

Orientation in Objects

JBoss Seam

Seam – oder wie Nähe ich ein Java EE Patchwork

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Version: 1.2

„Orientierung“ in Objekten

Orientation in Objects

Java und XML

) Akademie)

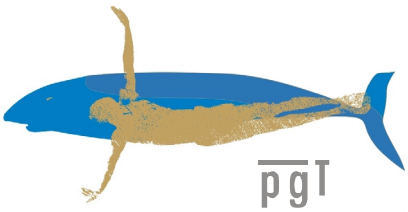
- **Schulungen**, **Coaching**, **Weiterbildungsberatung**, **Train & Solve**-Programme

) Beratung)

- **Methoden**, **Standards** und **Tools** für die Entwicklung von **offenen, unternehmens-weiten Systemen**

) Projekte)

- **Schlüsselfertige Realisierung** von Software
- **Unterstützung laufender Projekte**
- **Pilot- und Migrationsprojekte**



Papick Garcia Taboada

Software Architekt

Technologie-Scout

Beratung

Projekte

Training

Orientation in Objects

Serge Ndong

Entwickler

Berater

Trainer

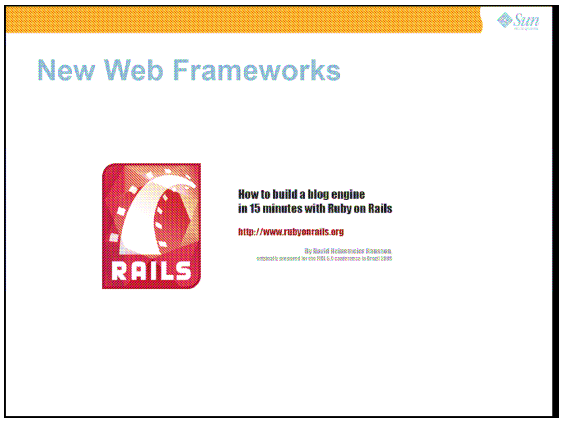
Gliederung

Orientation in Objects

- **Was ist Seam?**
- Wie funktioniert Seam?
- Fazit?

Blog in 15 min?

Orientation in Objects



Keynote W-JAX 2006
Tim Bray,
Director of Web Technologies (Sun)

Endlich ein Webapp Framework

Orientation in Objects

- Framework zur Erstellung von Enterprise Java Webapplikationen
- ein mächtiges Framework, um moderne Web 2.0 Anwendungen [sehr einfach] zu bauen

JBoss™ ???

Orientation in Objects

- **Ist JBoss Seam eine proprietäre Lösung?**
  - JBoss Seam ist Open Source
  - JBoss Seam ist mit anderen JBoss Produkten verzahnt
  - Aus Seam ist das JSR 299 entstanden: “WebBeans”
  - erster JSR unter der Leitung von JBoss

Lückenfüller?

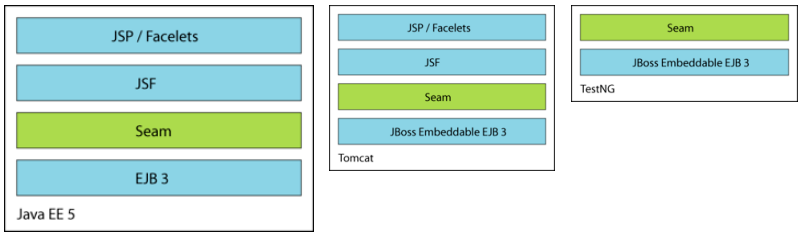
Orientation in Objects

- schließt eine wichtige Lücke in der Architektur von Java EE 5
  - vereint EJB 3.0 und Java Server Faces (JSF) in einem einzigen Programmiermodell
  - kein „Schichten Glue Code“ mehr nötig, Programmierzeit wird gespart

Wie leichtgewichtig ist es?

Orientation in Objects

- dank Java EE 5 leichtgewichtigeren Ansatz



Wie leichtgewichtig ist es?

Orientation in Objects

- EJB 3 Applikationen können in Web-Container laufen
  - EJB 3 Applikationen können in TestNG laufen
  - Seam auch mit JavaBeans und Hibernate einsetzbar
    - **mit JBoss Embeddable Container**

Geschichte

Orientation in Objects

- Lead durch Gavin King (Hibernate)
- Versionen
  - 1.1.0 BETA 1 seit 24.10.2006
  - 1.0.1 Produktion 18.06.2006
  - 1.0.0 Produktion 11.06.2006
  - 1. Beta 18.09.2005

EJB-zentrische Entwicklung

Orientation in Objects

- EJBs auch im Frontend
- EJB Komponenten sind annotierte POJOs
- Aufhebung der Schichtentrennung durch Facade, Business Delegate und Transfer Objects
- Schichten lediglich im Objektmodell abgebildet

AJAX-basiertes Remoting

Orientation in Objects

- Webclient hat direkten Zugriff auf EJB Session Beans
- Bereitstellung von EJB-Komponenten als JavaScript Proxy-Objekte
- Webclients können JMS Nachrichten empfangen

Prozessgetriebene Applikationen

Orientation in Objects

- eigene Business Processing Management Engine (jBPM) für die Implementierung von Workflows und Pageflows

Testbarkeit++

Orientation in Objects

- POJOs sind einfach zu testen
- Umfangreiche Testmöglichkeiten (mit TestNG)
  - z.B : Nachbilden einer Interaktion mit einem Benutzer des Systems

Zustandsbehaftete Applikationen

Orientation in Objects

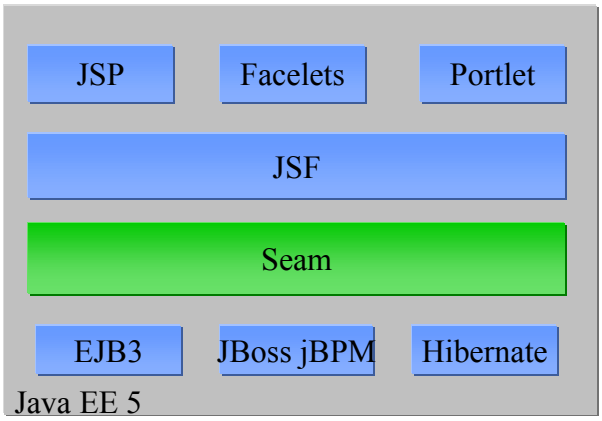
- Seam definiert zwei weitere Kontexte
  - Conversation Scope:
    entspricht einem kurzfristigen Dialog
  - Business Process Scope:
    entspricht einem langlebigen Dialog über mehrere Sitzungen.

Bijection

Orientation in Objects

- steht für „bidirectional injection“ oder injection und outjection
- Erweiterung des Grundgedankens „Dependency Injection“

„Nähkästchen“



Presentation Tier

Request Controller

Context Manager

State Management

EJB3 und mehr…

- JSF als Frontend-Technologie
  - + einige Erweiterungen
  - Facelets
  - aktuell Myfaces, andere Implementierungen denkbar…
- JBoss jBPM (Java Business Process Management)
- JBoss Rules (Rules Engine)

Facelets?

Orientation in Objects

- Aus der JBoss Seam FAQ, leider nicht mehr online:
  - **What view templating technologies can I use?**
  - *You can use any templating technology that support JSF. In practice, this probably means Facelets (if you take our advice) or JSP (for masochists).*

Gliederung

Orientation in Objects

- Was ist Seam?
- **Wie funktioniert Seam?**
- Fazit?

Seam Komponentenmodell? (1)

Orientation in Objects

- wichtig: Seam hat kein eigenes Komponentenmodell
  - Seam stimmt die Java EE Komponenten aufeinander ab!
  - sobald Seam die Komponenten kennt, werden die Brücken geschlagen

Seam Komponentenmodell? (2)

Orientation in Objects

- Seam Komponenten werden mit Kontext Variablen mittels @Name bzw. @Role definiert
- mittels der Annotation @Scope kann man Seam Komponenten einem Scope zuweisen

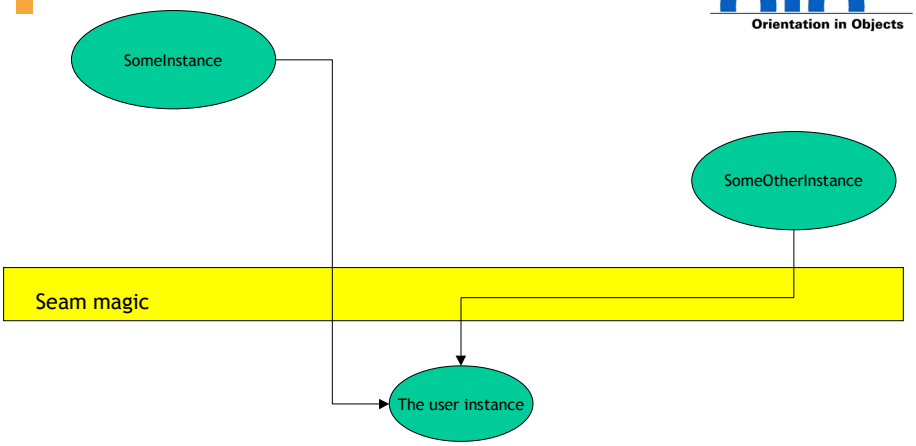
Bijection

Orientation in Objects

- Dependency Injection im Kontext
- Komponenten in grösseren Scopes können auf Komponenten in kleineren Scopes
- Bijection ist dynamisch, kontextbezogen und biderektional

Magic?

Orientation in Objects



@In …

Orientation in Objects

- Bijection durch annotation
  - an Attribut oder Getter- und Setter-Methoden
- @In wird eine "dependency injection" vermerkt

… and @Out

Orientation in Objects

- @Out wird die Instanzvariable in die Seam Laufzeitumgebung angemeldet
  - Outjected…

Bijection

Orientation in Objects

```
@Stateful
@Name("changeUserPassWord")
public class ChangeUserPassWord{
   @PersistenceContext
   private EntityManager em;

   @In @Out
   private User user;

   ...
}
```


Bijection

Orientation in Objects

- Möglichkeit Expression Language-Syntax in den Annotations zu verwenden

```
@Name("loginAction")
public class LoginAction implements Login {
      @In("#{user.username}")
      String username;
      ...
}
```

Servlet API Scopes

Orientation in Objects

- Die Servlet API stellt lediglich den Request, Session und Applikation Scope zur Verfügung.

Nicht akzeptabler Zustand…

Orientation in Objects

- Request und Session Scopes nicht ausreichend für die Entwicklung von Anwendungen
  - kleinere Sitzungen müssen durch den Entwickler verwaltet werden
  - Dialoge fehlen in der JSF Spezifikation
    - **Struts Shale bietet Implementierungen an…**

Seam Kontexte

Orientation in Objects

- bisher bekannte Kontexte
  - Stateless Context
  - Event Context
  - Page Context
  - Session Context
  - Application Context

Neu: Conversation Context

Orientation in Objects

- ein Dialog ist ein „Unit of Work“
- Anfang und Ende eines Dialoges werden durch annotierte Methoden definiert
- Speicherort für Dialogspezifische Informationen
- Konversationen können verschachtelt sein

Da guck!

Orientation in Objects

- Mit Seam-Lab Kopfschmerzenfrei loslegen und staunen…
- Das Seam Hotelreservierungsbeispiel…
- http://localhost:8080/seam-booking/

Neu: Business Process Context

Orientation in Objects

- beschreibt ein Dialog das über die Lebenszeit einer Sitzung hinaus dauern kann
- jBPM Engine ist für das Management und das Speichern der Sitzungsdaten verantwortlich

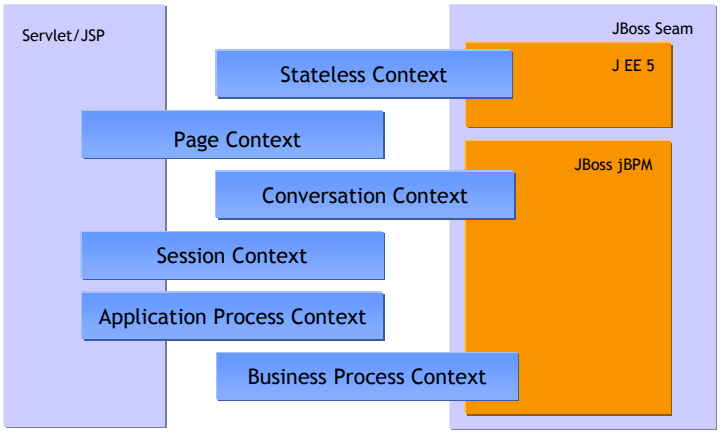
Anfang und Ende

Orientation in Objects

- Dialog Beginn und Ende werden durch Annotationen gekennzeichnet
  - Bei einer Conversation:
    - **@Begin**
    - **@End**
  - Bei einem Business Process:
    - **@BeginTask**
    - **@EndTask**

Übliche Seam Kontexte

Orientation in Objects



Komponententypen und die Scopes

Orientation in Objects

- Stateful Session Bean
  - default ist Conversation scope
- Stateless Session Bean
  - default ist Stateless scope
- Entity Bean
  - default ist Conversation scope
- Java Bean
  - default ist Event scope

EJB3:
Configuration
by exception!

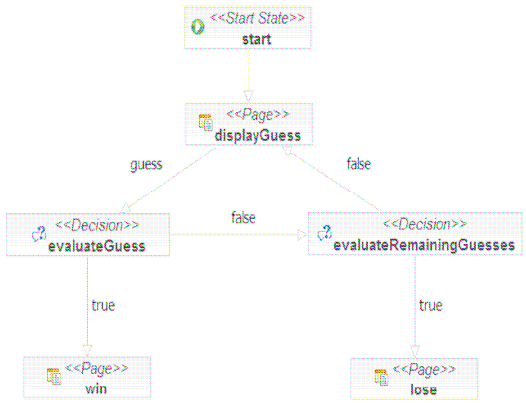
Navigation

Orientation in Objects

- JSF Navigation nicht ausreichend…
- Wunschliste
  - Navigation grafisch gestalten
  - Seitenfluss verbindlich vorgeben
  - Navigation wie bisher deskriptiv, Engine soll die Arbeit übernehmen

Grafische Pageflow Definition

Orientation in Objects



XML Pageflow Definition

Orientation in Objects

```
<pageflow-definition name="numberGuess">
  <start-page name="displayGuess" view-id="/numberGuess.jsp">
            <redirect/>
            <transition name="guess" to="evaluateGuess">
               <action expression="#{numberGuess.guess}" />
            </transition>
  </start-page>
  <decision name="evaluateGuess"
             expression="#{numberGuess.correctGuess}">
            <transition name="true" to="win"/>
            <transition name="false" to="evaluateRemainingGuesses"/>
  </decision>
 <decision name="evaluateRemainingGuesses"
            expression="#{numberGuess.lastGuess}">
            <transition name="true" to="lose"/>
            <transition name="false" to="displayGuess"/>
 </decision>
 <page name="win" view-id="/win.jsp">
            <redirect/>
            <end-conversation />
</page>
<page name="lose" view-id="/lose.jsp">
            <redirect/>
            <end-conversation />
 </page>
</pageflow-definition>
```

Pageflow Einbindung

Orientation in Objects

```
@Name("numberGuess")
@Scope(ScopeType.CONVERSATION)
public class NumberGuess {

  (...)

@Begin(pageflow="numberGuess")
public void begin()
{
        randomNumber = new Random().nextInt(100);
        guessCount = 0;
        biggest = 100;
        smallest = 1;
}
```


Grafische Business Prozess Definition

Orientation in Objects

XML jBPM Process Definition

Orientation in Objects

```
<process-definition name="todo">
  <start-state name="start">
     <transition to="todo"/>
  </start-state>

  <task-node name="todo">
     <task name="todo"description="#{todoList.description}">
        <assignment actor-id="#{actor.id}"/>
     </task>

     <transition to="done"/>

  </task-node>

  <end-state name="done"/>

  </process-definition>
```


Business Process Einbindung

Orientation in Objects

```
@Name("todoList")
public class TodoList {

  private String description;

  public String getDescription(){
     return description;
  }
  public void setDescription(String description) {
     this.description = description;
  }

  @CreateProcess(definition="todo“)
  public void createTodo() {}

  @StartTask @EndTask
  public void done() {}
}
```

JBoss IDE / Eclipse

Orientation in Objects

Stateless Session Bean im Frontend

Orientation in Objects

```
<h:commandButton type="submit" value="Register" action="#{register.register}"/>
```

```
@Stateless
@Name("register")
public class RegisterAction implements Register
{
  @In(create=true)
  private EntityManager em;

  public String register(String username)
  {
    List existing = em.createQuery("select username from User where username=:username")
                      .setParameter("username", username).getResultList();

      if (existing.size()==0){
              em.persist(user);
              return "/registered.jsp"; }
      }

}
```

Validierung auf Entitäten definieren

Orientation in Objects

```
@Entity
public class User
{
    @Id @GeneratedValue
    private Long id;

    @NotNull
    private String firstname;

    @Length(max=25) @NotNull
    private String lastname;

    @Length(max=200)
    private String description;

   (...)
```


Validierung im Frontend einschalten

Orientation in Objects

```
<f:form>
  <table>
    <s:validateAll>
      <tr><td>Vorname</td>
        <td><h:inputText value="#{user.firstname}"/></td>
      </tr>
      <tr><td>Nachname</td>
        <td><h:inputText value="#{user.lastname}"/></td>
      </tr>
      <tr><td>Beschreibung</td>
        <td><h:inputText value="#{user.description}"/></td>
      </tr>
    </s:validateAll>
  </table>
  <h:messages/>
  <h:commandButton type="submit" value="Save„ action="#{user.save}"/>
</f:form>
```

Seam Remoting

Orientation in Objects

- JavaScript Proxies werden dynamisch zur Laufzeit generiert und dem Client mittels ein Servlet zu Verfügung gestellt
- JavaScript- Entwickler können ohne technische Klimmzüge direkt auf die Proxy Objekte zugreifen
- Webclients können JMS Nachrichten empfangen

Seam Remoting: Stateless Seam Komponente

Orientation in Objects

```
@Stateless
@Name("userFinder")
public class UserFinderBean implements UserFinder {

@In
PersistenceContext em;

public List<User> find(String searchString) {
    return em.createQuery("from User where firstname like :search")
    .setParameter("search", searchString)
    .getResultList();
    }

}
```

Seam Remoting: Local Interface

Orientation in Objects

```
@Local
public interface UserFinder {

    @WebRemote List<User> find(String searchString);

}
```


Seam Remoting: Frontend

Orientation in Objects

```
<div>
    <input type="text" id="searchString"/>
    <input type="submit" value="Search"
           onclick="doSearch(); return false;"/>
</div>
<div>
    <table id="results">
           <!– Suchergebnis anzeigen -->
    </table>
</div>
```

Zugriff auf Session Bean

Orientation in Objects

```
function doSearch() {
    var searchString = document.getElementById("searchString").value;
    var userFinder = Seam.Component.getInstance("userFinder");
    userFinder.find(searchString, displayResults);
}

function displayResults(docs) {
    (...)
}
```


Wie?

Orientation in Objects

```
<servlet>
    <servlet-name>Seam Remoting</servlet-name>
    <servlet-class>
        org.jboss.seam.remoting.SeamRemotingServlet
    </servlet-class>
</servlet>

<servlet-mapping>
    <servlet-name>Seam Remoting</servlet-name>
    <url-pattern>/seam/remoting/*</url-pattern>
</servlet-mapping>
```

Transaktionen und Cache

Orientation in Objects

- Seam verschiebt die Transaktionsgrenzen vom EJB Programmiermodell auf die Seam Komponenten
  - bessere Reaktion auf die Eigenarten einer Webanwendung

Transaktionen und Cache

Orientation in Objects

- Seam verwendet einen erweiterten Persistenzkontext, der einen ganzen Dialog-Kontext umfasst
  - Dadurch sind die Komponenten auch nach einem Request-Response-Zyklus ansprechbar
  - kein LazyInitializationException mehr

Transaktionen und Cache

Orientation in Objects

- Seam verwendet zwei Transaktionskontexte in einem Response-Request-Zyklus
  - Der erste umfasst das Update des Models und die Applikationslogik
  - der zweite umfasst das Rendern der Ausgabe.

Seam Dialoge und der Cache

Orientation in Objects

- Seam nutzt den Dialog-Kontext als intelligente Cache Lösung
  - der Anwendungsentwickler muss nicht mehr stets sein Session-Kontext aufräumen
  - alle benötigten Daten sind im richtigen Kontext gespeichert und werden nach Beenden des Dialoges aus dem Speicher entfernt
  - keine kontextlosen Second-Level-Caches wie in Hibernate

Gliederung

Orientation in Objects

- Was ist Seam?
- Wie funktioniert Seam?
- **Fazit?**

Respekt..

Orientation in Objects

Sun

New Web Frameworks

Don't disrespect them, learn from them. Java EE 5 is moving in a good direction, but we need to do much better.

Keynote W-JAX 2006
Tim Bray,
Director of Web Technologies (Sun)

Finden wir es gut?

Orientation in Objects

- Klares „Jain“

;-)

Genauer?

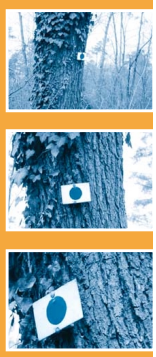
Orientation in Objects

- ☺ Vereinheitlichung des Programmiermodells
- ☺ JSF mit Dialoge (Struts Shale?)
- ☺ Remoting (nicht wirklich neu?)
- ☺ Weniger Konfigurationsaufwand

Genauer?

Orientation in Objects

- ☹ „One size fit all“ Architektur
  - Vereinfachungen umstritten
- ☹ „All in one package, take it or leave it“
  - JBPM
  - JBoss Rules
- ☹ Wenig Erweiterungsmöglichkeiten
  - Kein Spring?



Orientation in Objects

Fragen ?

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Version: 1.2

Orientation in Objects

Vielen Dank für Ihre Aufmerksamkeit !

Orientation in Objects GmbH

Weinheimer Str. 68
68309 Mannheim

www.oio.de
info@oio.de

Version: 1.2