Path("/save")
    @BadgerFish
    @POST
    public Response update(User user) {
        userService.saveUser(user);
        return Response.ok().build();
    }

    @Produces("text/plain")
    @Path("/email/{email}")
    @GET
    public String find(@PathParam("email") String email) {
        List<User> peeps = userService.findByEmail(email);
        if(peeps.size() > 0) {
            User p = peeps.get(0);
            return String.format("%s %s %s",p.getSalutation(),p.getFirstName(),p.getLastName());
        } else {
            return "none found.";
        }
    }
}
```

Arquillian Extension

- Persistence
- Performance
- Seam 2
- Drone
- Warp
- Graphene

Persistence

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

    @Deployment
    public static Archive<?> createDeploymentPackage() {
        return ShrinkWrap.create(JavaArchive.class, "test.jar")
                .addPackage(UserAccount.class.getPackage())
                .addAsManifestResource(EmptyAsset.INSTANCE, "beans.xml")
                .addAsManifestResource("test-persistence.xml", "persistence.xml");
    }

    @PersistenceContext
    EntityManager em;

    @Test
    @UsingDataSet("datasets/users.yml")
    @ShouldMatchDataSet("datasets/expected-users.yml")
    public void should_change_user_password() throws Exception {
        // given
        String expectedPassword = "LexLuthor";
        UserAccount user = em.find(UserAccount.class, 2L);

        // when
        user.setPassword("LexLuthor");
        user = em.merge(user);

        // then
        assertEquals(expectedPassword, user.getPassword());
    }
}
```

users.yml

```yaml
useraccount:
  - id: 1
    firstname: John
    lastname: Smith
    username: doovde
    password: password
  - id: 2
    firstname: Clark
    lastname: Kent
    username: superman
    password: kryptonite
```

Performance

```
// include other arquillian imports here...
import org.jboss.arquillian.performance.annotation.Performance;
import org.jboss.arquillian.performance.annotation.PerformanceTest;

@PerformanceTest(resultsThreshold=2)
@RunWith(Arquillian.class)
public class WorkHardCdiTestCase
{
   @Deployment
   public static JavaArchive createDeployment() {
       return ShrinkWrap.create(JavaArchive.class ,"test.jar")
               .addPackage( WorkHard.class.getPackage())
               .addAsManifestResource(
                       EmptyAsset.INSTANCE,
                       ArchivePaths.create("beans.xml"));
   }

   @Inject HardWorker worker;

   @Test
   @Performance(time=20)
   public void doHardWork() throws Exception
   {
      Assert.assertEquals(21, worker.workingHard(), 0d);
   }
}
```

Hint: double-click to select code

Seam 2

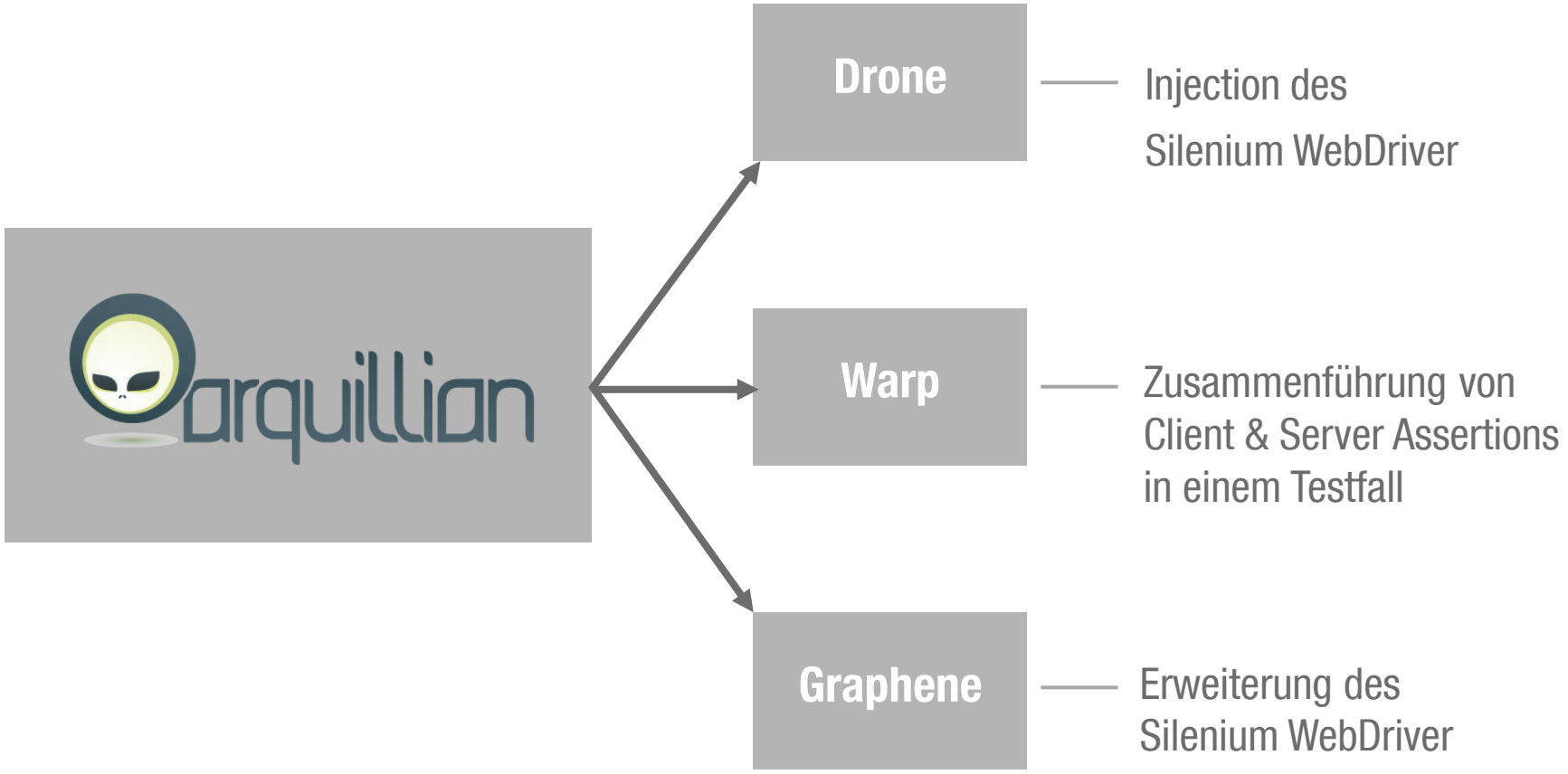
```java
import static org.fest.assertions.Assertions.assertThat; // Fluent assertions used in the test
// ... Arquillian and Seam 2 relevant imports

@RunWith(Arquillian.class)
public class ComponentInjectionTestCase
{
   @Deployment
   public static Archive<?> createDeployment()
   {
      return ShrinkWrap.create(WebArchive.class, "test.war")
                       .addClass(FluidOuncesConverter.class)
                       .addPackages(true, "org.fest") // Needed to run in managed / remote container
                       .addAsResource(EmptyAsset.INSTANCE, "seam.properties")
                       .setWebXML("web.xml");
   }

   @In
   FluidOuncesConverter fluidOuncesConverter;

   @Test
   public void shouldInjectSeamComponent() throws Exception
   {
      assertThat(fluidOuncesConverter).isNotNull();
   }
```

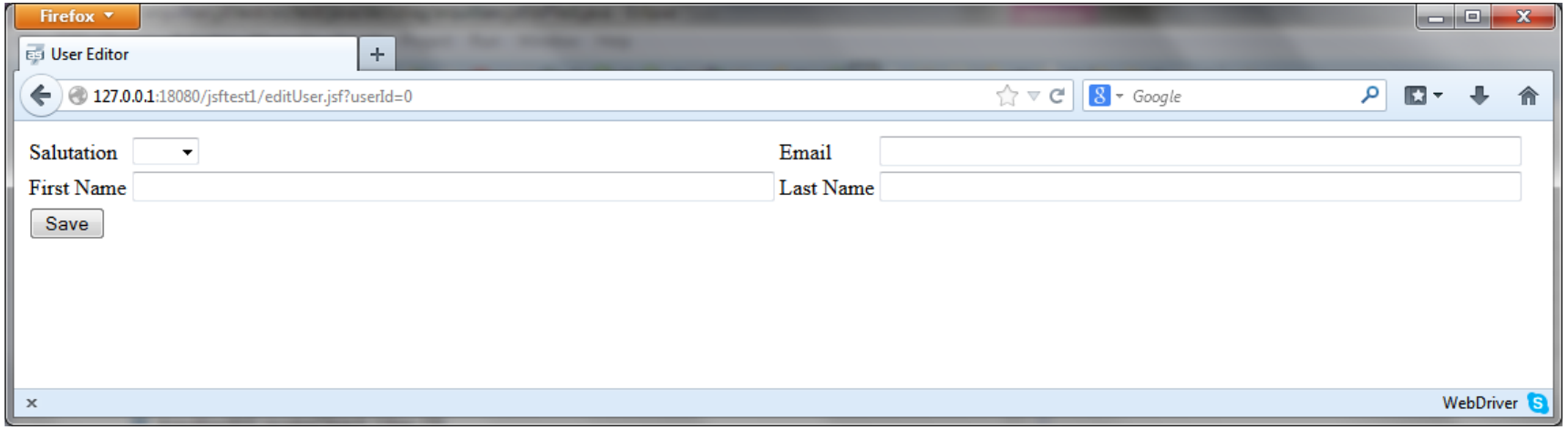
Arquillian Extensions für funktionale Tests



JSF Test

- **Container**
  - JBoss AS 7.1.1
- **Extensions**
  - Drone
  - Warp

Beispiel:



concepts | development | integration

cdi

Contacts

Sebastian Lammering

CDI Concepts Development Integration AG

Rhein/Ruhr

Lindemannstraße 79-81
D-44137 Dortmund

+49 231 - 108 762 0

Rhein/Main

Im Leuschnerpark 4
D-64347 Griesheim (Darmstadt)

+49 6155 - 605 359