Seam -

Java

2.2.3-SNAPSHOT

King Gavin [FAMILY Given], Muir Pete [FAMILY Given], Richards Norman [FAMILY Given], Bryzak Shane [FAMILY Given], Yuan Michael [FAMILY Given], Youngstrom Mike [FAMILY Given], Bauer Christian [FAMILY Given], Balunas Jay [FAMILY Given], Allen Dan [FAMILY Given], Andersen Max [FAMILY Given], Bernard Emmanuel [FAMILY Given], Karlsson Nicklas [FAMILY Given], Roth Daniel [FAMILY Given], Drees Matt [FAMILY Given], Orshalick Jacob [FAMILY Given], Forveille Denis [FAMILY Given], Novotny Marek [FAMILY Given], # Hartinger Jozef [FAMILY Given]

edited by Kittoli Samson [FAMILY Given]

and thanks to Cobb James [FAMILY Given] (##########), Weaver Cheyenne [FAMILY Given] (##########), Newton Mark [FAMILY Given], Ebersole Steve [FAMILY Given], Courcy Michael [FAMILY Given] (#######), Benaglia Nicola [FAMILY Given] (#######), Travelli Stefano [FAMILY Given] (#######), Milesi Francesco [FAMILY Given] (#######), # JBoss User Group Japan [FAMILY Given] (#####)

JBoss Seam

Seam ######### Java ######################################

######## *(###############)*

Seam ##Seam ### ###### Web ############################## ######## *(conversation context)* #######

Seam ##

Unlike plain Java EE or J2EE components, Seam components may *simultaneously* access state associated with the web request and state held in transactional resources (without the need to propagate web request state manually via method parameters). You might object that the application layering imposed upon you by the old J2EE platform was a Good Thing. Well, nothing stops you creating an equivalent layered architecture using Seam — the difference is that *you* get to architect your own application and decide what the layers are and how they work together.

JSF # EJB 3.0 ###

JSF # EJB 3.0 ##Java EE 5 ##################EJB3 ##JSF ### # EJB3 ##############################Java EE 5 ###

Seam # JSF# EJB3 ################# ################### (glue code) ######################################

It is possible to write Seam applications where "everything" is an EJB. This may come as a surprise if you're used to thinking of EJBs as coarse-grained, so-called "heavyweight" objects. However, version 3.0 has completely changed the nature of EJB from the point of view of the developer. An EJB is a fine-grained object — nothing more complex than an annotated JavaBean. Seam even encourages you to use session beans as JSF action listeners!

########### EJB 3.0 ############EJB 3.0 ##################### Java ######Seam ###################### Seam ##EJB ############################## (lightweight)#########################

AJAX

Seam################## JSF ### AJAX ##########JBoss RichFaces # ICEfaces ######################### JavaScript ############################ AJAX ################

###########Seam ###### JavaScript ## JavaScript

################################## JMS ###############AJAX ##########################

###############Seam ## AJAX ######################################

########################

#########Seam # jBPM ########################jBPM # Seam ##

Seam ##jBPM ###################### (jPDL) ######################################

JSF##Seam ##Seam ################## jBPM ######################################

########

EJB ##EJB 3.0 ## ############################## ## (set) ##### (get) ## ################################

Declarative application state management is made possible by the richness of the *context model* defined by Seam. Seam extends the context model defined by the servlet spec — request, session, application — with two new contexts — conversation and business process — that are more meaningful from the point of view of the business logic.

##Hibernate #### JPA #### ORM ### Seam ########################## `LazyInitializationException` ## ##post-then-redirect ################################Seam ##Web ##

########

(IoC: Inversion of Control) #### ##### (DI: Dependency Injection) ########################JSF # EJB3 ################################## ## (### JSF ####)###

######## *(bijection)* ####### *(dynamic)* ############ *(contextual)* ########### *(bidirectional)* ######## IoC ##################(###############################)##

######################

Seam ## (#######) #######################Seam ##

XML##########

####Java##J2EE ###### ###########XML ###Java ##########################Java 5 #####################

EJB 3.0 ## ### (configuration by exception)########## ###### JSF ###### XML ################## Seam ## EJB 3.0 ### ###### ###### JSF ##Bean######### ###### XML ######### XML ######## (JSF #########) ############

#########

Seam ############ Java ##Java # Web ######################################Seam #################Seam ## (JSP ###### Facelets ###)## JUnit #### TestNG ############################### IDE #####################Seam # ##### JBoss ##### EJB ####################

#########

Java EE###(####GET ###### JSF ##########)#Seam ##########Seam ######JCP ##

Web##########*HTML*##################

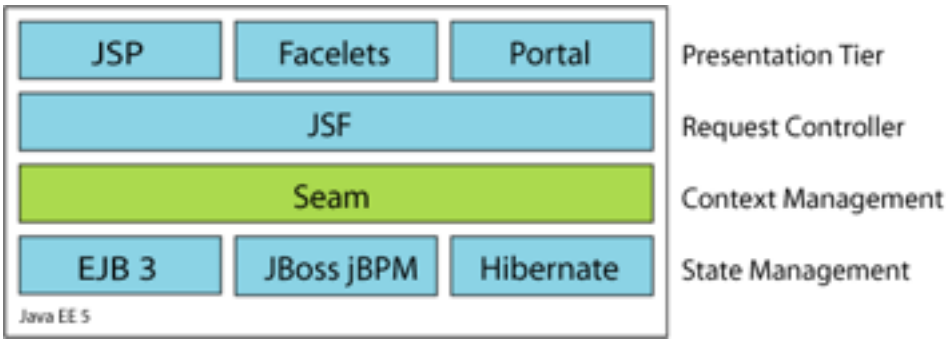
Web ########################Web ###########################Java #################################### Web ###E############PDF##################wiki #####Web ############################## Seam ####################################...

Seam######### JPA # Hibernate3 #################### EJB ######## Quartz########### jBPM############ JBoss Rules#E####### Meldware Mail# ############ Hibernate Search # Lucene########### JMS################### JBoss Cache #######Seam ##JAAS #JBoss Rules

##PDF ##################wiki ######## JSF #############Seam #########Web ################################ JavaScript #### Google Web Toolkit ######### JSF ###################

########## *!*

Seam #### Java EE ########################Tomcat ################### EJB 3.0 ## JPA #### Hibernate3 # Seam ###################################Tomcat ###### JBoss ########EJB 3.0 ######################



Seam # JSF # EJB3 ####### Java #### Web ##

1. Seam

Seam ########## *SeamFramework.org* [http://www.seamframework.org/Community/Contribute] ########

Seam

1.1. Seam

Seam # Seam ## ########################## Seam ################Seam ##### Seam ########### `examples` ################# ################`examples/registration` ###########

########################

- The `view` ########Web##
- `resources` ####################################
- `src` ##############################

######################## JBoss AS # Tomcat ####### ######################## Ant `build.xml` ########################## Ant ##########################

1.1.1. JBoss AS

The examples are configured for use on JBoss AS 4.2 or 5.0. You'll need to set `jboss.home`, in the shared `build.properties` file in the root folder of your Seam installation, to the location of your JBoss AS installation.

Once you've set the location of JBoss AS and started the application server, you can build and deploy any example by typing `ant explode` in the the directory for that example. Any example that is packaged as an EAR deploys to a URL like `/seam-example`, where *`example`* is the name of the example folder, with one exception. If the example folder begins with seam, the prefix "seam" is ommitted. For instance, if JBoss AS is running on port 8080, the URL for the registration example is *`http://localhost:8080/seam-registration/`* [http://localhost:8080/seam-registration/], whereas the URL for the seamspace example is *`http://localhost:8080/seam-space/`* [http://localhost:8080/seam-space/].

If, on the other hand, the example gets packaged as a WAR, then it deploys to a URL like `/jboss-seam-example`. Most of the examples can be deployed as a WAR to Tomcat with Embedded JBoss by typing `ant tomcat.deploy`. Several of the examples can only be deployed as a WAR. Those examples are groovybooking, hibernate, jpa, and spring.

1.1.2. Tomcat

Tomcat 6.0 ############Tomcat 6.0 #### JBoss######### *#30.6.1. #Embedded JBoss ##########* ###################### #### JBoss # Tomcat ## EJB3############ Seam ################## #### JBoss ###### Tomcat ####### non-EJB3

Seam ################## `build.properties` ###### `tomcat.home` # Tomcat ########################## Tomcat ####################

Tomcat ############## Ant ################### Tomcat ################# example ########## `ant tomcat.deploy` ##########

On Tomcat, the examples deploy to URLs like `/jboss-seam-example`, so for the registration example the URL would be *http://localhost:8080/jboss-seam-registration/* [http://localhost:8080/jboss-seam-registration/]. The same is true for examples that deploy as a WAR, as mentioned in the previous section.

1.1.3.

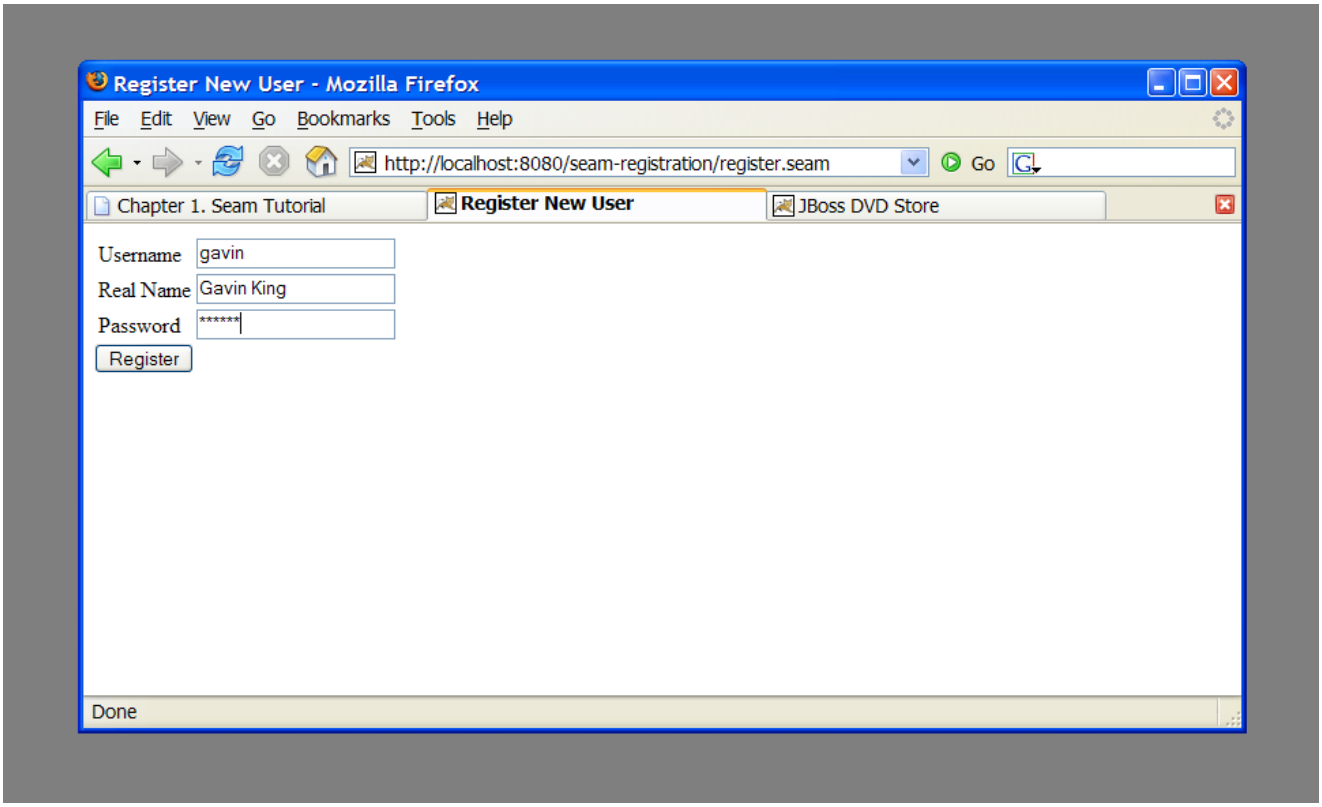
Most of the examples come with a suite of TestNG integration tests. The easiest way to run the tests is to run `ant test`. It is also possible to run the tests inside your IDE using the TestNG plugin. Consult the readme.txt in the examples directory of the Seam distribution for more information.

1.2. #### Seam ########:

################################## ### ################################## ######## Seam ######################### JSF ########### EJB3 ##### Bean ######### #### Seam ##############

EJB 3.0 ########################## ###########

###################################### ############################## ##############################



1.2.1.

########### Facelets ################ Bean ################ Bean ########## "bottom" ###############

1.2.1.1. ###### Bean: `User.java`

############EJB###### Bean###### ################### ### # ######## ############## Seam ##

 # 1.1. User.java

```
@Entity                                                    (1)
@Name("user")                                              (2)
@Scope(SESSION)                                            (3)
@Table(name="users")                                       (4)
public class User implements Serializable
{
  private static final long serialVersionUID = 1881413500711441951L;

  private String username;                                 (5)
  private String password;
  private String name;

  public User(String name, String password, String username)
  {
    this.name = name;
    this.password = password;
    this.username = username;
  }

  public User() {}                                         (6)

  @NotNull @Length(min=5, max=15)                          (7)
  public String getPassword()
  {
    return password;
  }

  public void setPassword(String password)
```

```
  {
    this.password = password;
  }

  @NotNull
  public String getName()
  {
    return name;
  }

  public void setName(String name)
  {
    this.name = name;
  }


  @Id @NotNull @Length(min=5, max=15)
  public String getUsername()
  {
    return username;
  }

  public void setUsername(String username)
  {
    this.username = username;
  }

}
```

1. EJB3 ## `@Entity` ######### `User` ########## Bean ##############
2. Seam ######## *`@Name`* ############## ######## ###### ##### Seam ########################## JSF # Seam # Seam ## (null) #### Seam ## ########## JSF #### `user` ############ Seam # `User` ############
3. Seam ######################## ########## ## ############ *`@Scope`* ################### `User` ######################
4. EJB ## `@Table` ######### `User` #### `users` ########################
5. `name#password# username` ## ###### Bean ######## ########################## ############################ JSF ##############################
6. ##########EJB # Seam ################

7 `@NotNull` # `@Length` ######### Hibernate Validator ############## Seam # Hibernate Validator ########## ########################## (#### Hiberenate ##################)#

8 EJB ## `@Id` ######### ###### Bean ##################

########################## `@Name` # `@Scope` ########## ################## Seam ######################

#####`User` ########## ## JSF ############### ############ JSF ################## JSP ########## Bean ####################################

Bean ################################ ########## JSF ###################### ##### ##### Bean

1.2.1.2. ########### Bean ###: `RegisterAction.java`

Seam ############### Bean # JSF ################## (###### JavaBean ########)

########## JSF ################ ########Bean ##### ########### ######### ################ `User` Bean ############ ########### Bean #########

############################

1.2. RegisterAction.java

```
@Stateless                                                    1
@Name("register")
public class RegisterAction implements Register
{
  @In
  private User user;                                          2

  @PersistenceContext
  private EntityManager em;                                   3

  @Logger
  private Log log;                                            4

  public String register()
  {                                                           5
    List existing = em.createQuery(
      "select username from User where username = #{user.username}")
```

```
        .getResultList();                                    (6)

    if (existing.size()==0)
    {
      em.persist(user);
      log.info("Registered new user #{user.username}");

      return "/registered.xhtml";                            (7)

    }                                                        (8)
    else
    {
      FacesMessages.instance().add("User #{user.username} already exists");

      return null;                                           (9)
    }
  }

}
```

1. EJB `@Stateless` ########################## Bean ##############
2. *`@In`* ######### Seam ############# Bean ############### ##########`user` (#########) ############################
3. EJB ## `@PersistenceContext` ######### EJB3 ###### Entity Manager ####################
4. Seam `@Logger` ################ `Log` ############################
5. ################################ ## EJB3 `EntityManager` API #####JSF ## (outcome) ###### ######## Bean #### `register()` ############ ##
6. Seam ## EJB-QL ## JSF EL ############## ###################### ######## JPA `setParameter()` ### JPA `Query` ################## ######?
7. The `Log` API lets us easily display templated log messages which can also make use of JSF EL expressions.
8. JSF ### (outcome) ###### null ## (outcome) (#####void ############) ## ############ ### JSF ### ## (outcome) ## JSF ### id ######### ## JSF ######### ############ ################################### ######################### Seam #### (outcome) ### JSF ### id ######### ###################### *## (outcome) ###### id ######## Seam ############################*
9. Seam provides a number of *built-in components* to help solve common problems. The `FacesMessages` component makes it easy to display templated error or success messages. (As of Seam 2.1, you can use `StatusMessages` instead to remove the semantic dependency on JSF). Built-in Seam components may be obtained by injection, or by calling the `instance()` method on the class of the built-in component.

####@Scope ######################## # Seam
Bean
########################## ###############

####### Bean ##
Seam############### ##### Web
############### Seam
###

Seam
###
############################

1.2.1.3. ##### Bean ########### : `Register.java`

######## Bean ####################

1.3. Register.java

```
@Local
public interface Register
{
  public String register();
}
```

Java ######### ## ##########

1.2.1.4. ### : `register.xhtml` # `registered.xhtml`

Seam ################## JSF ############################ #########JSP
############## Facelets #########

1.4. register.xhtml

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:s="http://jboss.com/products/seam/taglib"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:f="http://java.sun.com/jsf/core">

  <head>
    <title>Register New User</title>
```

```
  </head>
  <body>
    <f:view>
      <h:form>
        <s:validateAll>
          <h:panelGrid columns="2">
            Username: <h:inputText value="#{user.username}" required="true"/>
            Real Name: <h:inputText value="#{user.name}" required="true"/>
            Password: <h:inputSecret value="#{user.password}" required="true"/>
          </h:panelGrid>
        </s:validateAll>
        <h:messages/>
        <h:commandButton value="Register" action="#{register.register}"/>
      </h:form>
    </f:view>
  </body>

</html>
```

Seam ####### `<s:validateAll>` ####### ## JSF ######## ######################## Bean ###### Hibernate Validator ################## JSF

1.5. registered.xhtml

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
   xmlns:f="http://java.sun.com/jsf/core">

   <head>
      <title>Successfully Registered New User</title>
   </head>
   <body>
      <f:view>
         Welcome, #{user.name}, you are successfully registered as #{user.username}.
      </f:view>
   </body>

</html>
```

This is a simple Facelets page using some inline EL. There's nothing specific to Seam here.

1.2.1.5. Seam ################## : components.xml

Seam ######################### #######Seam ###################### ############Seam############################## ##############################

Java ##################### ############################## XML ## Seam ################## XML ################ ############### #### Seam ################ XML ########## ##### XML##################

(###Seam ##############)# ############## ############## WEB-INF ########## components.xml ###################### Seam # JNDI # EJB ###################### components.xml

1.6. components.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<components xmlns="http://jboss.com/products/seam/components"
   xmlns:core="http://jboss.com/products/seam/core"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="
      http://jboss.com/products/seam/core
      http://jboss.com/products/seam/core-2.2.xsd
      http://jboss.com/products/seam/components
      http://jboss.com/products/seam/components-2.2.xsd">

   <core:init jndi-pattern="@jndiPattern@"/>

</components>
```

This code configures a property named `jndiPattern` of a built-in Seam component named `org.jboss.seam.core.init`. The funny `@` symbols are there because our Ant build script puts the correct JNDI pattern in when we deploy the application, which it reads from the components.properties file. You learn more about how this process works in *#5.2. #components.xml##############*.

1.2.1.6. WEB ########## : web.xml

######################## WAR ########## ######Web ################

1.7. web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="
      http://java.sun.com/xml/ns/javaee
      http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
   version="2.5">

   <listener>
      <listener-class>org.jboss.seam.servlet.SeamListener</listener-class>
   </listener>

   <context-param>
      <param-name>javax.faces.DEFAULT_SUFFIX</param-name>
      <param-value>.xhtml</param-value>
   </context-param>

   <servlet>
      <servlet-name>Faces Servlet</servlet-name>
      <servlet-class>javax.faces.webapp.FacesServlet</servlet-class>
      <load-on-startup>1</load-on-startup>
   </servlet>

   <servlet-mapping>
      <servlet-name>Faces Servlet</servlet-name>
      <url-pattern>*.seam</url-pattern>
   </servlet-mapping>

   <session-config>
      <session-timeout>10</session-timeout>
   </session-config>

</web-app>
```

`web.xml` ##### Seam # JSF ####### ######## Seam

1.2.1.7. JSF ## : `faces-config.xml`

Seam ###################### JSF ########## ##### `faces-config.xml` ###### ########## Facelets #########JSF ############## Faceles

1.8. faces-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<faces-config xmlns="http://java.sun.com/xml/ns/javaee"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="
      http://java.sun.com/xml/ns/javaee
      http://java.sun.com/xml/ns/javaee/web-facesconfig_1_2.xsd"
   version="1.2">

   <application>
      <view-handler>com.sun.facelets.FaceletViewHandler</view-handler>
   </application>

</faces-config>
```

Note that we don't need any JSF managed bean declarations! Our managed beans are annotated Seam components. In Seam applications, the `faces-config.xml` is used much less often than in plain JSF. Here, we are simply using it to enable Facelets as the view handler instead of JSP.

In fact, once you have all the basic descriptors set up, the *only* XML you need to write as you add new functionality to a Seam application is orchestration: navigation rules or jBPM process definitions. Seam's stand is that *process flow* and *configuration data* are the only things that truly belong in XML.

############ ### id ################## #################

1.2.1.8. EJB ########## : `ejb-jar.xml`

`ejb-jar.xml` ###### ################ Bean # `SeamInterceptor` ########## EJB3 #######

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar xmlns="http://java.sun.com/xml/ns/javaee"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="
      http://java.sun.com/xml/ns/javaee
      http://java.sun.com/xml/ns/javaee/ejb-jar_3_0.xsd"
   version="3.0">

   <interceptors>
      <interceptor>
         <interceptor-class>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
      </interceptor>
```

```
    </interceptors>

    <assembly-descriptor>
        <interceptor-binding>
            <ejb-name>*</ejb-name>
            <interceptor-class>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
        </interceptor-binding>
    </assembly-descriptor>

</ejb-jar>
```

1.2.1.9. EJB ############ : `persistence.xml`

`persistence.xml` ######EJB ######### ###################################### ###################################

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence xmlns="http://java.sun.com/xml/ns/persistence"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="
        http://java.sun.com/xml/ns/persistence
        http://java.sun.com/xml/ns/persistence/persistence_1_0.xsd"
    version="1.0">

    <persistence-unit name="userDatabase">
        <provider>org.hibernate.ejb.HibernatePersistence</provider>
        <jta-data-source>java:/DefaultDS</jta-data-source>
        <properties>
            <property name="hibernate.hbm2ddl.auto" value="create-drop"/>
        </properties>
    </persistence-unit>

</persistence>
```

1.2.1.10. EAR ########## : `application.xml`

####EAR### ######################################

1.9.

```
<?xml version="1.0" encoding="UTF-8"?>
<application xmlns="http://java.sun.com/xml/ns/javaee"
```

```
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="
      http://java.sun.com/xml/ns/javaee
      http://java.sun.com/xml/ns/javaee/application_5.xsd"
   version="5">

   <display-name>Seam Registration</display-name>

   <module>
      <web>
         <web-uri>jboss-seam-registration.war</web-uri>
         <context-root>/seam-registration</context-root>
      </web>
   </module>
   <module>
      <ejb>jboss-seam-registration.jar</ejb>
   </module>
   <module>
      <ejb>jboss-seam.jar</ejb>
   </module>
   <module>
      <java>jboss-el.jar</java>
   </module>

</application>
```

WEB################## `/seam-registration`

##################################

1.2.2.

################ JSF ##Seam # `user` ################### ###################(## Seam ########)# Seam ##`user` ################ ### Seam ################## `User` ###### Bean ####### JSF ######

########## `User` ########### Hibernate Validator ############################ JSF ############## ###### ######### `User` ###### Bean ###############

Next, JSF asks Seam to resolve the variable named `register`. Seam uses the JNDI pattern mentioned earlier to locate the stateless session bean, wraps it as a Seam component, and returns it. Seam then presents this component to JSF and JSF invokes the `register()` action listener method.

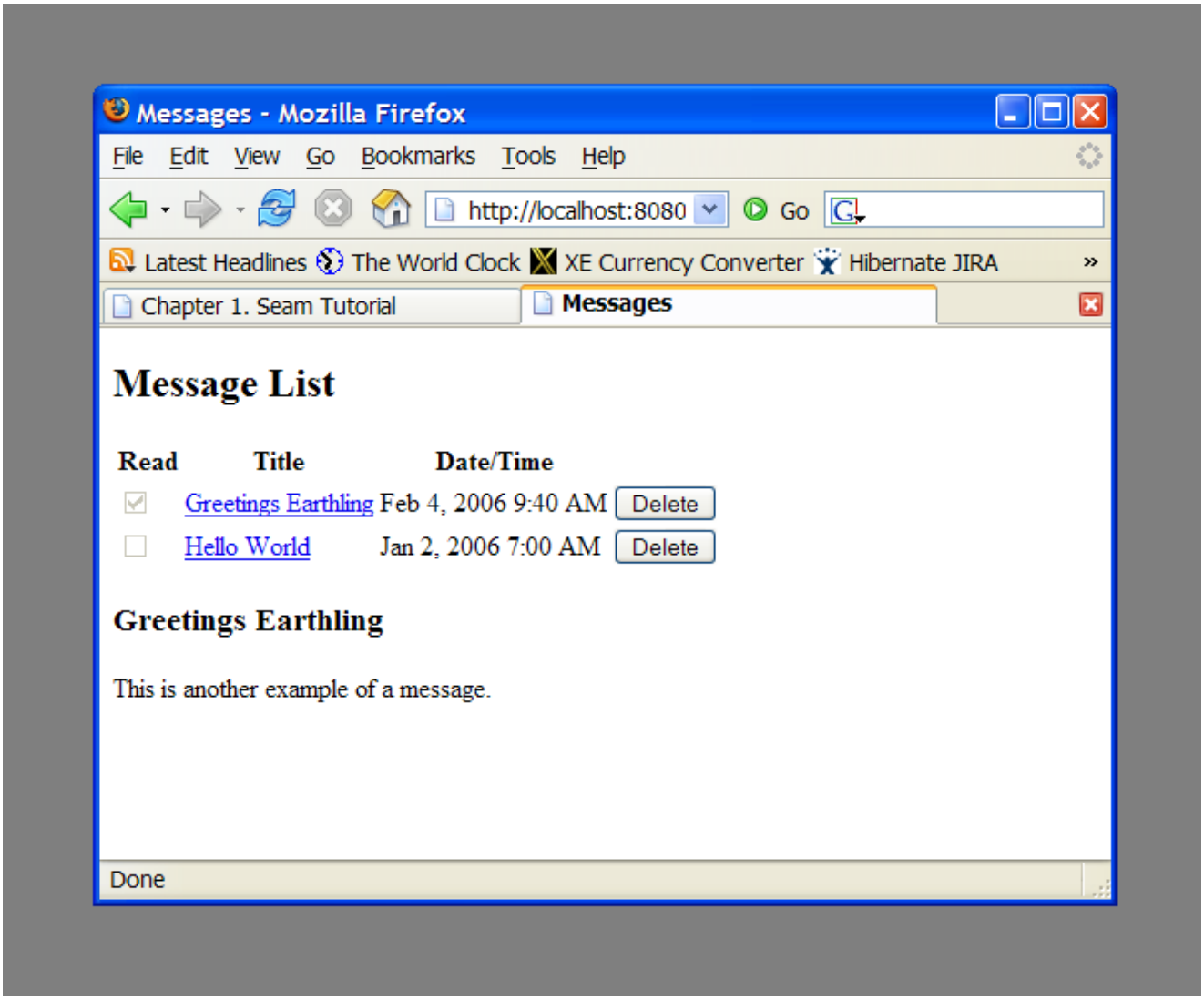
But Seam is not done yet. Seam intercepts the method call and injects the `User` entity from the Seam session context, before allowing the invocation to continue.

`register()` ################################ ####### ######### `FacesMessages` ################ null ## (outcome) ######################## `FacesMessages` ######################## JSF ###### ### # JSF `FacesMessage` #######

####################`"/registered.xhtml"` ## (outcome) ### `registered.xhtml` ###################### JSF ################## Seam # `user` ################ Seam ################ `User` ##################

1.3. Seam ###########:

############################ ################################Seam ##EJB-QL ###HQL ################## #####JSF `<h:dataTable>` ################################ JSF ############## ########################



1.3.1.

########## ######### Bean ### `Message`# ######## Bean ### `MessageListBean`# ###### JSP ##########

1.3.1.1. ###### Bean : `Message.java`

`Message` ###### Bean ## ######################### ###############################

 # 1.10. Message.java

```
@Entity
@Name("message")
@Scope(EVENT)
public class Message implements Serializable
{
  private Long id;
  private String title;
  private String text;
  private boolean read;
  private Date datetime;

  @Id @GeneratedValue
  public Long getId()
  {
    return id;
  }
  public void setId(Long id)
  {
    this.id = id;
  }

  @NotNull @Length(max=100)
  public String getTitle()
  {
    return title;
  }
  public void setTitle(String title)
  {
    this.title = title;
  }

  @NotNull @Lob
  public String getText()
```

```
  {
    return text;
  }
  public void setText(String text)
  {
    this.text = text;
  }

  @NotNull
  public boolean isRead()
  {
    return read;
  }
  public void setRead(boolean read)
  {
    this.read = read;
  }

  @NotNull
  @Basic @Temporal(TemporalType.TIMESTAMP)
  public Date getDatetime()
  {
    return datetime;
  }
  public void setDatetime(Date datetime)
  {
    this.datetime = datetime;
  }

}
```

1.3.1.2. ########### Bean : `MessageManagerBean.java`

########## ######## Bean `MessageManagerBean` ######
######################
###################### ############################

But `MessageManagerBean` is also responsible for fetching the list of messages the first time we navigate to the message list page. There are various ways the user could navigate to the page, and not all of them are preceded by a JSF action — the user might have bookmarked the page, for example. So the job of fetching the message list takes place in a Seam *factory method*, instead of in an action listener method.

################################### ########### Bean #########

1.11. MessageManagerBean.java

```
@Stateful
@Scope(SESSION)
@Name("messageManager")
public class MessageManagerBean implements Serializable, MessageManager
{
  @DataModel
  private List<Message> messageList;                    (1)

  @DataModelSelection
  @Out(required=false)                                  (2)
  private Message message;                              (3)

  @PersistenceContext(type=EXTENDED)
  private EntityManager em;                             (4)

  @Factory("messageList")
  public void findMessages()                            (5)
  {
    messageList = em.createQuery("select msg from Message msg order by msg.datetime desc")
                    .getResultList();
  }

  public void select()
  {                                                     (6)
    message.setRead(true);
  }

  public void delete()
  {                                                     (7)
    messageList.remove(message);
    em.remove(message);
    message=null;
  }

  @Remove
  public void destroy() {}                              (8)
```

```
}
```

1. `@DataModel` ######### `java.util.List` ######## `javax.faces.model.DataModel` ######### JSF ########## ######################## JSF `<h:dataTable>` ############## ######### `DataModel` ## `messageList` ##############################
2. `@DataModelSelection` ######### Seam ################ `List` ####################
3. The `@Out` annotation then exposes the selected value directly to the page. So every time a row of the clickable list is selected, the `Message` is injected to the attribute of the stateful bean, and the subsequently *outjected* to the event context variable named `message`.
4. ######## Bean ##EJB3 ########## ######## ## Bean ######## ######### messages ################## #### ########### Bean ############ #### `EntityManager` ########## ############
5. ### JSP ############# `messageList` ################### `@Factory` #########Seam # `MessageManagerBean` ############## ########## `findMessages()` ############ `findMessages()` # `messages` # ###############
6. `select()` ############### ##### `Message`# ## ####### ################
7. `delete()` ############### ##### `Message` ###############
8. ############### Bean # Seam #########`@Remove` ################################## #####Seam ################################## Seam ####### Bean #########

############## Seam ######################## ## ######################## (Seam ##)

1.3.1.3. ##### Bean ############ : `MessageManager.java`

############### Bean ##############

1.12. MessageManager.java

```
@Local
public interface MessageManager
{
  public void findMessages();
  public void select();
  public void delete();
  public void destroy();
}
```

##

components.xml#persistence.xml#web.xml#ejb-jar.xml#faces-config.xml ###
application.xml ########################## JSP ########

1.3.1.4. ###: messages.jsp

##JSP#### JSF <h:dataTable> ##################### Seam ###############

1.13. messages.jsp

```
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<html>
 <head>
  <title
>Messages</title>
 </head>
 <body>
  <f:view>
   <h:form>
    <h2
>Message List</h2>
    <h:outputText value="No messages to display"
                  rendered="#{messageList.rowCount==0}"/>
    <h:dataTable var="msg" value="#{messageList}"
                 rendered="#{messageList.rowCount
>0}">
      <h:column>
        <f:facet name="header">
          <h:outputText value="Read"/>
        </f:facet>
        <h:selectBooleanCheckbox value="#{msg.read}" disabled="true"/>
      </h:column>
      <h:column>
        <f:facet name="header">
          <h:outputText value="Title"/>
        </f:facet>
        <h:commandLink value="#{msg.title}" action="#{messageManager.select}"/>
      </h:column>
      <h:column>
        <f:facet name="header">
          <h:outputText value="Date/Time"/>
        </f:facet>
        <h:outputText value="#{msg.datetime}">
```

```
            <f:convertDateTime type="both" dateStyle="medium" timeStyle="short"/>
          </h:outputText>
        </h:column>
        <h:column>
          <h:commandButton value="Delete" action="#{messageManager.delete}"/>
        </h:column>
      </h:dataTable>
      <h3
><h:outputText value="#{message.title}"/></h3>
      <div
><h:outputText value="#{message.text}"/></div>
    </h:form>
  </f:view>
 </body>
</html
>
```

1.3.2.

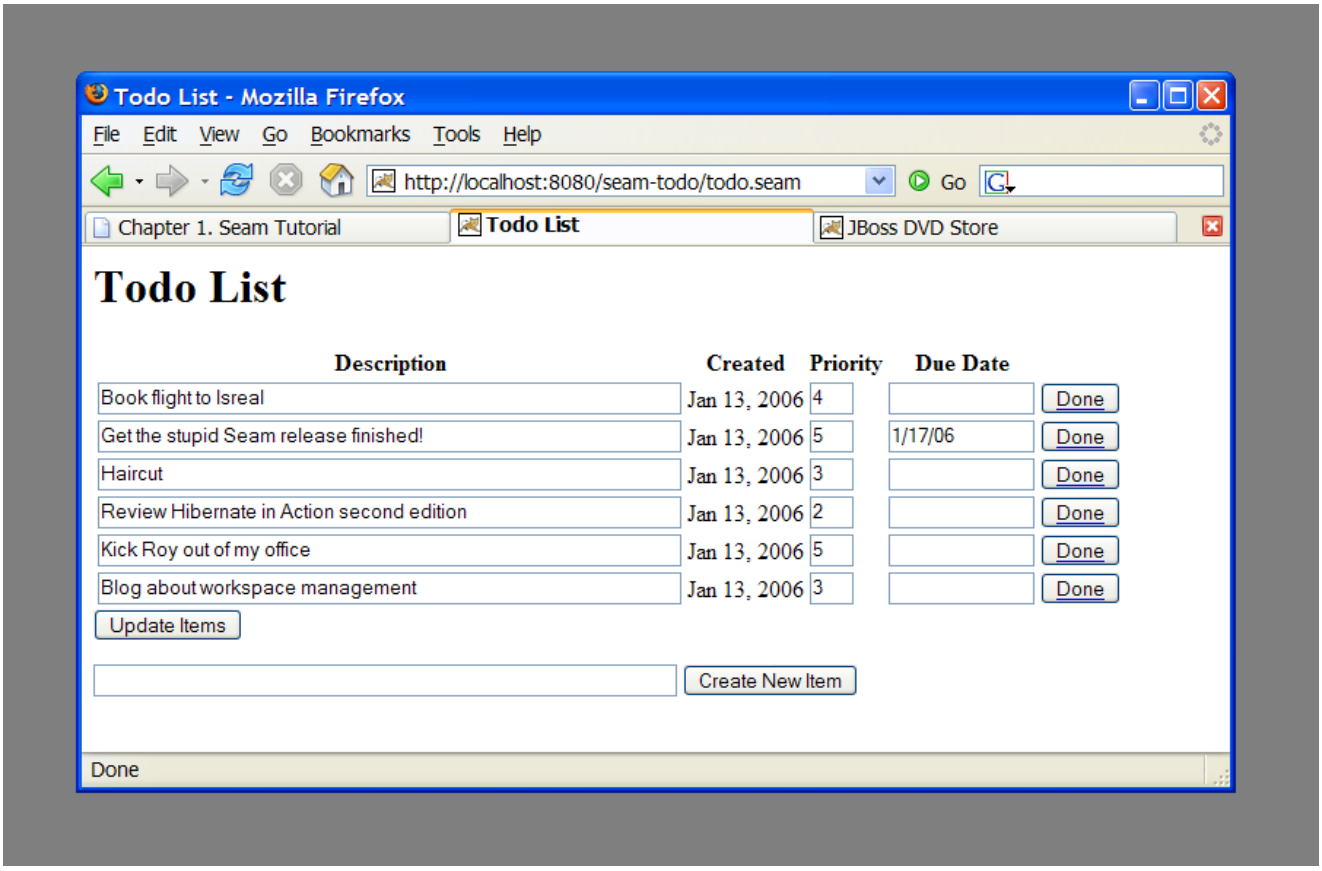
messages.jsp ################## messageList ##################### ######################## Seam ########## findMessages()######## ################## ####### (outject) ### DataModel ########### ## DataModel # <h:dataTable>

<h:commandLink> ######### JSF # select() ################# Seam ########################## messageManager ######## message ############# ################ ## Message ############ ########## Seam #### Message # message ########################## ### EJB ################### Message ######################## #### ##############################

<h:commandButton> ######### JSF ##delete() ################# Seam ################ ########## messageList ######## message ############# ############## ## Message ########## EntityManager # remove() ######### ########## Seam # messageList ############ message ##################### EJB # ################## ######## Message ####### #### ################

1.4. Seam # jBPM : TO-DO

jBPM ############################## ##### jBPM # Seam ################ ### To-Do ################### ###############jBPM ########### ######## Java ##############



1.4.1.

############jBPM ########## ### JSP ########## JavaBean ###### (############################ ##### Bean ###############) ####################

 # 1.14. todo.jpdl.xml

```
<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"/>                                    (5)

</process-definition
>
```

(1) `<start-state>` ###################### ############ ###### `todo` #######

(2) `<task-node>` ######### ###### ##

(3) `<task>` ############################ ########################## ################ ############## ####### `todoList` ###### Seam ####### (JavaBean # ##) ## description #######

(4) ## ################################## ######## `actor` ########## Seam ############## ##### Seam ####################################

(5) `<end-state>`########################## ################ ##################

JBossIDE ################################## ###########

##################### ######## ######## ################################### ### ######### ########## ###############

JavaBean ####### login.jsp ####### ##### actor ############ jBPM actor id ###########

1.15. Login.java

```
@Name("login")
public class Login
{
  @In
  private Actor actor;

  private String user;

  public String getUser()
  {
    return user;
  }

  public void setUser(String user)
  {
    this.user = user;
  }

  public String login()
  {
    actor.setId(user);
    return "/todo.jsp";
  }
}
```

######### Actor ###################### @In ################

JSP

1.16. login.jsp

```
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h"%>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f"%>
<html>
<head>
<title
>Login</title>
</head>
```

```
<body>
<h1
>Login</h1>
<f:view>
   <h:form>
     <div>
       <h:inputText value="#{login.user}"/>
       <h:commandButton value="Login" action="#{login.login}"/>
     </div>
   </h:form>
</f:view>
</body>
</html
>
```

JavaBean

1.17. TodoList.java

```
@Name("todoList")
public class TodoList
{
  private String description;


  public String getDescription()                                    (1)
  {
    return description;
  }

  public void setDescription(String description)
  {
    this.description = description;
  }
                                                                    (2)
  @CreateProcess(definition="todo")
  public void createTodo() {}
                                                                    (3)
  @StartTask @EndTask
  public void done() {}
```

```
}
```

1. The description property accepts user input from the JSP page, and exposes it to the process definition, allowing the task description to be set.
2. Seam `@CreateProcess` ######################## jBPM ##################
3. Seam `@StartTask` ######################## `@EndTask` ##############################

############ `@StartTask` # `@EndTask` ################## ##

#################### `todo.jsp` ######

1.18. todo.jsp

```
<%@ taglib uri="http://java.sun.com/jsf/html" prefix="h" %>
<%@ taglib uri="http://java.sun.com/jsf/core" prefix="f" %>
<%@ taglib uri="http://jboss.com/products/seam/taglib" prefix="s" %>
<html>
<head>
<title
>Todo List</title>
</head>
<body>
<h1
>Todo List</h1>
<f:view>
  <h:form id="list">
    <div>
      <h:outputText value="There are no todo items."
                    rendered="#{empty taskInstanceList}"/>
      <h:dataTable value="#{taskInstanceList}" var="task"
                   rendered="#{not empty taskInstanceList}">
        <h:column>
          <f:facet name="header">
            <h:outputText value="Description"/>
          </f:facet>
          <h:inputText value="#{task.description}"/>
        </h:column>
        <h:column>
          <f:facet name="header">
            <h:outputText value="Created"/>
          </f:facet>
          <h:outputText value="#{task.taskMgmtInstance.processInstance.start}">
```

```
                <f:convertDateTime type="date"/>
            </h:outputText>
        </h:column>
        <h:column>
            <f:facet name="header">
                <h:outputText value="Priority"/>
            </f:facet>
            <h:inputText value="#{task.priority}" style="width: 30"/>
        </h:column>
        <h:column>
            <f:facet name="header">
                <h:outputText value="Due Date"/>
            </f:facet>
            <h:inputText value="#{task.dueDate}" style="width: 100">
                <f:convertDateTime type="date" dateStyle="short"/>
            </h:inputText>
        </h:column>
        <h:column>
            <s:button value="Done" action="#{todoList.done}" taskInstance="#{task}"/>
        </h:column>
      </h:dataTable>
    </div>
    <div>
    <h:messages/>
    </div>
    <div>
      <h:commandButton value="Update Items" action="update"/>
    </div>
  </h:form>
  <h:form id="new">
    <div>
      <h:inputText value="#{todoList.description}"/>
      <h:commandButton value="Create New Item" action="#{todoList.createTodo}"/>
    </div>
  </h:form>
</f:view>
</body>
</html
>
```

```
#############

###################### ####taskInstanceList ##### Seam ##################
#####JSF################
```

1.19. todo.jsp

```
<h:form id="list">
  <div>
    <h:outputText value="There are no todo items." rendered="#{empty taskInstanceList}"/>
    <h:dataTable value="#{taskInstanceList}" var="task"
                 rendered="#{not empty taskInstanceList}">
      ...
    </h:dataTable>
  </div>
</h:form
>
```

####### jBPM ### `TaskInstance` ########## ## #### (Description) # ### (Priority) ## #### (Due Date) ###### ##################################

```
<h:column>
   <f:facet name="header">
     <h:outputText value="Description"/>
   </f:facet>
   <h:inputText value="#{task.description}"/>
</h:column>
<h:column>
   <f:facet name="header">
      <h:outputText value="Created"/>
   </f:facet>
   <h:outputText value="#{task.taskMgmtInstance.processInstance.start}">
      <f:convertDateTime type="date"/>
   </h:outputText>
</h:column>
<h:column>
   <f:facet name="header">
      <h:outputText value="Priority"/>
   </f:facet>
   <h:inputText value="#{task.priority}" style="width: 30"/>
</h:column>
<h:column>
   <f:facet name="header">
      <h:outputText value="Due Date"/>
   </f:facet>
   <h:inputText value="#{task.dueDate}" style="width: 100">
```

```
      <f:convertDateTime type="date" dateStyle="short"/>
   </h:inputText>
</h:column
>
```

##

Seam provides a default JSF date converter for converting a string to a date (no time). Thus, the converter is not necessary for the field bound to `#{task.dueDate}`.

`#######` `@StartTask` `@EndTask` `#####################################` ####task id ########## Seam ######

```
<h:column>
   <s:button value="Done" action="#{todoList.done}" taskInstance="#{task}"/>
</h:column
>
```

`###` `seam-ui.jar` `#######` Seam `<s:button>` JSF `##########################` `##############################` ################Seam # jBPM ################ `##########################`

```
<h:commandButton value="Update Items" action="update"/>
```

`####` `################################` `@CreateProcess########################################`

```
<h:form id="new">
   <div>
      <h:inputText value="#{todoList.description}"/>
      <h:commandButton value="Create New Item" action="#{todoList.createTodo}"/>
   </div>
</h:form
>
```

1.4.2.

######todo.jsp ################ To-Do ######### taskInstanceList ## todo ########"Create New Item" ############{todoList.createTodo} ########## todo.jpdl.xml ################

###################start ########## todo ########################## ############################## #{todoList.description} ######## ############################## Seam # actor ############### ## ################################ ##############################

todo.jsp ##########taskInstanceList ################### #### h:dataTable ######## ######### #{task.description}# #{task.priority}# #{task.dueDate} ############## ############################

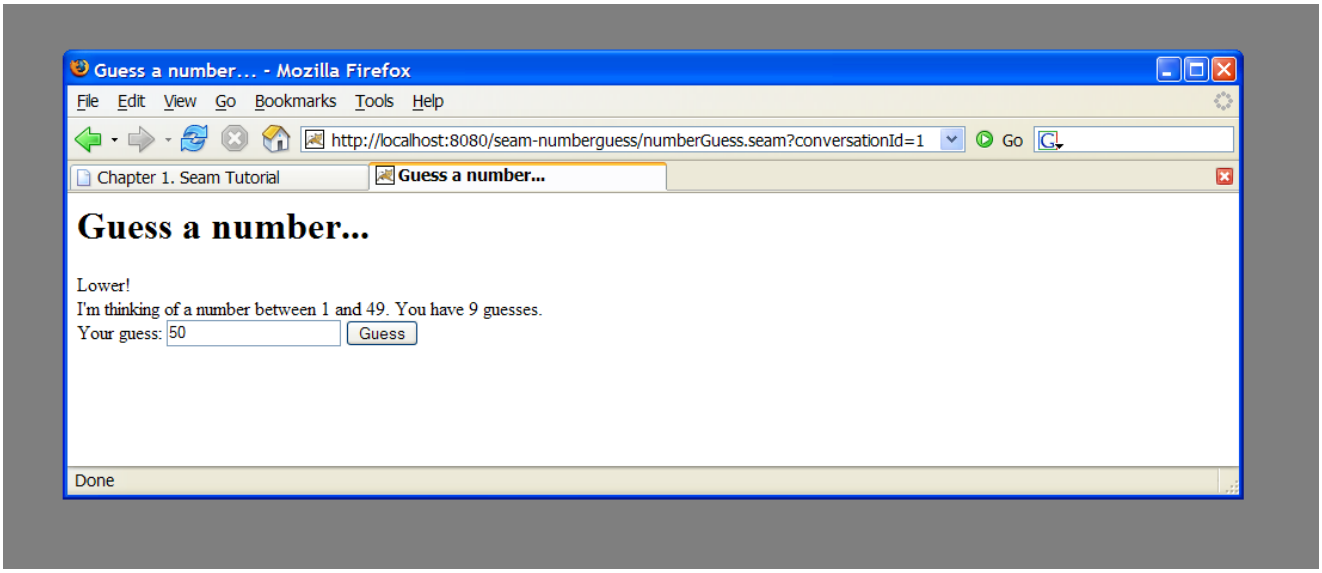
#To-Do### "Done" ############ #{todoList.done} ######## todoList ########## s:button # taskInstance="#{task}" ## @StartTast # @EndTask ######## ################################## ####################### done ######### ############ ################################

todo.jsp ###################### taskInstanceList ########## ##

1.5. Seam ######:

(#####) ######## Seam ############ JSF/Seam ################################## ############################ ########################## ## ########### ####### ######

Seam ##jPDL ############################ ##



1.5.1.

######## ###JavaBean#### JSP ######## jPDL ###################### ################

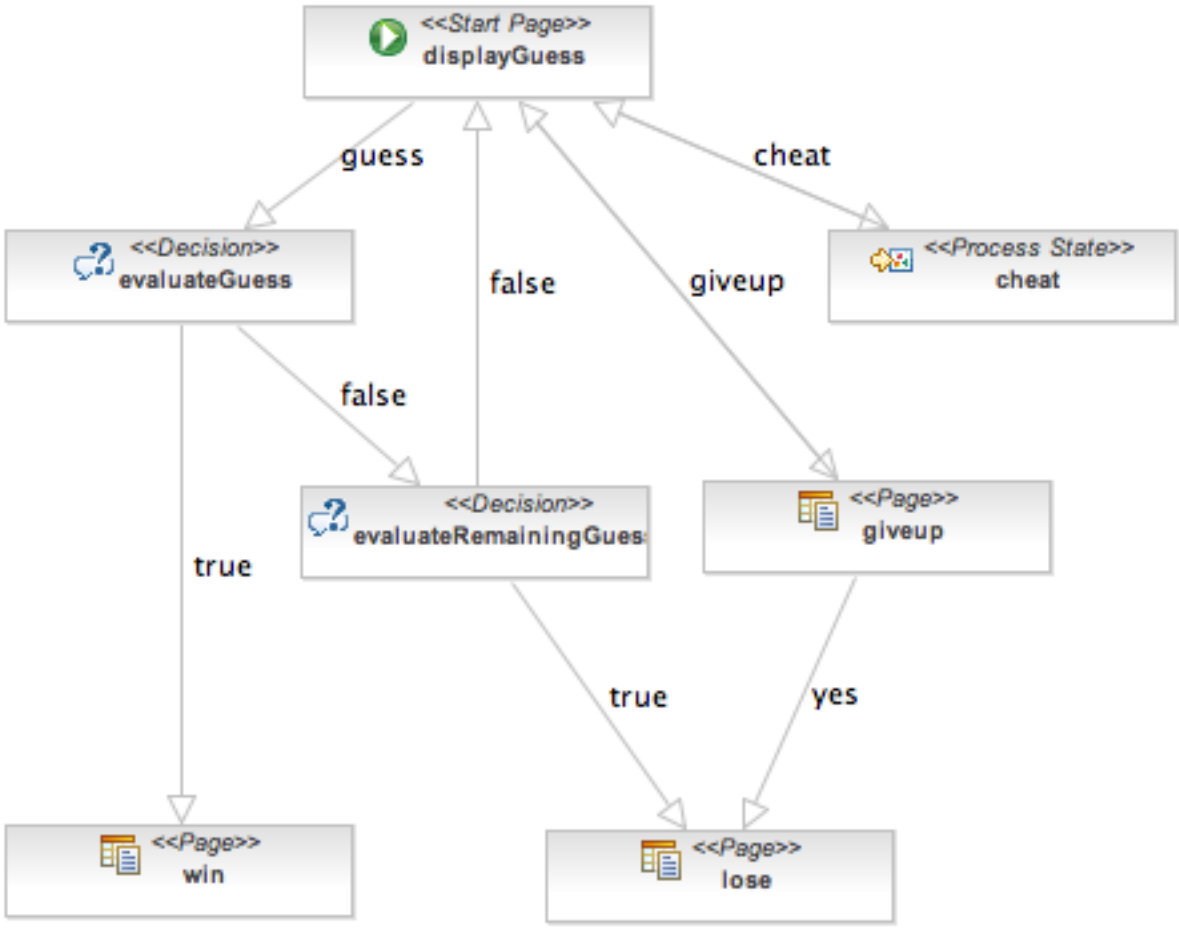
1.20. pageflow.jpdl.xml

```
<pageflow-definition
      xmlns="http://jboss.com/products/seam/pageflow"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://jboss.com/products/seam/pageflow
                  http://jboss.com/products/seam/pageflow-2.2.xsd"
      name="numberGuess">


  <start-page name="displayGuess" view-id="/numberGuess.jspx">                    (1)
    <redirect/>

    <transition name="guess" to="evaluateGuess">                                  (2)

      <action expression="#{numberGuess.guess}"/>                                 (3)
    </transition>
    <transition name="giveup" to="giveup"/>
    <transition name="cheat" to="cheat"/>
  </start-page>

                                                                                  (4)
  <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="giveup" view-id="/giveup.jspx">
    <redirect/>
    <transition name="yes" to="lose"/>
    <transition name="no" to="displayGuess"/>
  </page>

  <process-state name="cheat">
    <sub-process name="cheat"/>
    <transition to="displayGuess"/>
  </process-state>

  <page name="win" view-id="/win.jspx">
    <redirect/>
    <end-conversation/>
  </page>

  <page name="lose" view-id="/lose.jspx">
    <redirect/>
    <end-conversation/>
  </page>

</pageflow-definition>
```

1. `<page>` ################## ############# JSF ####################### `view-id` #### JSF ################## JSF ### ######## ############## `redirect` ####Seam # post-then-redirect ############ (######### URL ########)
2. `<transition>` ### JSF ## (outcome) ######### JSF ########## (outcome) ####### transition ######## jBPM ################ ########################
3. transition # `<action>` ## jBPM # transition############## JSF ########## ############# Seam ####################
4. `<decision>` ################ JSF EL ##############################

JBoss Developer Studio



################ ##############################

################## `numberGuess.jspx` ###

1.21. numberGuess.jspx

```
<<?xml version="1.0"?>
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
        xmlns:h="http://java.sun.com/jsf/html"
        xmlns:f="http://java.sun.com/jsf/core"
        xmlns:s="http://jboss.com/products/seam/taglib"
        xmlns="http://www.w3.org/1999/xhtml"
        version="2.0">
  <jsp:output doctype-root-element="html"
            doctype-public="-//W3C//DTD XHTML 1.0 Transitional//EN"
            doctype-system="http://www.w3c.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"/>
  <jsp:directive.page contentType="text/html"/>
```

```
<html>
<head>
  <title>Guess a number...</title>
  <link href="niceforms.css" rel="stylesheet" type="text/css" />
  <script language="javascript" type="text/javascript" src="niceforms.js" />
</head>
<body>
  <h1>Guess a number...</h1>
  <f:view>
    <h:form styleClass="niceform">

      <div>
      <h:messages globalOnly="true"/>
      <h:outputText value="Higher!"
            rendered="#{numberGuess.randomNumber gt numberGuess.currentGuess}"/>
      <h:outputText value="Lower!"
            rendered="#{numberGuess.randomNumber lt numberGuess.currentGuess}"/>
      </div>

      <div>
      I'm thinking of a number between
      <h:outputText value="#{numberGuess.smallest}"/> and
      <h:outputText value="#{numberGuess.biggest}"/>. You have
      <h:outputText value="#{numberGuess.remainingGuesses}"/> guesses.
      </div>

      <div>
      Your guess:
      <h:inputText value="#{numberGuess.currentGuess}" id="inputGuess"
                required="true" size="3"
                rendered="#{(numberGuess.biggest-numberGuess.smallest) gt 20}">
        <f:validateLongRange maximum="#{numberGuess.biggest}"
                        minimum="#{numberGuess.smallest}"/>
      </h:inputText>
      <h:selectOneMenu value="#{numberGuess.currentGuess}"
                    id="selectGuessMenu" required="true"
                    rendered="#{(numberGuess.biggest-numberGuess.smallest) le 20 and
                              (numberGuess.biggest-numberGuess.smallest) gt 4}">
        <s:selectItems value="#{numberGuess.possibilities}" var="i" label="#{i}"/>
      </h:selectOneMenu>
      <h:selectOneRadio value="#{numberGuess.currentGuess}" id="selectGuessRadio"
                    required="true"
                    rendered="#{(numberGuess.biggest-numberGuess.smallest) le 4}">
        <s:selectItems value="#{numberGuess.possibilities}" var="i" label="#{i}"/>
```

```
      </h:selectOneRadio>
      <h:commandButton value="Guess" action="guess"/>
      <s:button value="Cheat" view="/confirm.jspx"/>
      <s:button value="Give up" action="giveup"/>
      </div>

      <div>
      <h:message for="inputGuess" style="color: red"/>
      </div>

    </h:form>
  </f:view>
 </body>
 </html>
</jsp:root>
```

################## ##############guess transition###################

win.jspx ###############

1.22. win.jspx

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
        xmlns:h="http://java.sun.com/jsf/html"
        xmlns:f="http://java.sun.com/jsf/core"
        xmlns="http://www.w3.org/1999/xhtml"
        version="2.0">
 <jsp:output doctype-root-element="html"
           doctype-public="-//W3C//DTD XHTML 1.0 Transitional//EN"
           doctype-system="http://www.w3c.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"/>
 <jsp:directive.page contentType="text/html"/>
 <html>
 <head>
   <title
>You won!</title>
   <link href="niceforms.css" rel="stylesheet" type="text/css" />
 </head>
 <body>
   <h1
>You won!</h1>
   <f:view>
     Yes, the answer was <h:outputText value="#{numberGuess.currentGuess}" />.
     It took you <h:outputText value="#{numberGuess.guessCount}" /> guesses.
     <h:outputText value="But you cheated, so it doesn't count!"
```

```
                rendered="#{numberGuess.cheat}"/>
    Would you like to <a href="numberGuess.seam"
>play again</a
>?
   </f:view>
  </body>
  </html>
</jsp:root>
```

`lose.jspx` ######## #########

##########################

1.23. NumberGuess.java

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

  private int randomNumber;
  private Integer currentGuess;
  private int biggest;
  private int smallest;
  private int guessCount;
  private int maxGuesses;
  private boolean cheated;


  @Create                                                    (1)
  public void begin()
  {
    randomNumber = new Random().nextInt(100);
    guessCount = 0;
    biggest = 100;
    smallest = 1;
  }

  public void setCurrentGuess(Integer guess)
  {
    this.currentGuess = guess;
  }

  public Integer getCurrentGuess()
  {
```

```
    return currentGuess;
  }

  public void guess()
  {
    if (currentGuess
>randomNumber)
    {
      biggest = currentGuess - 1;
    }
    if (currentGuess<randomNumber)
    {
      smallest = currentGuess + 1;
    }
    guessCount ++;
  }

  public boolean isCorrectGuess()
  {
    return currentGuess==randomNumber;
  }

  public int getBiggest()
  {
    return biggest;
  }

  public int getSmallest()
  {
    return smallest;
  }

  public int getGuessCount()
  {
    return guessCount;
  }

  public boolean isLastGuess()
  {
    return guessCount==maxGuesses;
  }

  public int getRemainingGuesses() {
    return maxGuesses-guessCount;
```

```
  }

  public void setMaxGuesses(int maxGuesses) {
    this.maxGuesses = maxGuesses;
  }

  public int getMaxGuesses() {
    return maxGuesses;
  }

  public int getRandomNumber() {
    return randomNumber;
  }

  public void cheated()
  {
    cheated = true;
  }

  public boolean isCheat() {
    return cheated;
  }

  public List<Integer
> getPossibilities()
  {
    List<Integer
> result = new ArrayList<Integer
>();
    for(int i=smallest; i<=biggest; i++) result.add(i);
    return result;
  }

}
```

1 ####JSP #### `numberGuess` ################ Seam ################## ####`@Create` ######### ######################

`pages.xml` ##### Seam ## *(conversation)* #### (#####)##################################

1.24. pages.xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<pages xmlns="http://jboss.com/products/seam/pages"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://jboss.com/products/seam/pages  http://jboss.com/products/seam/pages-2.2.xsd">

  <page view-id="/numberGuess.jspx">
    <begin-conversation join="true" pageflow="numberGuess"/>
  </page>

</pages>
```

########## Seam ####################! ################################## ############################

1.5.2.

######################## #### numberGuess.jspx ######## ##############pages.xml ########## numberGuess ################ ####### #######start-page ######### numberGuess.xhtml ########

numberGuess ############## ############################ @Create ######################## ######### #{numberGuess.currentGuess} ###### h:form

"Guess" #### guess ############## Seam ################################## evaluateGuess ############### ####### #{numberGuess.guess} # guess count # numberGuess ######### highest/lowest suggestions #######

evaluateGuess ### #{numberGuess.correctGuess} ######## win ### evaluatingRemainingGuesses ########## ######################## evaluatingRemainingGuesses ####### ### decision ###### ######################## #{numberGuess.lastGuess} ######## ############ (lastGuess # false)#### displayGuess ########## ####page ############# /numberGuess.jspx ########### redirect ###########Seam ##################################

######## win ### lose ################################## / win.jspx ### /lose.jspx ############## ###### Seam ##

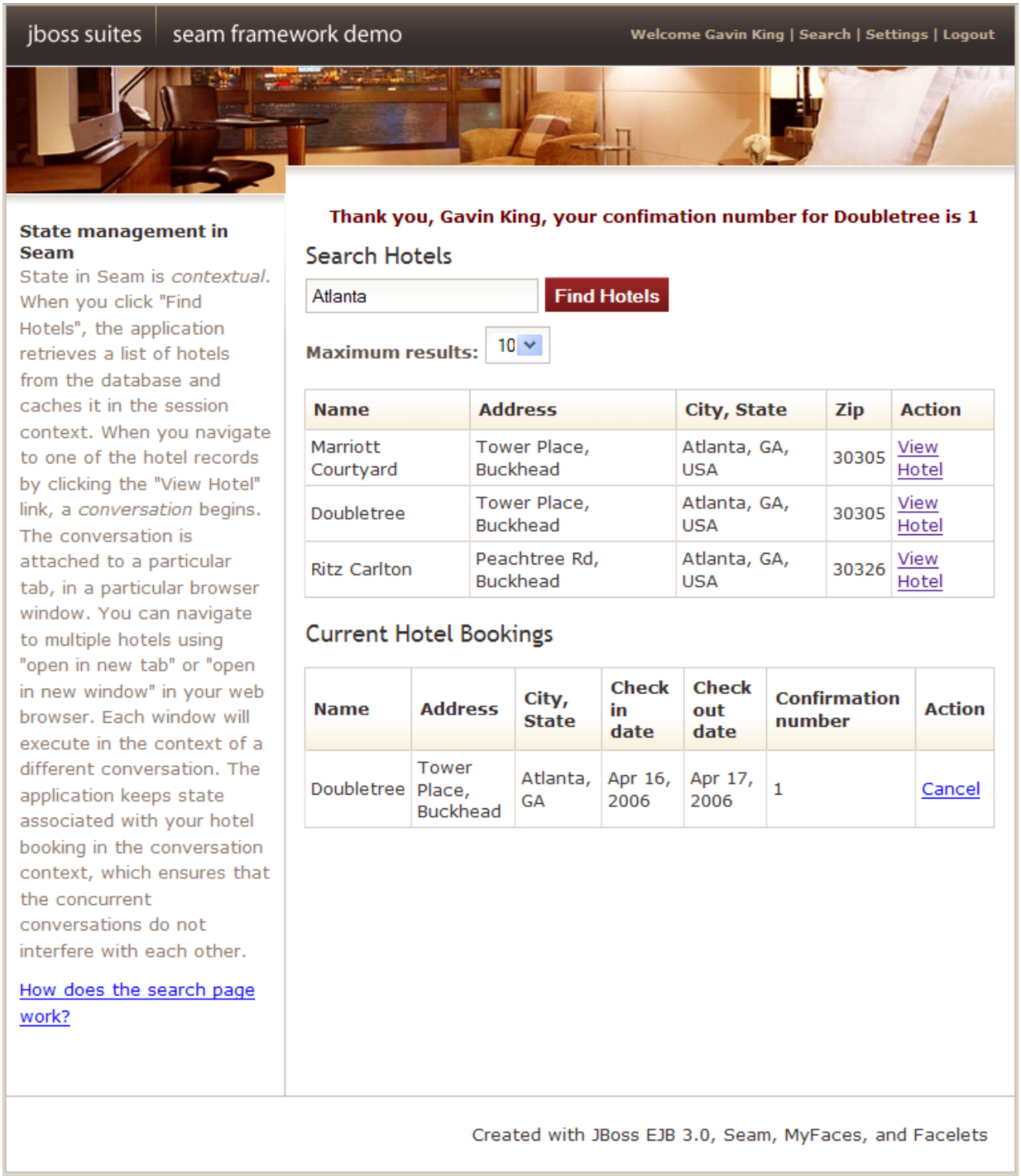
The numberguess example also contains Giveup and Cheat buttons. You should be able to trace the pageflow state for both actions relatively easily. Pay particular attention to the `cheat` transition, which loads a sub-process to handle that flow. Although it's overkill for this application, it does demonstrate how complex pageflows can be broken down into smaller parts to make them easier to understand.

1.6. ### Seam #########:

1.6.1.

##

- ######
- ####
- #####
- #######
- #####
- #####
- ####
- ####
- #######



############## JSF#EJB 3.0#Seam ########## Facelet ######### JSF#Facelets#Seam#JavaBeans ####Hibernate3 ##################

#################### ##############################
##
Seam

WEB ############################## ################################ Seam ######################

###################### ############################ ################## ##

1.6.2.

###################### ######################*1.1. #Seam ######### ########## ###################### ###### *http://localhost:8080/seam-booking/* [http://localhost:8080/seam-booking/]###############

################################## 6 ######## Bean #########

- `AuthenticatorAction` ##################
- `BookingListAction` ##############################
- `ChangePasswordAction` ############################
- `HotelBookingAction` ###################### ##### ## ########## ############################
- `HotelSearchingAction` ###############
- `RegisterAction` ####################

######### Bean ############################

- `Hotel` ############### Bean ###
- `Booking` ################ Bean ###
- `User` ######################## Bean ###

1.6.3. Seam

############################## ## ############ ################################### ######## ################### ##

WEB ## ################################Java WEB ############################ ##### URL ######## ################ `HttpSession`

########################## ################################## ##Java ## (2 ####) ########## ###################### ## LRU ################## #### ############################ ##

HttpSession ################### HttpSession ######################## ## #####################################
#######################HTTP ##################HTTP #################################
##
Seam

Seam ####################### *(conversation context)* ######### ############################ ######################### ######### ##
##

######################## Bean ###################### Java #################### Bean ########################### ########### Java ################################## ########################### Bean ################################# ####JBoss AS ############# ####### bean ############## ###### Bean #################### HttpSession ############# #################################### Bean ## Web ## Bean ##
#####################################Seam ####### Bean ##### POJO ############# Seam ##############

################## ################ ## #################### #################### ######Seam ################# ####################### ########## #################### ################# #########################

JavaScript ## RichFaces Ajax

####################### Bean ############ ###########################

1.25. HotelSearchingAction.java

```
@Stateful                                      (1)
@Name("hotelSearch")
@Scope(ScopeType.SESSION)

@Restrict("#{identity.loggedIn}")                  (2)
public class HotelSearchingAction implements HotelSearching
{

   @PersistenceContext
   private EntityManager em;
```

```
  private String searchString;
  private int pageSize = 10;
  private int page;


  @DataModel
  private List<Hotel
> hotels;

  public void find()
  {
    page = 0;
    queryHotels();
  }
  public void nextPage()
  {
    page++;
    queryHotels();
  }

  private void queryHotels()
  {
    hotels =
        em.createQuery("select h from Hotel h where lower(h.name) like #{pattern} " +
                    "or lower(h.city) like #{pattern} " +
                    "or lower(h.zip) like #{pattern} " +
                    "or lower(h.address) like #{pattern}")
          .setMaxResults(pageSize)
          .setFirstResult( page * pageSize )
          .getResultList();
  }

  public boolean isNextPageAvailable()
  {
    return hotels!=null && hotels.size()==pageSize;
  }

  public int getPageSize() {
    return pageSize;
  }

  public void setPageSize(int pageSize) {
    this.pageSize = pageSize;
  }
```

```
  @Factory(value="pattern", scope=ScopeType.EVENT)
  public String getSearchPattern()
  {
    return searchString==null ?
        "%" : '%' + searchString.toLowerCase().replace('*', '%') + '%';
  }

  public String getSearchString()
  {
    return searchString;
  }

  public void setSearchString(String searchString)
  {
    this.searchString = searchString;
  }

  @Remove
  public void destroy() {}
}
```

1. EJB ## `@Stateful` ######### ################## Bean ############## ########### Bean ## ##########################
2. `@Restrict` ################################## ####################################### ###########Seam ##########################
3. *`@DataModel`* ######## JSF `ListDataModel` ### `List` ####### ############################## ############ `hotels` ########### `ListDataModel` ##############
4. EJB ### `@Remove` ### Bean ################## Seam ############ Bean ######### `@Remove` ################## Seam ################################

################## Facelets ###### ######################

1.26. main.xhtml

```
<div class="section">

   <span class="errors">
     <h:messages globalOnly="true"/>
   </span>
```

```
  <h1>Search Hotels</h1>

  <h:form id="searchCriteria">
  <fieldset>
    <h:inputText id="searchString" value="#{hotelSearch.searchString}"
                 style="width: 165px;">
      <a:support event="onkeyup" actionListener="#{hotelSearch.find}"
                 reRender="searchResults" />
    </h:inputText>
     
    <a:commandButton id="findHotels" value="Find Hotels" action="#{hotelSearch.find}"
                     reRender="searchResults"/>
     

    <a:status>
      <f:facet name="start">
        <h:graphicImage value="/img/spinner.gif"/>
      </f:facet>
    </a:status>
    <br/>
    <h:outputLabel for="pageSize">Maximum results:</h:outputLabel> 
    <h:selectOneMenu value="#{hotelSearch.pageSize}" id="pageSize">
      <f:selectItem itemLabel="5" itemValue="5"/>
      <f:selectItem itemLabel="10" itemValue="10"/>
      <f:selectItem itemLabel="20" itemValue="20"/>
    </h:selectOneMenu>
  </fieldset>
  </h:form>

</div>


<a:outputPanel id="searchResults">
 <div class="section">
  <h:outputText value="No Hotels Found"
                rendered="#{hotels != null and hotels.rowCount==0}"/>
  <h:dataTable id="hotels" value="#{hotels}" var="hot"
               rendered="#{hotels.rowCount>0}">
    <h:column>
      <f:facet name="header">Name</f:facet>
      #{hot.name}
    </h:column>
    <h:column>
```

```
        <f:facet name="header">Address</f:facet>
        #{hot.address}
      </h:column>
      <h:column>
        <f:facet name="header">City, State</f:facet>
        #{hot.city}, #{hot.state}, #{hot.country}
      </h:column>
      <h:column>
        <f:facet name="header">Zip</f:facet>
        #{hot.zip}
      </h:column>
      <h:column>
        <f:facet name="header">Action</f:facet>

        <s:link id="viewHotel" value="View Hotel"
              action="#{hotelBooking.selectHotel(hot)}"/>
      </h:column>
    </h:dataTable>
    <s:link value="More results" action="#{hotelSearch.nextPage}"
        rendered="#{hotelSearch.nextPageAvailable}"/>
  </div>
</a:outputPanel>
```

(4)

1. RichFaces Ajax `<a:support>` ### `onkeyup` #### JavaScript ########### JSF ################### `XMLHttpRequest` ###################### ###########`reRender` ### JSF ##
2. RichFaces Ajax `<a:status>` ##
3. RichFaces Ajax `<a:outputPanel>` ###################################
4. Seam `<s:link>` #### JSF ############# (# JavaScript) HTML ############### ## `<h:commandLink>` ############# "###########" # "########"################## ##### `#{hotelBooking.selectHotel(hot)}` ################################## ########### EL# ######### Seam ## ############################## EL# #########

   ########################`WEB-INF/pages.xml` ############################## ######## *#6.7. #########* ########

############################ ########## `HotelBookingAction` # `selectHotel()` ############### ######################

Bean ############## ##

1.27. HotelBookingAction.java

```
@Stateful
@Name("hotelBooking")
@Restrict("#{identity.loggedIn}")
public class HotelBookingAction implements HotelBooking
{


  @PersistenceContext(type=EXTENDED)                                    (1)
  private EntityManager em;

  @In
  private User user;

  @In(required=false) @Out
  private Hotel hotel;

  @In(required=false)
  @Out(required=false)                                                  (2)
  private Booking booking;

  @In
  private FacesMessages facesMessages;

  @In
  private Events events;

  @Logger
  private Log log;

  private boolean bookingValid;


  @Begin                                                                (3)
  public void selectHotel(Hotel selectedHotel)
  {
    hotel = em.merge(selectedHotel);
  }

  public void bookHotel()
  {
    booking = new Booking(hotel, user);
```

```
    Calendar calendar = Calendar.getInstance();
    booking.setCheckinDate( calendar.getTime() );
    calendar.add(Calendar.DAY_OF_MONTH, 1);
    booking.setCheckoutDate( calendar.getTime() );
  }

  public void setBookingDetails()
  {
    Calendar calendar = Calendar.getInstance();
    calendar.add(Calendar.DAY_OF_MONTH, -1);
    if ( booking.getCheckinDate().before( calendar.getTime() ) )
    {
      facesMessages.addToControl("checkinDate", "Check in date must be a future date");
      bookingValid=false;
    }
    else if ( !booking.getCheckinDate().before( booking.getCheckoutDate() ) )
    {
      facesMessages.addToControl("checkoutDate",
                          "Check out date must be later than check in date");
      bookingValid=false;
    }
    else
    {
      bookingValid=true;
    }
  }

  public boolean isBookingValid()
  {
    return bookingValid;
  }


  @End                                                    4
  public void confirm()
  {
    em.persist(booking);
    facesMessages.add("Thank you, #{user.name}, your confimation number " +
                  " for #{hotel.name} is #{booki g.id}");
    log.info("New booking: #{booking.id} for #{user.username}");
    events.raiseTransactionSuccessEvent("bookingConfirmed");
  }

  @End
  public void cancel() {}
```

```
@Remove
public void destroy() {}
```

(5)

(1) ## Bean ##EJB3 ########## ####### ################### ########### Bean ####################

(2) *@Out* ## *(outject)* ############## ################################## `hotel` ############ `hotel` ##################

(3) *@Begin* ######## ################## *(long-running conversation)* ################ ################################ ####### ### `@End` ############################

(4) *@End* ## ###################################

(5) Seam ################## EJB remove ################ ########################

`HotelBookingAction` ## ############################## ####### `HttpSession` #### get/set ##

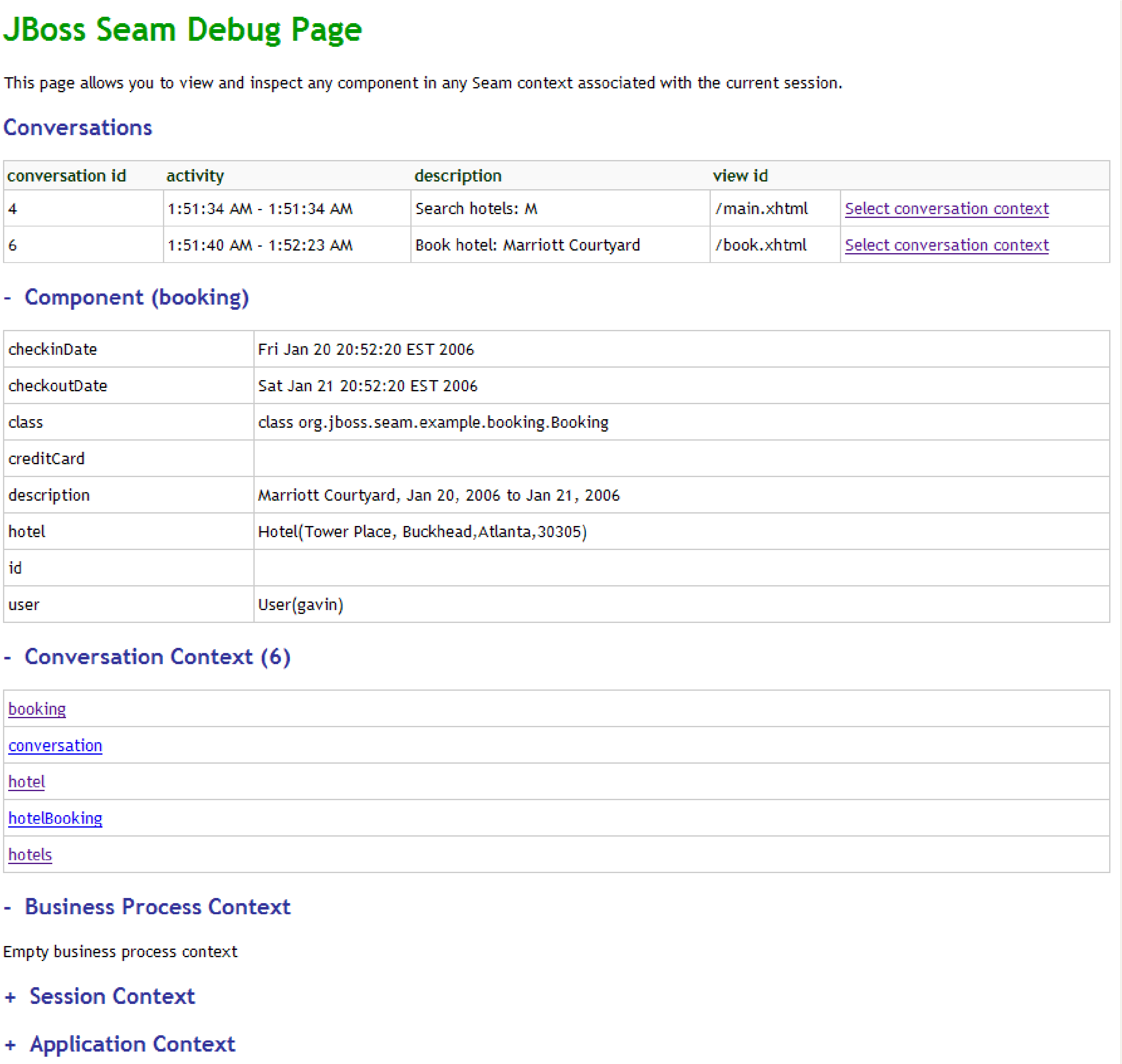
Seam ######################## ###################################### Seam

1.6.4. Seam

WAR # `seam-debug.jar` ###### Seam ######## `WEB-INF/lib` # Facelets ####### jar ########`init` ######## debug ############################

```
<core:init jndi-pattern="@jndiPattern@" debug="true"/>
```

############################## Seam ######## Seam ######################## ############ *http://localhost:8080/seam-booking/debug.seam* [http://localhost:8080/seam-booking/debug.seam] ##########



JBoss Seam Debug Page

This page allows you to view and inspect any component in any Seam context associated with the current session.

Conversations

conversation id	activity	description	view id	
4	1:51:34 AM - 1:51:34 AM	Search hotels: M	/main.xhtml	Select conversation context
6	1:51:40 AM - 1:52:23 AM	Book hotel: Marriott Courtyard	/book.xhtml	Select conversation context

- Component (booking)

checkinDate	Fri Jan 20 20:52:20 EST 2006
checkoutDate	Sat Jan 21 20:52:20 EST 2006
class	class org.jboss.seam.example.booking.Booking
creditCard	
description	Marriott Courtyard, Jan 20, 2006 to Jan 21, 2006
hotel	Hotel(Tower Place, Buckhead,Atlanta,30305)
id	
user	User(gavin)

- Conversation Context (6)

booking
conversation
hotel
hotelBooking
hotels

- Business Process Context

Empty business process context

+ Session Context

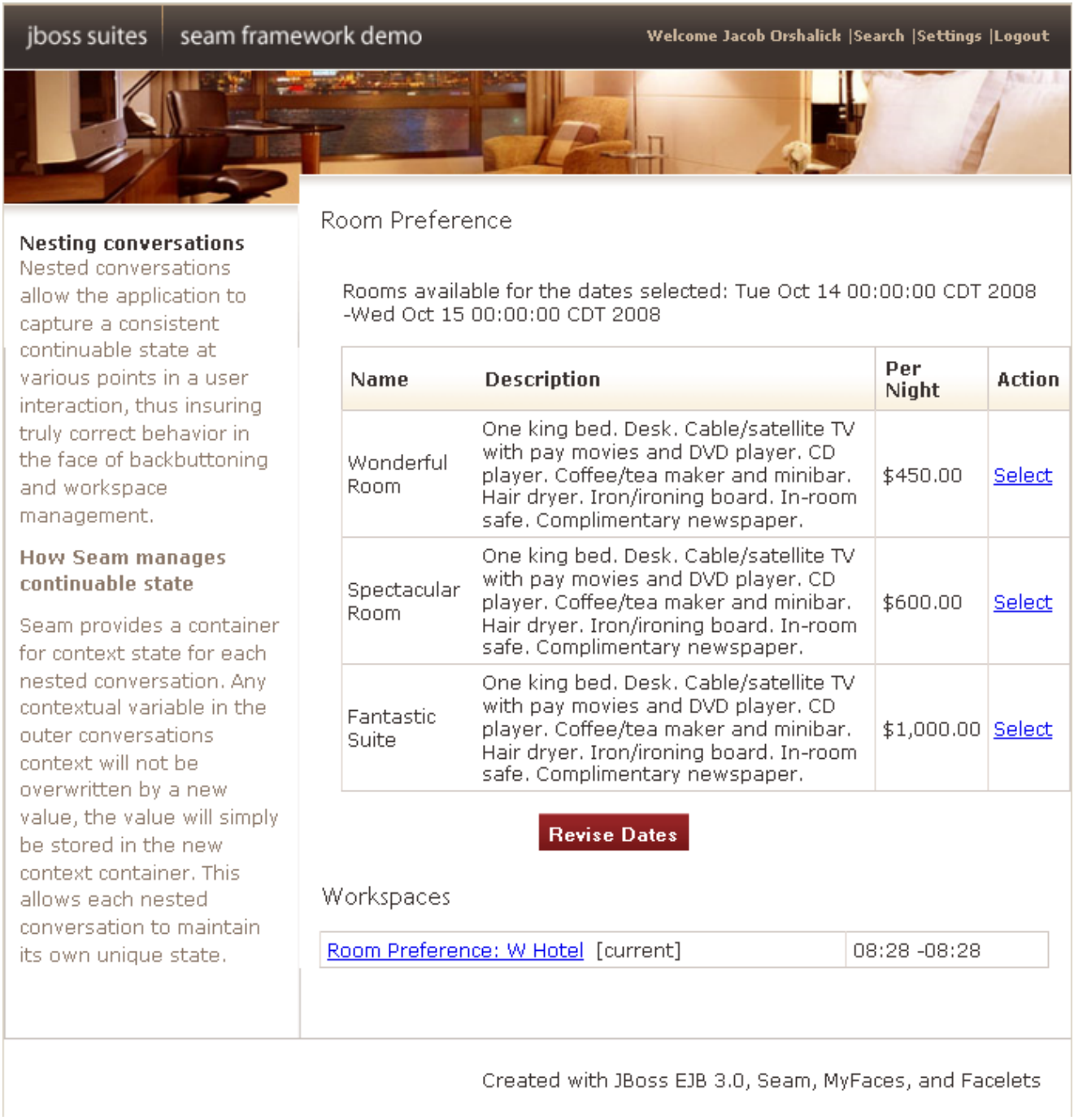
+ Application Context

1.7. ######## :

1.7.1.

##
###############################
###

##
##
#####################################



##
HTTPSession
###

Wonderful Room ############### ## *Fantastic Suite* ############### ################################ *Wonderful Room*

##
##

1.7.2.

###
####################################

1.28. RoomPreferenceAction.java

```
@Stateful
@Name("roomPreference")
@Restrict("#{identity.loggedIn}")
public class RoomPreferenceAction implements RoomPreference
{

  @Logger
  private Log log;

  @In private Hotel hotel;

  @In private Booking booking;

  @DataModel(value="availableRooms")
  private List<Room> availableRooms;

  @DataModelSelection(value="availableRooms")
  private Room roomSelection;

  @In(required=false, value="roomSelection")
  @Out(required=false, value="roomSelection")
  private Room room;

  @Factory("availableRooms")

  public void loadAvailableRooms()                                    (1)
  {
                        availableRooms    =    hotel.getAvailableRooms(booking.getCheckinDate(),
booking.getCheckoutDate());
    log.info("Retrieved #0 available rooms", availableRooms.size());
  }

  public BigDecimal getExpectedPrice()
  {
    log.info("Retrieving price for room #0", roomSelection.getName());
```

```
    return booking.getTotal(roomSelection);
  }

  @Begin(nested=true)                                          2
  public String selectPreference()
  {
    log.info("Room selected");

    this.room = this.roomSelection;                            3

    return "payment";
  }

  public String requestConfirmation()
  {
    // all validations are performed through the s:validateAll, so checks are already
    // performed
    log.info("Request confirmation from user");

    return "confirm";
  }

  @End(beforeRedirect=true)
                                                               4
  public String cancel()
  {
    log.info("ending conversation");

    return "cancel";
  }

  @Destroy @Remove
  public void destroy() {}
}
```

1. The `hotel` instance is injected from the conversation context. The hotel is loaded through an *extended persistence context* so that the entity remains managed throughout the conversation. This allows us to lazily load the `availableRooms` through an `@Factory` method by simply walking the association.
2. *`@Begin(nested=true)`* ################################ ##

```
③ roomSelection       #       @DataModelSelection       ####################
   ##########################roomSelection   ##################################
   ################################################
④ @End ################################## ############ roomSelection ########
```

When we begin a nested conversation it is pushed onto the conversation stack. In the `nestedbooking` example, the conversation stack consists of the outer long-running conversation (the booking) and each of the nested conversations (room selections).

1.29. rooms.xhtml

```
<div class="section">
   <h1>Room Preference</h1>
</div>

<div class="section">
   <h:form id="room_selections_form">
      <div class="section">
         <h:outputText styleClass="output"
            value="No rooms available for the dates selected: "
            rendered="#{availableRooms != null and availableRooms.rowCount == 0}"/>
         <h:outputText styleClass="output"
            value="Rooms available for the dates selected: "
            rendered="#{availableRooms != null and availableRooms.rowCount > 0}"/>

         <h:outputText styleClass="output" value="#{booking.checkinDate}"/> -
         <h:outputText styleClass="output" value="#{booking.checkoutDate}"/>

         <br/><br/>
                                                    ①
         <h:dataTable value="#{availableRooms}" var="room"
               rendered="#{availableRooms.rowCount > 0}">
            <h:column>
               <f:facet name="header">Name</f:facet>
               #{room.name}
            </h:column>
            <h:column>
               <f:facet name="header">Description</f:facet>
               #{room.description}
            </h:column>
            <h:column>
               <f:facet name="header">Per Night</f:facet>
               <h:outputText value="#{room.price}">
```

```
                <f:convertNumber type="currency" currencySymbol="$"/>
              </h:outputText>
            </h:column>

            <h:column>                                                    2
              <f:facet name="header">Action</f:facet>
              <h:commandLink id="selectRoomPreference"
                action="#{roomPreference.selectPreference}">Select</h:commandLink>
            </h:column>
          </h:dataTable>
        </div>
        <div class="entry">
          <div class="label"> </div>

          <div class="input">                                             3
            <s:button id="cancel" value="Revise Dates" view="/book.xhtml"/>
          </div>
        </div>
    </h:form>
</div>
```

1. EL ##########RoomPreferenceAction ###### @Factory ####### #{availableRooms} ######### @Factory ##### *@DataModel* ############################## 1 ##########
2. #{roomPreference.selectPreference} ###################### @DataModelSelection ########## ################################
3. ######### /book.xhtml ####### ################ # room preference ######## ################################ <s:button > ######## /book.xhtml ############################

HotelBookingAction.#########################

1.30. HotelBookingAction.java

```
@Stateful
@Name("hotelBooking")
@Restrict("#{identity.loggedIn}")
public class HotelBookingAction implements HotelBooking
{

   @PersistenceContext(type=EXTENDED)
   private EntityManager em;
```

```
  @In
  private User user;

  @In(required=false) @Out
  private Hotel hotel;

  @In(required=false)
  @Out(required=false)
  private Booking booking;

  @In(required=false)
  private Room roomSelection;

  @In
  private FacesMessages facesMessages;

  @In
  private Events events;

  @Logger
  private Log log;

  @Begin
  public void selectHotel(Hotel selectedHotel)
  {
    log.info("Selected hotel #0", selectedHotel.getName());
    hotel = em.merge(selectedHotel);
  }

  public String setBookingDates()
  {
    // the result will indicate whether or not to begin the nested conversation
    // as well as the navigation.  if a null result is returned, the nested
    // conversation will not begin, and the user will be returned to the current
    // page to fix validation issues
    String result = null;

    Calendar calendar = Calendar.getInstance();
    calendar.add(Calendar.DAY_OF_MONTH, -1);

    // validate what we have received from the user so far
    if ( booking.getCheckinDate().before( calendar.getTime() ) )
    {
      facesMessages.addToControl("checkinDate", "Check in date must be a future date");
```

```
    }
    else if ( !booking.getCheckinDate().before( booking.getCheckoutDate() ) )
    {
        facesMessages.addToControl("checkoutDate", "Check out date must be later than check
in date");
    }
    else
    {
       result = "rooms";
    }

    return result;
  }

  public void bookHotel()
  {
    booking = new Booking(hotel, user);
    Calendar calendar = Calendar.getInstance();
    booking.setCheckinDate( calendar.getTime() );
    calendar.add(Calendar.DAY_OF_MONTH, 1);
    booking.setCheckoutDate( calendar.getTime() );
  }

  @End(root=true)
                                                                    (1)
  public void confirm()
  {
    // on confirmation we set the room preference in the booking.  the room preference
    // will be injected based on the nested conversation we are in.
    booking.setRoomPreference(roomSelection);
                                                                    (2)
    em.persist(booking);
     facesMessages.add("Thank you, #{user.name}, your confimation number for #{hotel.name}
is #{booking.id}");
    log.info("New booking: #{booking.id} for #{user.username}");
    events.raiseTransactionSuccessEvent("bookingConfirmed");
  }


  @End(root=true, beforeRedirect=true)                              (3)
  public void cancel() {}

  @Destroy @Remove
  public void destroy() {}
```

```
}
```

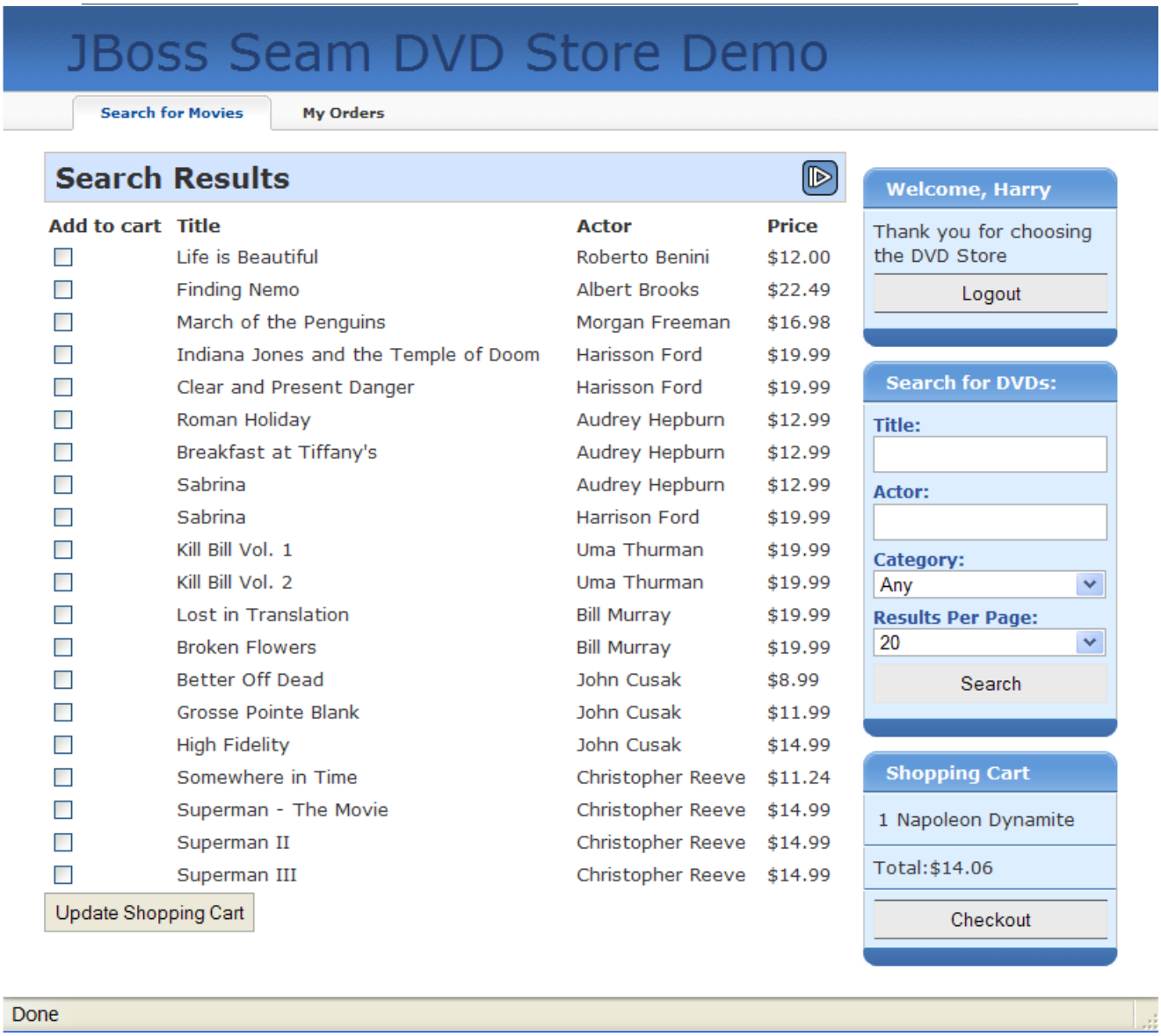
1. Annotating an action with `@End(root=true)` ends the root conversation which effectively destroys the entire conversation stack. When any conversation is ended, its nested conversations are ended as well. As the root is the conversation that started it all, this is a simple way to destroy and release all state associated with a workspace once the booking is confirmed.
2. `roomSelection` ########## `booking` #########
   ##
   ##
3. `@End(root=true,` `beforeRedirect=true)`
   ##

##
##

1.8. Seam # jBPM ################: DVD

DVD ################ ################ jBPM ##################

############################## jPDL ###############

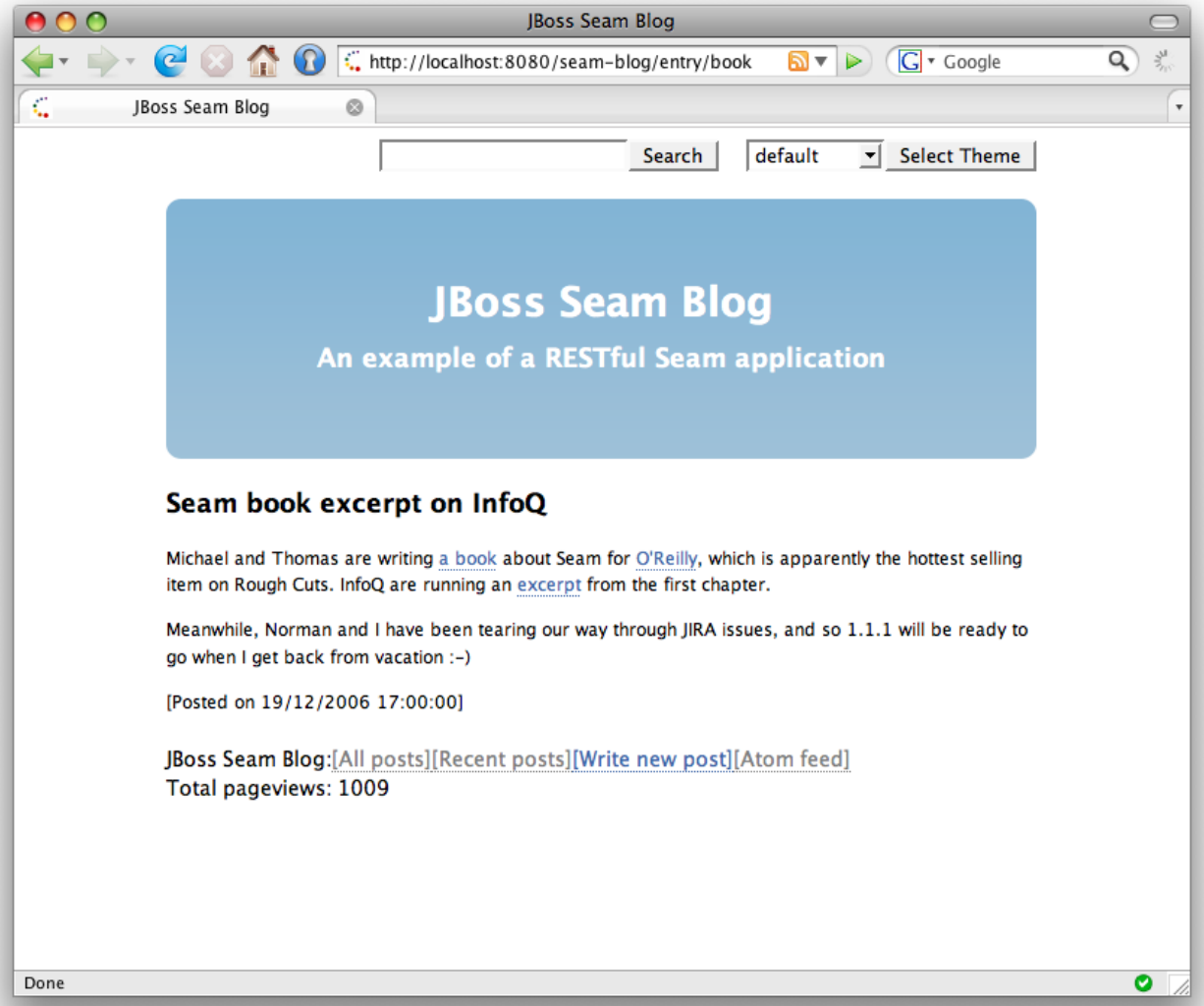

JBoss Seam DVD Store Demo
Search for Movies
My Orders
Search Results
Add to cart
Title
Actor
Price
Life is Beautiful
Roberto Benini
$12.00
Finding Nemo
Albert Brooks
$22.49
March of the Penguins
Morgan Freeman
$16.98
Indiana Jones and the Temple of Doom
Harisson Ford
$19.99
Clear and Present Danger
Harisson Ford
$19.99
Roman Holiday
Audrey Hepburn
$12.99
Breakfast at Tiffany's
Audrey Hepburn
$12.99
Sabrina
Audrey Hepburn
$12.99
Sabrina
Harrison Ford
$19.99
Kill Bill Vol. 1
Uma Thurman
$19.99
Kill Bill Vol. 2
Uma Thurman
$19.99
Lost in Translation
Bill Murray
$19.99
Broken Flowers
Bill Murray
$19.99
Better Off Dead
John Cusak
$8.99
Grosse Pointe Blank
John Cusak
$11.99
High Fidelity
John Cusak
$14.99
Somewhere in Time
Christopher Reeve
$11.24
Superman - The Movie
Christopher Reeve
$14.99
Superman II
Christopher Reeve
$14.99
Superman III
Christopher Reeve
$14.99
Update Shopping Cart
Welcome, Harry
Thank you for choosing the DVD Store
Logout
Search for DVDs:
Title:
Actor:
Category:
Any
Results Per Page:
20
Search
Shopping Cart
1 Napoleon Dynamite
Total:$14.06
Checkout
Done

################################## jBPM #######
##

Seam DVD ###################### `dvdstore` ###############

1.9. Blog ############## URL

Seam ###################################### ################################## (######## *(content)* ##########) ############################## ################################## ## ## Blog ##### Seam ##### RESTful ###################### ##



Blog ######"#### (PULL) " - #### MVC ############ ## ################################### (PULL)

1.9.1. "PULL" # MVC

`index.xhtml` facelets ##############################

 # 1.31.

```
<h:dataTable value="#{blog.recentBlogEntries}" var="blogEntry" rows="3">
 <h:column>
   <div class="blogEntry">
     <h3
>#{blogEntry.title}</h3>
```

```
      <div>
       <s:formattedText value="#{blogEntry.excerpt==null ? blogEntry.body : blogEntry.excerpt}"/>
      </div>
      <p>
        <s:link view="/entry.xhtml" rendered="#{blogEntry.excerpt!=null}" propagation="none"
            value="Read more...">
          <f:param name="blogEntryId" value="#{blogEntry.id}"/>
        </s:link>
      </p>
      <p>
        [Posted on 
        <h:outputText value="#{blogEntry.date}">
           <f:convertDateTime timeZone="#{blog.timeZone}" locale="#{blog.locale}" type="both"/>
        </h:outputText
>]
         
        <s:link view="/entry.xhtml" propagation="none" value="[Link]">
          <f:param name="blogEntryId" value="#{blogEntry.id}"/>
        </s:link>
      </p>
    </div>
 </h:column>
</h:dataTable
>
```

If we navigate to this page from a bookmark, how does the `#{blog.recentBlogEntries}` data used by the `<h:dataTable>` actually get initialized? The `Blog` is retrieved lazily — "pulled" — when needed, by a Seam component named `blog`. This is the opposite flow of control to what is used in traditional action-based web frameworks like Struts.

1.32.

```
@Name("blog")
@Scope(ScopeType.STATELESS)
@AutoCreate
public class BlogService
{

  @In EntityManager entityManager;                    (1)

  @Unwrap                                             (2)
  public Blog getBlog()
```

```
  {
        return (Blog) entityManager.createQuery("select distinct b from Blog b left join fetch
 b.blogEntries")
         .setHint("org.hibernate.cacheable", true)
         .getSingleResult();
  }

}
```

1. ########## *Seam* ########## *(seam-managed persistence context)* ######### ##################################EJB3 ######### Seam ########## ######### Web ##
2. The `@Unwrap` annotation tells Seam to provide the return value of the method — the `Blog` — instead of the actual `BlogService` component to clients. This is the Seam *manager component pattern*.

################## ##

1.9.2.

Blog ###### ############ Blog ######################### ####facelet #######`template.xhtml` ##### `menu.xhtml`

1.33.

```
<div id="search">
  <h:form>
    <h:inputText value="#{searchAction.searchPattern}"/>
    <h:commandButton value="Search" action="/search.xhtml"/>
  </h:form>
</div
>
```

######################## ################### ######################## ####### (outcome) ### JSF ### ID ########## Seam ###################### ### ID ########## ######################################

```
<navigation-rule>
  <navigation-case>
    <from-outcome
>searchResults</from-outcome>
    <to-view-id
>/search.xhtml</to-view-id>
```

```
    <redirect/>
  </navigation-case>
</navigation-rule
>
```

##########################

```
<div id="search">
  <h:form>
    <h:inputText value="#{searchAction.searchPattern}"/>
    <h:commandButton value="Search" action="searchResults"/>
  </h:form>
</div
>
```

But when we redirect, we need to include the values submitted with the form in the URL to get a bookmarkable URL like `http://localhost:8080/seam-blog/search/`. JSF does not provide an easy way to do this, but Seam does. We use two Seam features to accomplish this: *page parameters* and *URL rewriting*. Both are defined in `WEB-INF/pages.xml`:

1.34.

```
<pages>
  <page view-id="/search.xhtml">
    <rewrite pattern="/search/{searchPattern}"/>
    <rewrite pattern="/search"/>

    <param name="searchPattern" value="#{searchService.searchPattern}"/>

  </page>
  ...
</pages
>
```

Seam # searchPattern ############## #{searchService.searchPattern} ################# Seam # URL ###################################

Without URL rewriting, the URL for a search on the term `book` would be `http://localhost:8080/seam-blog/seam/search.xhtml?searchPattern=book`. This is nice, but Seam can make the URL even simpler using a rewrite rule. The first rewrite rule, for the pattern `/search/{searchPattern}`, says that any time we have a URL for search.xhtml with a searchPattern

request parameter, we can fold that URL into the simpler URL. So,the URL we saw earlier, `http://localhost:8080/seam-blog/seam/search.xhtml?searchPattern=book` can be written instead as `http://localhost:8080/seam-blog/search/book`.

##########URL ########## Seam ###### URL
URL
################### `components.xml` ###############

```
<web:rewrite-filter view-mapping="/seam/*" />
```

########## `search.xhtml` ##########

```
<h:dataTable value="#{searchResults}" var="blogEntry">
  <h:column>
    <div>
      <s:link view="/entry.xhtml" propagation="none" value="#{blogEntry.title}">
        <f:param name="blogEntryId" value="#{blogEntry.id}"/>
      </s:link>
      posted on
      <h:outputText value="#{blogEntry.date}">
         <f:convertDateTime timeZone="#{blog.timeZone}" locale="#{blog.locale}" type="both"/>
      </h:outputText>
    </div>
  </h:column>
</h:dataTable
>
```

Hibernate ################### "PULL" # MVC

```
@Name("searchService")
public class SearchService
{

   @In
   private FullTextEntityManager entityManager;

   private String searchPattern;

   @Factory("searchResults")
   public List<BlogEntry
> getSearchResults()
```

```
  {
    if (searchPattern==null || "".equals(searchPattern) ) {
      searchPattern = null;
            return  entityManager.createQuery("select be from BlogEntry be order by date
 desc").getResultList();
    }
    else
    {
      Map<String,Float
> boostPerField = new HashMap<String,Float
>();
      boostPerField.put( "title", 4f );
      boostPerField.put( "body", 1f );
      String[] productFields = {"title", "body"};
      QueryParser parser = new MultiFieldQueryParser(productFields, new StandardAnalyzer(),
 boostPerField);
      parser.setAllowLeadingWildcard(true);
      org.apache.lucene.search.Query luceneQuery;
      try
      {
        luceneQuery = parser.parse(searchPattern);
      }
      catch (ParseException e)
      {
        return null;
      }

      return entityManager.createFullTextQuery(luceneQuery, BlogEntry.class)
          .setMaxResults(100)
          .getResultList();
    }
  }

  public String getSearchPattern()
  {
    return searchPattern;
  }

  public void setSearchPattern(String searchPattern)
  {
    this.searchPattern = searchPattern;
  }
```

```
}
```

1.9.3. RESTful ######### "PUSH" # MVC

#####RESTful ######### PUSH # MVC ################## ####Seam ########## ########## Blog ###### Blog ###### `entry.xhtml` ################## ######################PULL # MVC #################

`entryAction` ######### Struts ######## PUSH # MVC ########################

```
@Name("entryAction")
@Scope(STATELESS)
public class EntryAction
{
  @In Blog blog;

  @Out BlogEntry blogEntry;

  public void loadBlogEntry(String id) throws EntryNotFoundException
  {
    blogEntry = blog.getBlogEntry(id);
    if (blogEntry==null) throw new EntryNotFoundException(id);
  }

}
```

##########`pages.xml` #########

```
<pages>
  ...

   <page view-id="/entry.xhtml"
>
      <rewrite pattern="/entry/{blogEntryId}" />
      <rewrite pattern="/entry" />

      <param name="blogEntryId"
           value="#{blogEntry.id}"/>

      <action execute="#{entryAction.loadBlogEntry(blogEntry.id)}"/>
   </page>
```

```
    <page view-id="/post.xhtml" login-required="true">
        <rewrite pattern="/post" />

        <action execute="#{postAction.post}"
                if="#{validation.succeeded}"/>

        <action execute="#{postAction.invalid}"
                if="#{validation.failed}"/>

        <navigation from-action="#{postAction.post}">
            <redirect view-id="/index.xhtml"/>
        </navigation>
    </page>

    <page view-id="*">
        <action execute="#{blog.hitCount.hit}"/>
    </page>

</pages
>
```

##
JSF EL ############Seam ############### JSF

When the `entry.xhtml` page is requested, Seam first binds the page parameter `blogEntryId` to the model. Keep in mind that because of the URL rewriting, the blogEntryId parameter name won't show up in the URL. Seam then runs the page action, which retrieves the needed data — the `blogEntry` — and places it in the Seam event context. Finally, the following is rendered:

```
<div class="blogEntry">
    <h3
>#{blogEntry.title}</h3>
    <div>
        <s:formattedText value="#{blogEntry.body}"/>
    </div>
    <p>
    [Posted on 
    <h:outputText value="#{blogEntry.date}">
        <f:convertDateTime timeZone="#{blog.timeZone}" locale="#{blog.locale}" type="both"/>
    </h:outputText
>]
    </p>
```

```
</div
>
```

blog ###################### `EntryNotFoundException` ########### exception is thrown. ##### 505 ####### 404 ########## ####################

```
@ApplicationException(rollback=true)
@HttpError(errorCode=HttpServletResponse.SC_NOT_FOUND)
public class EntryNotFoundException extends Exception
{
  EntryNotFoundException(String id)
  {
    super("entry not found: " + id);
  }
}
```

####################################

```
@Name("entryAction")
@Scope(STATELESS)
public class EntryAction
{
  @In(create=true)
  private Blog blog;

  @In @Out
  private BlogEntry blogEntry;

  public void loadBlogEntry() throws EntryNotFoundException
  {
    blogEntry = blog.getBlogEntry( blogEntry.getId() );
    if (blogEntry==null) throw new EntryNotFoundException(id);
  }
}
```

```
<pages>
  ...

  <page view-id="/entry.xhtml" action="#{entryAction.loadBlogEntry}">
    <param name="blogEntryId" value="#{blogEntry.id}"/>
```

```
  </page>

  ...
</pages
>
```

####################

atom

seam-gen #### Seam

Seam ################################## Eclipse ############## Seam ############# ##

####Seam ############ ### ############################## Ruby ################################ ############

#########seam-gen # JBoss AS ############ ### J2EE # Java 5 ####################

Eclipse ##### seam-gen ################### Eclipse ################################ Eclipse ############### ################### — ##########################

seam-gen is basically just an intricate Ant script wrapped around Hibernate Tools, together with some templates. That makes it easy to customize if you need to.

2.1.

Make sure you have JDK 5 or JDK 6 (see *#42.1. #JDK #####* for details), JBoss AS 4.2 or 5.0 and Ant 1.7.0, along with recent versions of Eclipse, the JBoss IDE plugin for Eclipse and the TestNG plugin for Eclipse correctly installed before starting. Add your JBoss installation to the JBoss Server View in Eclipse. Start JBoss in debug mode. Finally, start a command prompt in the directory where you unzipped the Seam distribution.

JBoss # WAR # EAR ######################## ###### JVM ######### — ########## — EAR ################### JVM # perm gen ###################### ################# perm gen space ######## JVM # JBoss ############### JBoss IDE ## JBoss ########## #VM ################################ ##############

```
-Xms512m -Xmx1024m -XX:PermSize=256m -XX:MaxPermSize=512m
```

##########################

```
-Xms256m -Xmx512m -XX:PermSize=128m -XX:MaxPermSize=256m
```

######### JBoss ########### `bin/run.conf` # JVM #############

############################ — `OutOfMemoryException` ################

2.2. Setting up a new project

The first thing we need to do is configure seam-gen for your environment: JBoss AS installation directory, project workspace, and database connection. It's easy, just type:

```
cd jboss-seam-2.2.x
seam setup
```

############################

```
~/workspace/jboss-seam$ ./seam setup
Buildfile: build.xml

init:

setup:
     [echo] Welcome to seam-gen :-)
     [input] Enter your project workspace (the directory that contains your Seam projects) [C:/
Projects] [C:/Projects]
/Users/pmuir/workspace
    [input] Enter your JBoss home directory [C:/Program Files/jboss-4.2.3.GA] [C:/Program Files/
jboss-4.2.3.GA]
/Applications/jboss-4.2.3.GA
    [input] Enter the project name [myproject] [myproject]
helloworld
     [echo] Accepted project name as: helloworld
    [input] Select a RichFaces skin (not applicable if using ICEFaces) [blueSky] ([blueSky], classic,
 ruby, wine, deepMarine, emeraldTown, sakura, DEFAULT)

     [input] Is this project deployed as an EAR (with EJB components) or a WAR (with no EJB
 support) [ear]  ([ear], war, )

      [input] Enter the Java package name for your session beans [com.mydomain.helloworld]
 [com.mydomain.helloworld]
org.jboss.helloworld
         [input] Enter the Java package name for your entity beans [org.jboss.helloworld]
 [org.jboss.helloworld]

        [input] Enter the Java package name for your test cases [org.jboss.helloworld.test]
 [org.jboss.helloworld.test]
```

```
    [input] What kind of database are you using? [hsql]  ([hsql], mysql, oracle, postgres, mssql,
 db2, sybase, enterprisedb, h2)
mysql
     [input] Enter the Hibernate dialect for your database [org.hibernate.dialect.MySQLDialect]
 [org.hibernate.dialect.MySQLDialect]

   [input] Enter the filesystem path to the JDBC driver jar [lib/hsqldb.jar] [lib/hsqldb.jar]
/Users/pmuir/java/mysql.jar
          [input] Enter JDBC driver class for your database [com.mysql.jdbc.Driver]
 [com.mysql.jdbc.Driver]

   [input] Enter the JDBC URL for your database [jdbc:mysql:///test] [jdbc:mysql:///test]
jdbc:mysql:///helloworld
   [input] Enter database username [sa] [sa]
pmuir
   [input] Enter database password [] []

   [input] skipping input as property hibernate.default_schema.new has already been set.
   [input] Enter the database catalog name (it is OK to leave this blank) [] []

   [input] Are you working with tables that already exist in the database? [n]  (y, [n], )
y
   [input] Do you want to drop and recreate the database tables and data in import.sql each time
 you deploy? [n]  (y, [n], )
n
   [input] Enter your ICEfaces home directory (leave blank to omit ICEfaces) [] []

[propertyfile] Creating new property file: /Users/pmuir/workspace/jboss-seam/seam-gen/
build.properties
     [echo] Installing JDBC driver jar to JBoss server
     [echo] Type 'seam create-project' to create the new project

BUILD SUCCESSFUL
Total time: 1 minute 32 seconds
~/workspace/jboss-seam $
```

########################## ########### enter #############

########################## EAR ######## WAR ################## EAR ######## EJB 3.0 #### Java EE 5 ###### WAR ######## EJB 3.0 ######### J2EE ################ WAR # EAR ######################## JBoss #### EJB3 ################################ `ear` ########## ##### `war` ########## ### ########### EAR ###################### WAR ################################

#################### ######################## seam-gen #############

`seam-gen/build.properties` ########## #### `seam setup`

###############Eclipse ############### #################

```
seam new-project
```

```
C:\Projects\jboss-seam>seam new-project
Buildfile: build.xml

...

new-project:
     [echo] A new Seam project named 'helloworld' was created in the C:\Projects directory
     [echo] Type 'seam explode' and go to http://localhost:8080/helloworld
    [echo] Eclipse Users: Add the project into Eclipse using File > New > Project and select General
 > Project (not Java Project)
     [echo] NetBeans Users: Open the project in NetBeans

BUILD SUCCESSFUL
Total time: 7 seconds
C:\Projects\jboss-seam>
```

Seam jar##### jar ### JDBC #### jar #### Eclipse ############ Eclipse ######## Ant ################################## facelets ########## ############## `## -> ######... -> ## -> ###### -> ##` ############### ####### (##### `helloworld`) ###### ## ######### Eclipse ################ JBoss AS ################## ############### `Java` ###### ############

Eclipse ###### JDK # Java SE 5 #### Java SE 6 # JDK ###### `###### -> ##### -> Java` ##### #####Java SE 5 ### JDK ###########

########Eclise ##### `seam explode` ############################

Go to `http://localhost:8080/helloworld` to see a welcome page. This is a facelets page, `view/home.xhtml`, using the template `view/layout/template.xhtml`. You can edit this page, or the template, in Eclipse, and see the results *immediately*, by clicking refresh in your browser.

############# XML ############################## ########### Java EE ######## ####### ############### # Seam ####### 90% ######## (seam-gen ##############################)

################################## HSQLDB #### TestNG ############### `persistence-test.xml` # `import-test.sql` ############ `import-test.sql` ## `myproject-dev-`

ds.xml#persistence-dev.xml # import-dev.sql ##################################
seam-gen ##
myproject-prod-ds.xml# persistence-prod.xml # import-prod.sql ######
################################### ############################

2.3.

############# Web ################ ######### Java ###################### Web
######################### ################

```
seam new-action
```

Seam ################################## facelets page # Seam ###############

```
C:\Projects\jboss-seam>seam new-action
Buildfile: build.xml

validate-workspace:

validate-project:

action-input:
   [input] Enter the Seam component name
ping
   [input] Enter the local interface name [Ping]

   [input] Enter the bean class name [PingBean]

   [input] Enter the action method name [ping]

   [input] Enter the page name [ping]


setup-filters:

new-action:
    [echo] Creating a new stateless session bean component with an action method
    [copy] Copying 1 file to C:\Projects\helloworld\src\hot\org\jboss\helloworld
    [copy] Copying 1 file to C:\Projects\helloworld\src\hot\org\jboss\helloworld
    [copy] Copying 1 file to C:\Projects\helloworld\src\hot\org\jboss\helloworld\test
    [copy] Copying 1 file to C:\Projects\helloworld\src\hot\org\jboss\helloworld\test
    [copy] Copying 1 file to C:\Projects\helloworld\view
    [echo] Type 'seam restart' and go to http://localhost:8080/helloworld/ping.seam
```

```
BUILD SUCCESSFUL
Total time: 13 seconds
C:\Projects\jboss-seam>
```

Seam ############### ############################## seam restart ####### Eclipse ############### build.xml #### # restart ######################## ############## Eclipse # resources/META-INF/application.xml ############## ################ *JBoss*

###http://localhost:8080/helloworld/ping.seam ###################### ####### src directory ###################################### ping() ################## ################

####PingTest.xml ##### test ########## Eclipse # TestNG ###################### ######## seam test ############### ########## test ############

2.4.

############################

```
seam new-form
```

```
C:\Projects\jboss-seam>seam new-form
Buildfile: C:\Projects\jboss-seam\seam-gen\build.xml

validate-workspace:

validate-project:

action-input:
   [input] Enter the Seam component name
hello
   [input] Enter the local interface name [Hello]

   [input] Enter the bean class name [HelloBean]

   [input] Enter the action method name [hello]

   [input] Enter the page name [hello]


setup-filters:
```

```
new-form:
    [echo] Creating a new stateful session bean component with an action method
    [copy] Copying 1 file to C:\Projects\hello\src\hot\com\hello
    [copy] Copying 1 file to C:\Projects\hello\src\hot\com\hello
    [copy] Copying 1 file to C:\Projects\hello\src\hot\com\hello\test
    [copy] Copying 1 file to C:\Projects\hello\view
    [copy] Copying 1 file to C:\Projects\hello\src\hot\com\hello\test
    [echo] Type 'seam restart' and go to http://localhost:8080/hello/hello.seam

BUILD SUCCESSFUL
Total time: 5 seconds
C:\Projects\jboss-seam>
```

################ http://localhost:8080/helloworld/hello.seam ###### ################ ########## ################ Seam ################## (Java ######################################)

2.5.

####################### (######################### seam setup #######) ##############

```
seam generate-entities
```

########## http://localhost:8080/helloworld ######### ########## ############ ################## ##################################### Seam ### ## seam-gen ######################

2.6. ### JPA/EJB3

################ src/main ##################

```
seam generate-ui
```

################http://localhost:8080/helloworld #########

2.7. EAR

Java EE 5 ### ### seam unexplode ############################ EAR ########## ########## seam deploy

################ build ###### deploy ########### seam undeploy ### undeploy ####################

################ *dev profile* ########## EAR # persistence-dev.xml ##### import-dev.sql ######## myproject-dev-ds.xml ############## ############# ########## *prod profile* #######

```
seam -Dprofile=prod deploy
```

—— ####persistence-staging.xml#import-staging.sql # myproject-staging-ds.xml ##### -Dprofile=staging

2.8. Seam

############ Seam ################## ########################## components.xml ############## Seam # Facelets ########################

```
<core:init debug="true"
>
```

######### Web#####################################

- facelets ###
- pages.xml ####

Java ################ ######################## (JBoss ################################## EAR ######### application.xml# WAR ######### web.xml ###)

########/#####/############### Seam # JavaBean ############################## ############### JavaBean ######## WEB-INF/dev ################### ############# WAR #### EAR ########### ### Seam

###################

- ######## JavaBean ################ EJB3 Bean ###### (###########)
- ##########################
- components.xml ##################################
- ################## WEB-INF/dev ######################

- Seam ########## `jboss-seam-debug.jar` # `WEB-INF/lib` ##########
- web.xml # Seam ######################
- ##

seam-gen ##### WAR ############## ##############`src/hot` ############################ ####### seam-gen # EAR ############################

2.9. JBoss 4.0 # Seam

Seam 2 was developed for JavaServer Faces 1.2. When using JBoss AS, we recommend using JBoss 4.2 or JBoss 5.0, both of which bundle the JSF 1.2 reference implementation. However, it is still possible to use Seam 2 on the JBoss 4.0 platform. There are two basic steps required to do this: install an EJB3-enabled version of JBoss 4.0 and replace MyFaces with the JSF 1.2 reference implementation. Once you complete these steps, Seam 2 applications can be deployed to JBoss 4.0.

2.9.1. JBoss 4.0

JBoss 4.0 does not ship a default configuration compatible with Seam. To run Seam, you must install JBoss 4.0.5 using the JEMS 1.2 installer with the ejb3 profile selected. Seam will not run with an installation that doesn't include EJB3 support. The JEMS installer can be downloaded from *http://www.jboss.org/jbossas/downloads*.

2.9.2. JSF 1.2 RI

JBoss 4.0 # Web ### `server/default/deploy/jbossweb-tomcat55.sar` ###### `jsf-libs` ######## `myfaces-api.jar` # `myfaces-impl.jar` ############# ########### `jsf-api.jar#jsf-impl.jar#el-api.jar` # `el-ri.jar` ############## JSF # JAR # Seam `lib` ############ el # JAR # Seam 1.2 #############

`conf/web.xml` ############# `myfaces-impl.jar` # `jsf-impl.jar`

JBoss Tools #### Seam

JBoss Tools # Eclipse ############## JBoss Tools # Seam ###############facelets # Java #### Unified Expression Language (EL) #########jPDL################Seam ######################Eclipse ## Seam #################

#######Eclipse #########JBoss Tools ###########

seam-gen ## JBoss Tools # JBoss AS ## ################## seam-gen ##########

3.1.

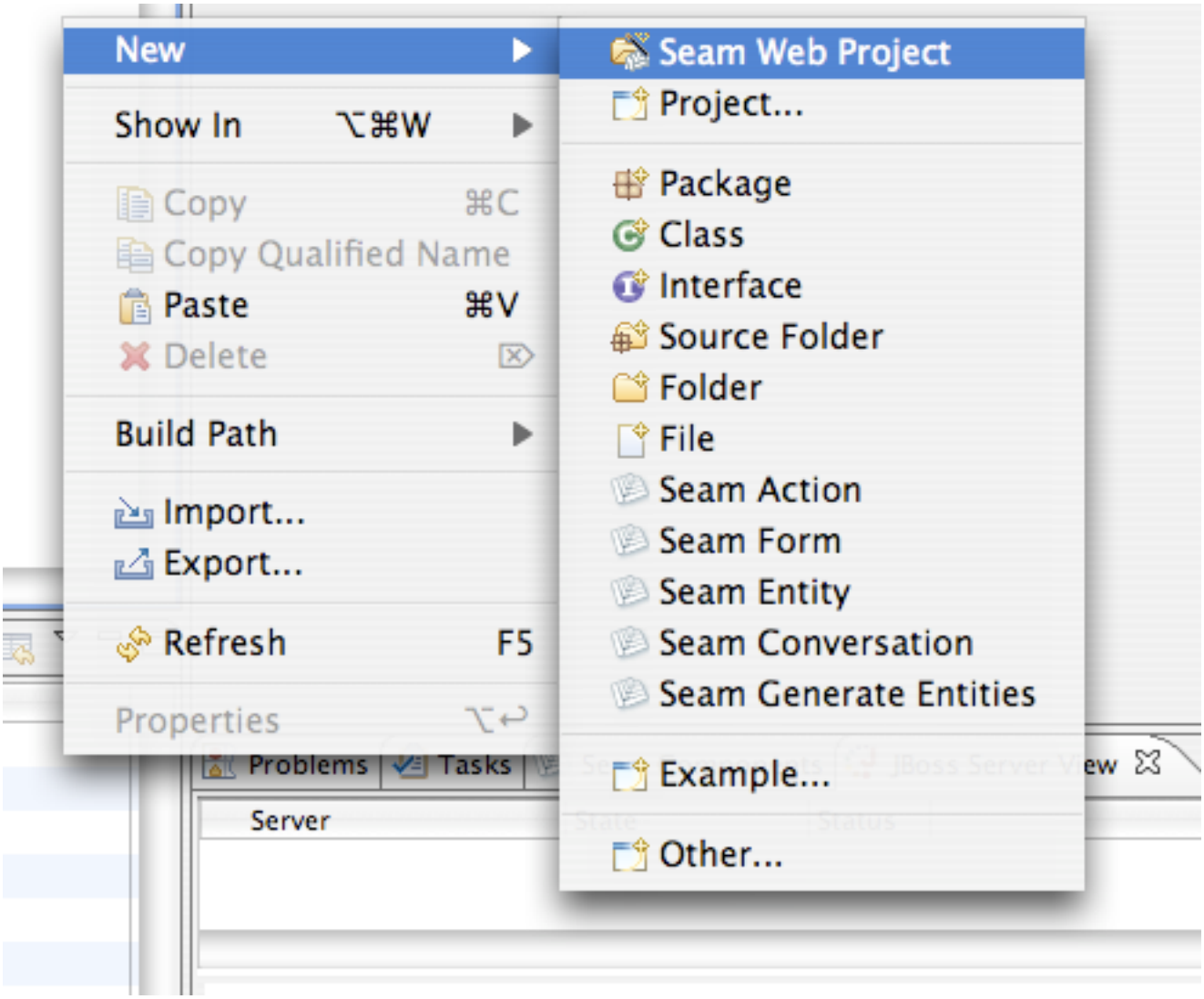
Make sure you have JDK 5, JBoss AS 4.2 or 5.0, Eclipse 3.3, the JBoss Tools plugins (at least Seam Tools, the Visual Page Editor, jBPM Tools and JBoss AS Tools) and the TestNG plugin for Eclipse correctly installed before starting.

Please see the official *JBoss Tools installation* [http://www.jboss.org/tools/download/installation] page for the quickest way to get JBoss Tools setup in Eclipse. You can also check out the *Installing JBoss Tools* [http://www.jboss.org/community/wiki/InstallingJBossTools] page on the JBoss community wiki for the gory details and a set of alternative installation approaches.

3.2. ### Eclipse

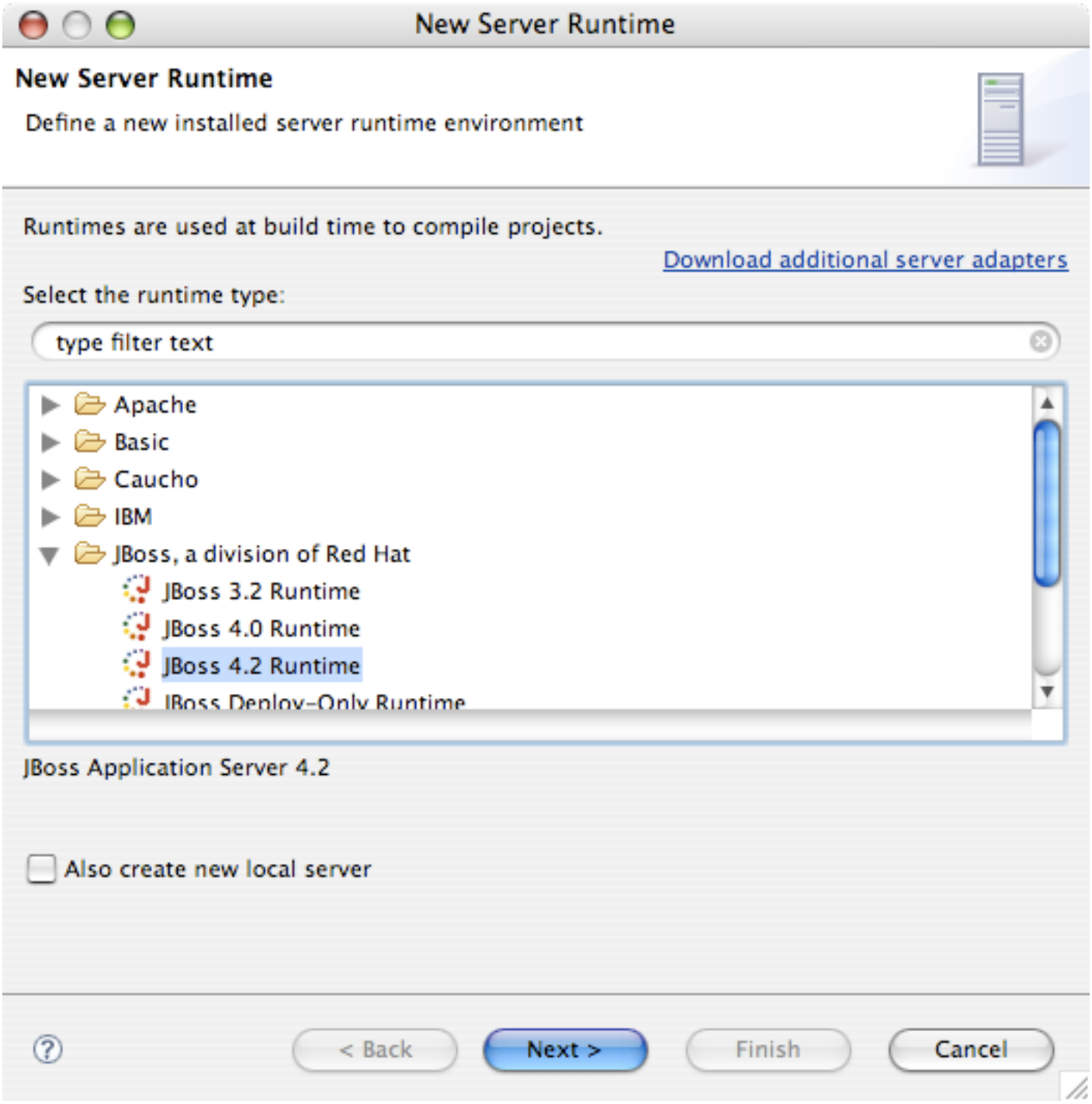
Eclipse ##### *Seam* ##################

File -> New -> Seam Web Project #######

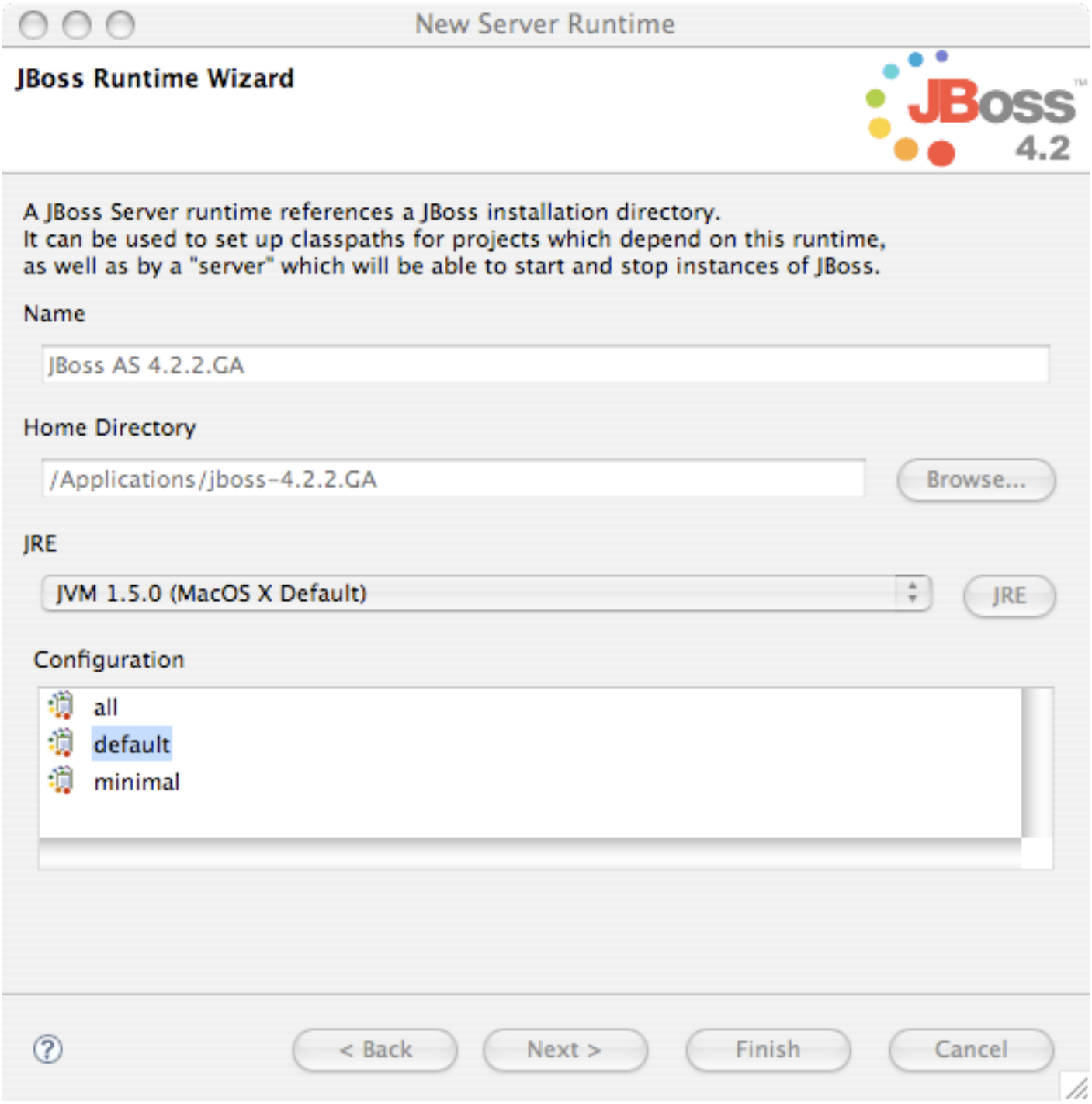


################## ########### `helloworld` #####

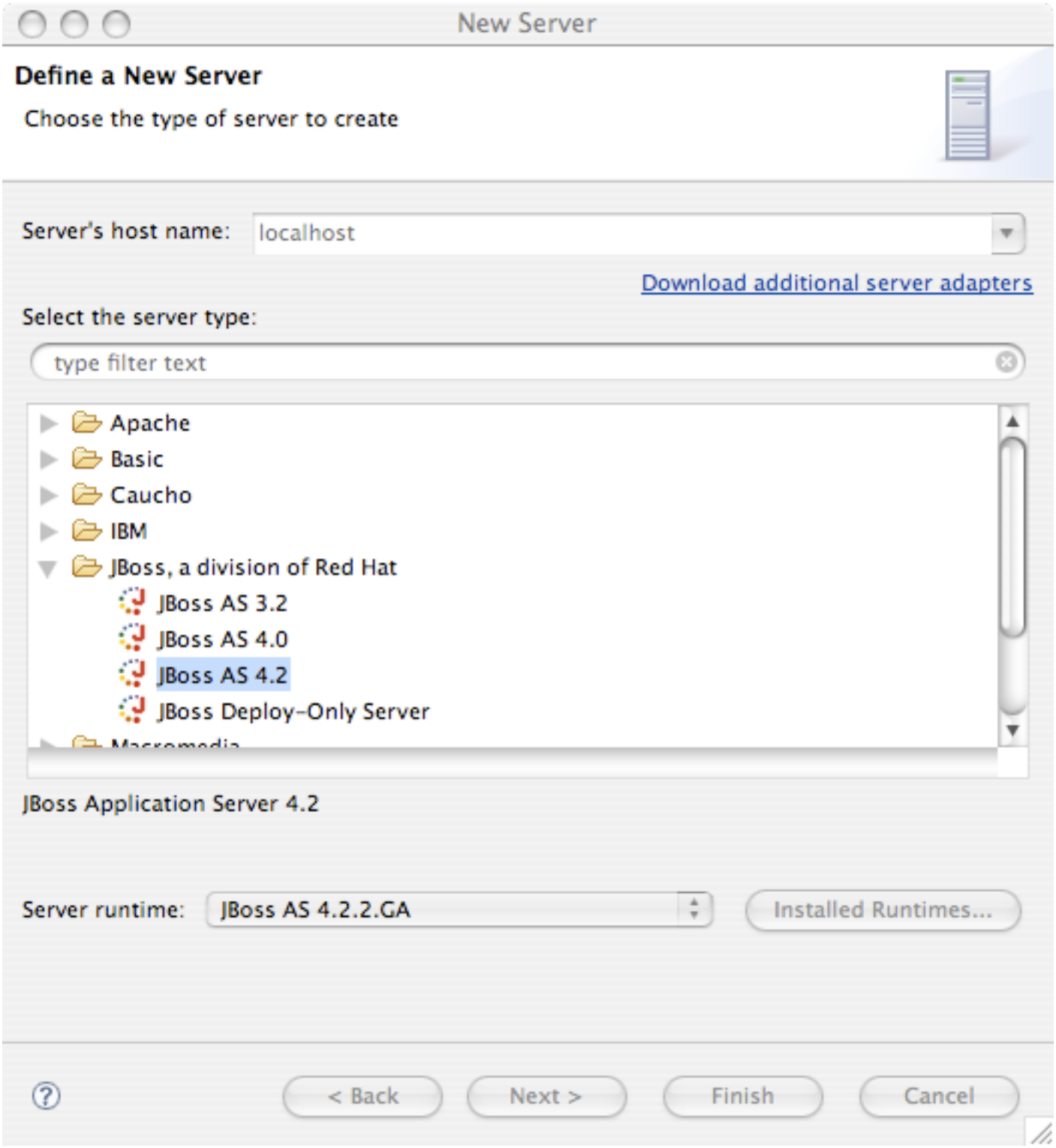
Now, we need to tell JBoss Tools about JBoss AS. In this example we are using JBoss AS 4.2, though you can certainly use JBoss AS 5.0 as well. Selecting JBoss AS is a two step process. First we need to define a runtime. Again, we'll choose JBoss AS 4.2 in this case:


New Server Runtime
New Server Runtime
Define a new installed server runtime environment
Runtimes are used at build time to compile projects.
Download additional server adapters
Select the runtime type:
type filter text
Apache
Basic
Caucho
IBM
JBoss, a division of Red Hat
JBoss 3.2 Runtime
JBoss 4.0 Runtime
JBoss 4.2 Runtime
JBoss Deploy-Only Runtime
JBoss Application Server 4.2
Also create new local server
< Back
Next >
Finish
Cancel

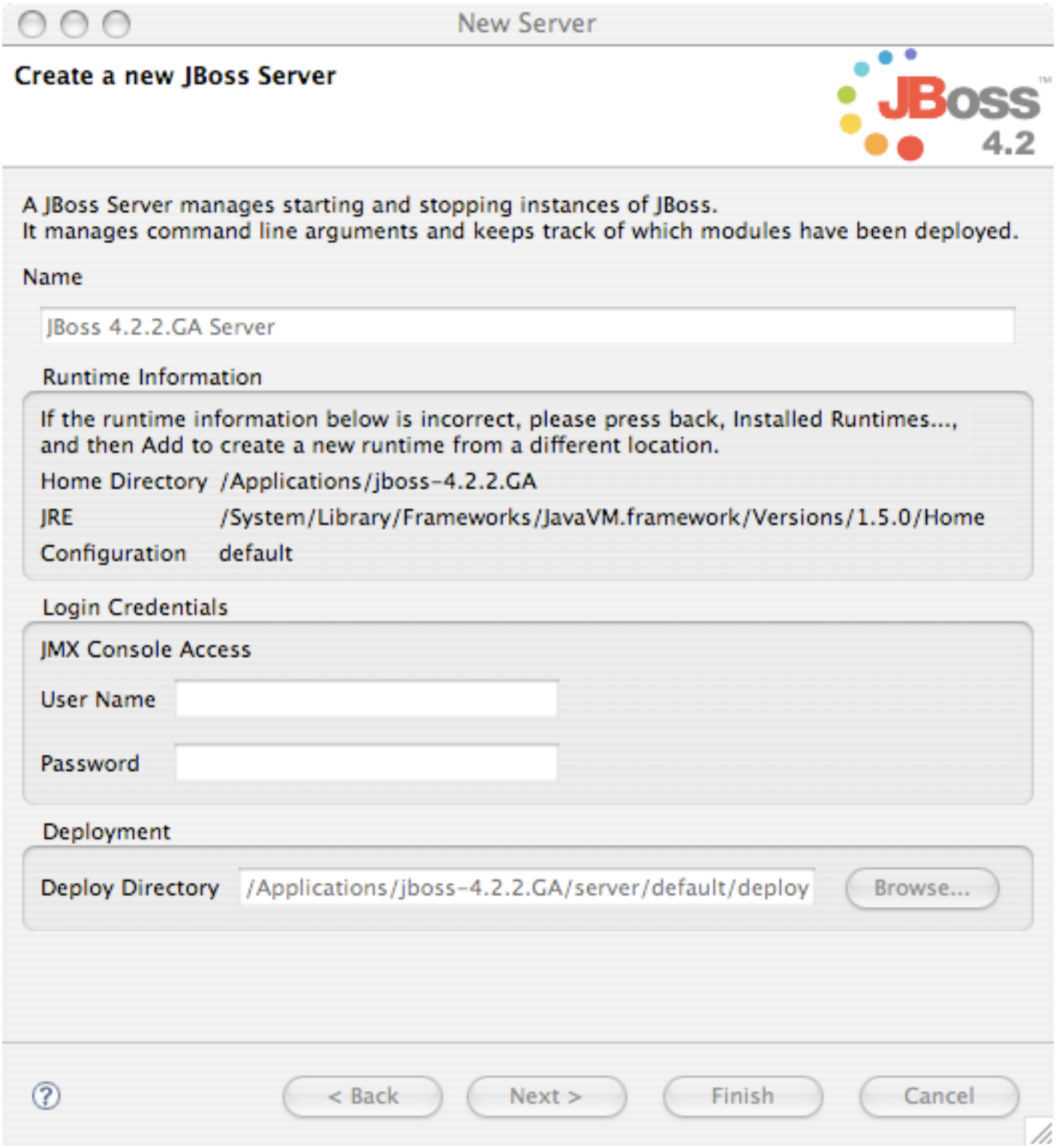
##################################



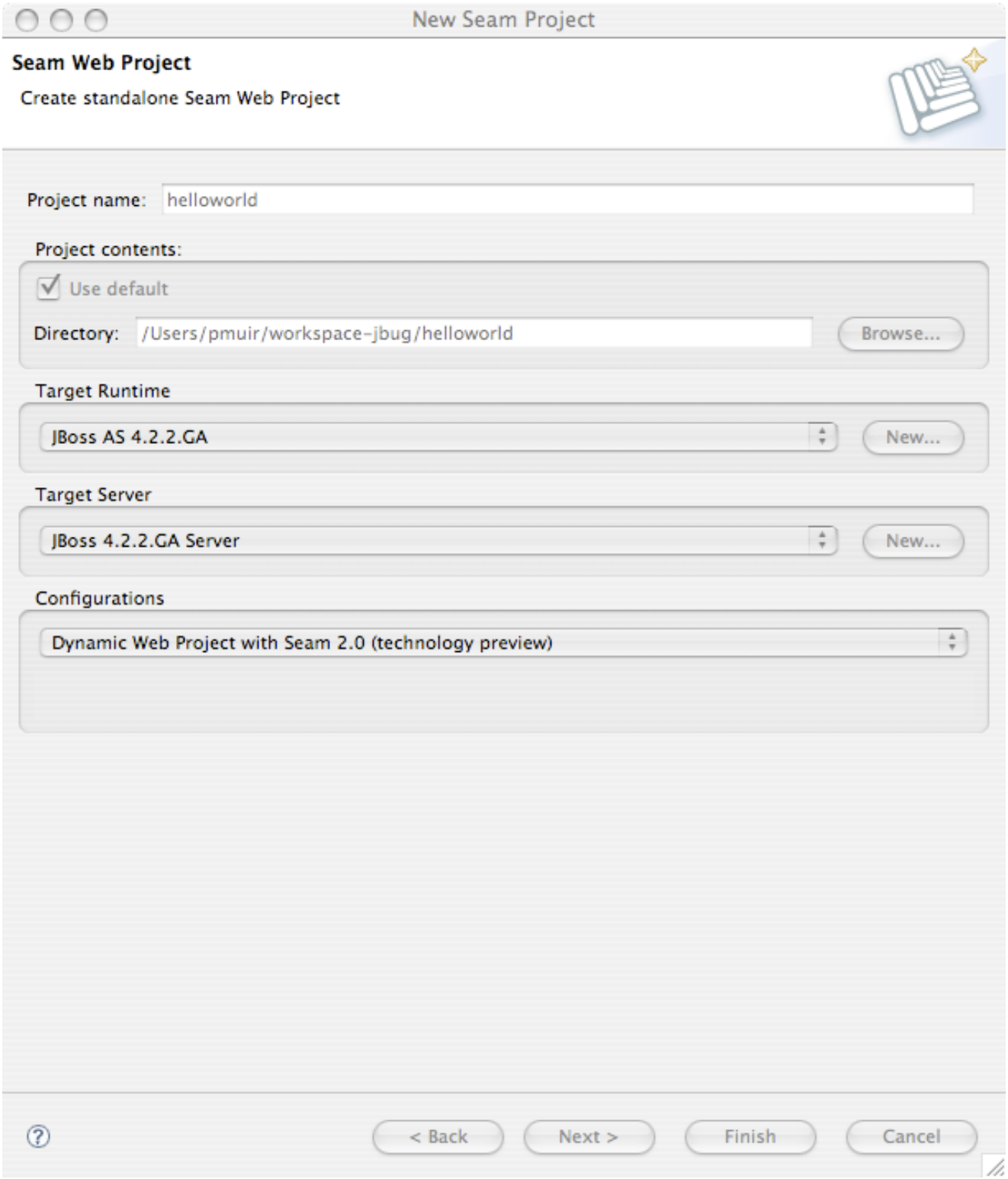
###JBoss Tools ################################ #### JBoss AS 4.2 ########################


New Server
Define a New Server
Choose the type of server to create
Server's host name:
localhost
Download additional server adapters
Select the server type:
type filter text
Apache
Basic
Caucho
IBM
JBoss, a division of Red Hat
JBoss AS 3.2
JBoss AS 4.0
JBoss AS 4.2
JBoss Deploy-Only Server
JBoss Application Server 4.2
Server runtime:
JBoss AS 4.2.2.GA
Installed Runtimes...
< Back
Next >
Finish
Cancel

##################*Finish* #########

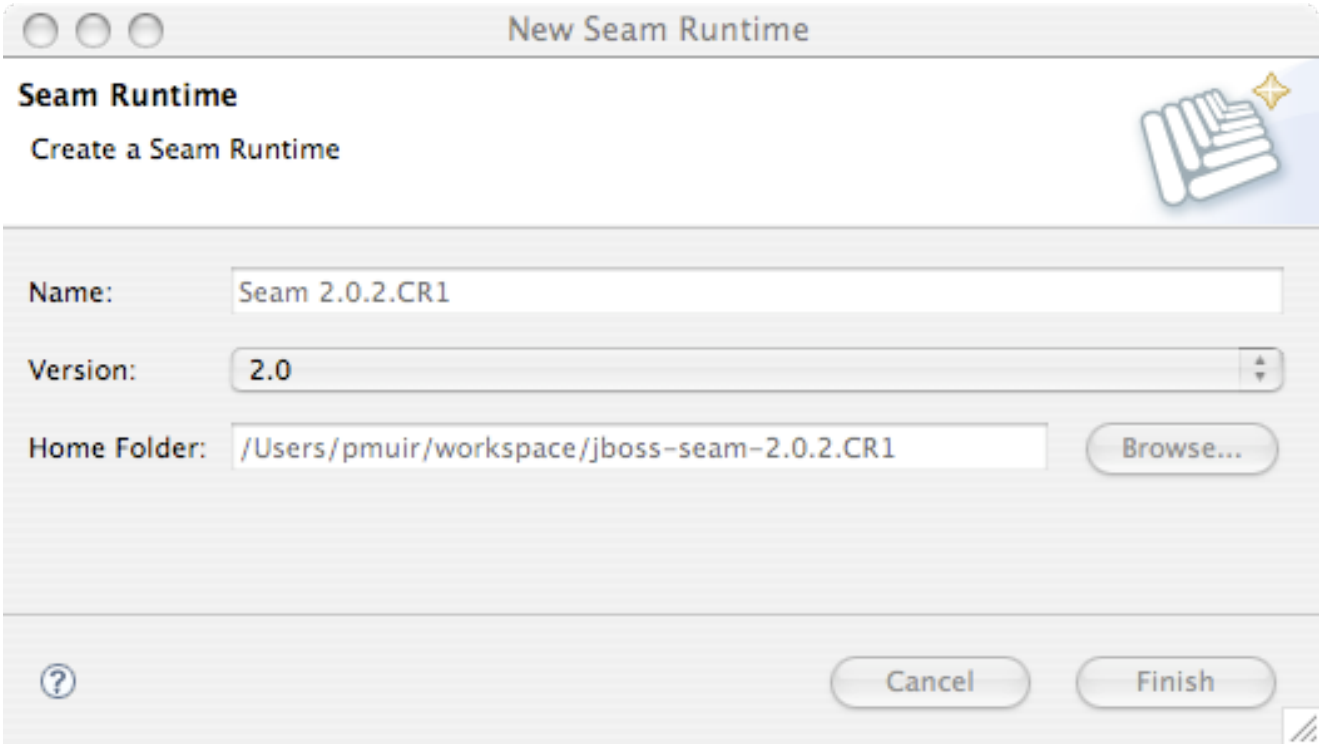


##########################Dynamic Web Project with Seam 2.0 (technology preview) ##### Next #########



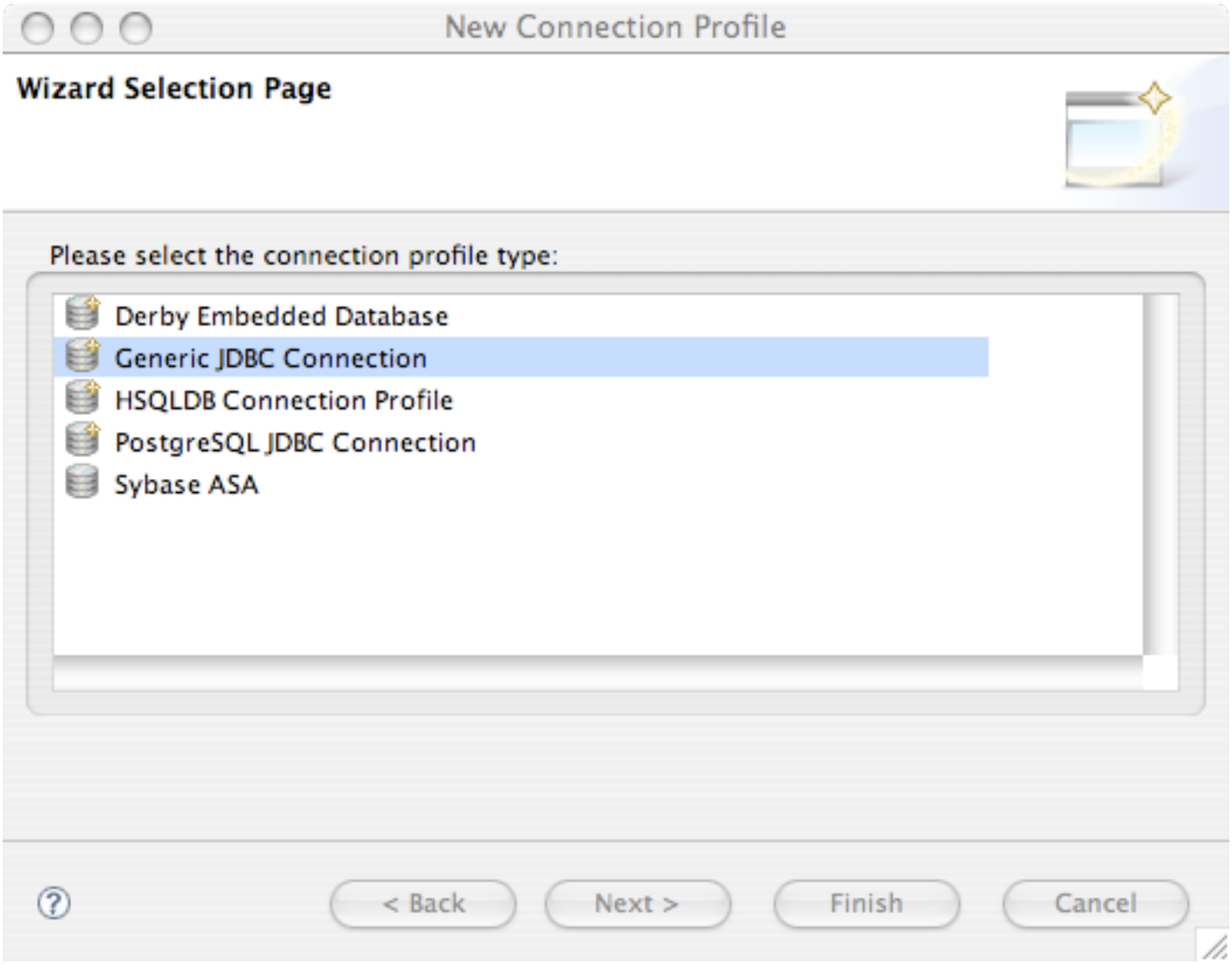
The next 3 screens allow you to further customize your new project, but for us the defaults are fine. So just hit *Next* until you reach the final screen.

######## JBoss Tools ########## Seam ############ ### *Seam Runtime* ####### - ################ *2.0* ##########

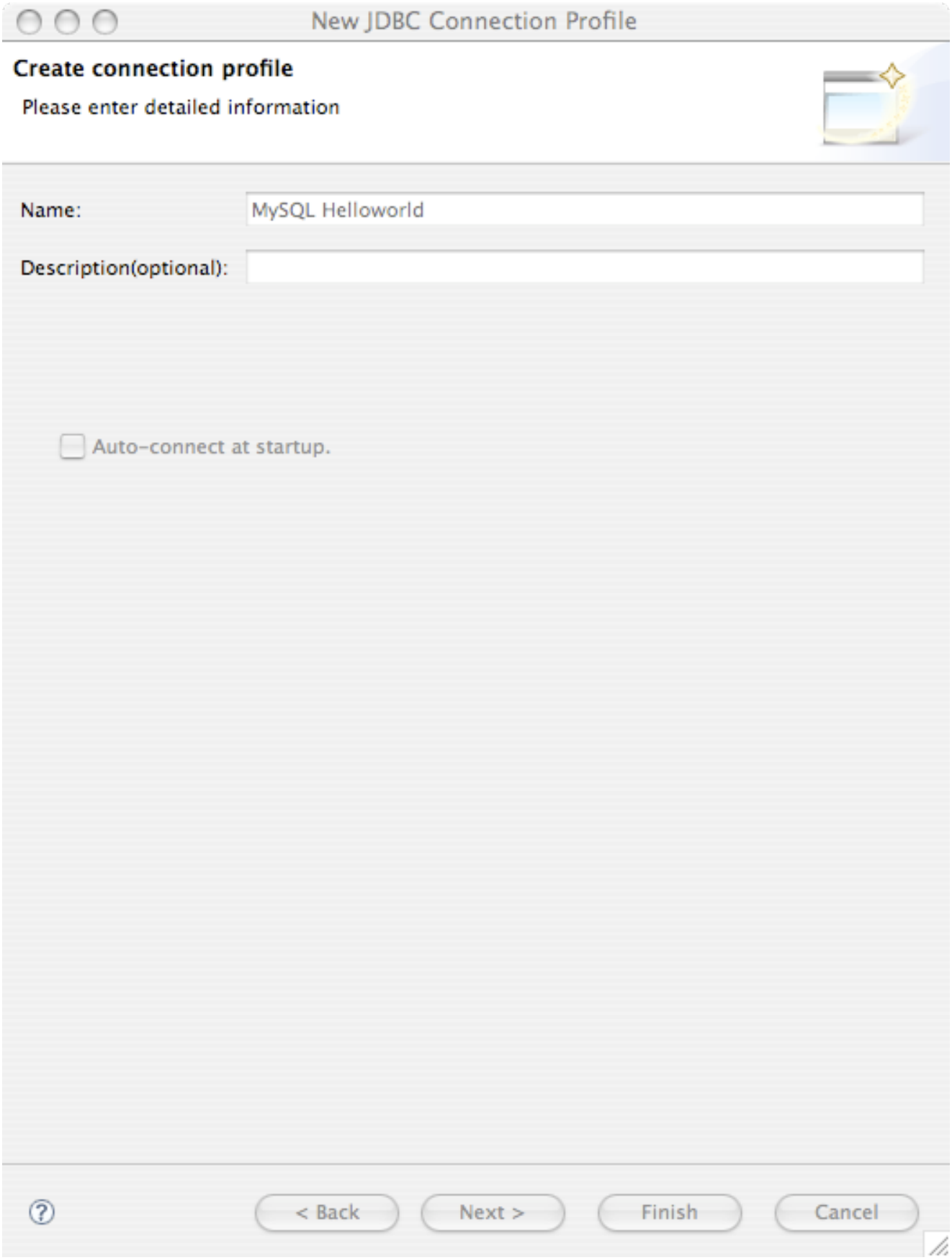


######################### EAR ####### WAR ################## EAR ####### EJB 3.0 #### Java EE 5 ###### WAR ####### EJB 3.0 ########## J2EE ########### WAR # EAR ###################### JBoss #### EJB3 ############################### EAR ########## ##### WAR ########## ### ########### WAR ###################### EAR ##################################

#################### ############## MySQL ###################### JBoss Tools ################################## *MySQL* ################################ *Generic JDBC Connection* ##########

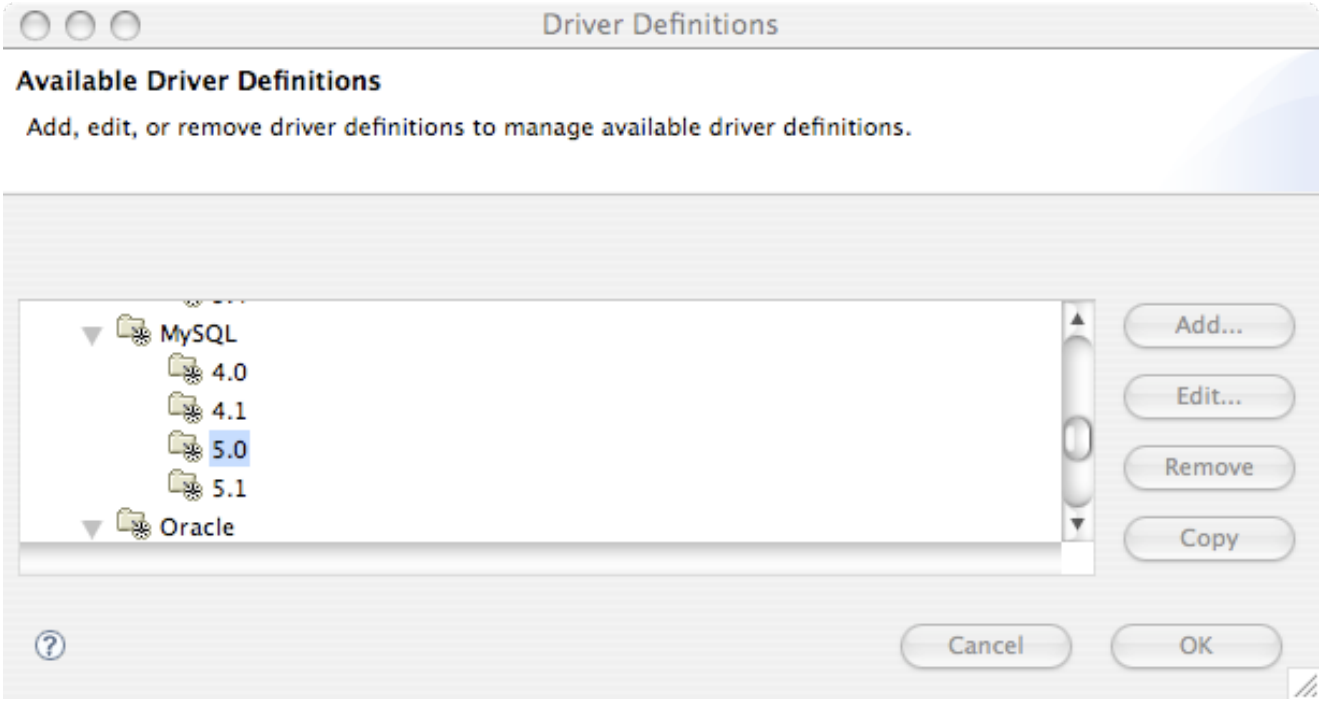

New Connection Profile
Wizard Selection Page
Please select the connection profile type:
Derby Embedded Database
Generic JDBC Connection
HSQLDB Connection Profile
PostgreSQL JDBC Connection
Sybase ASA
< Back
Next >
Finish
Cancel

############

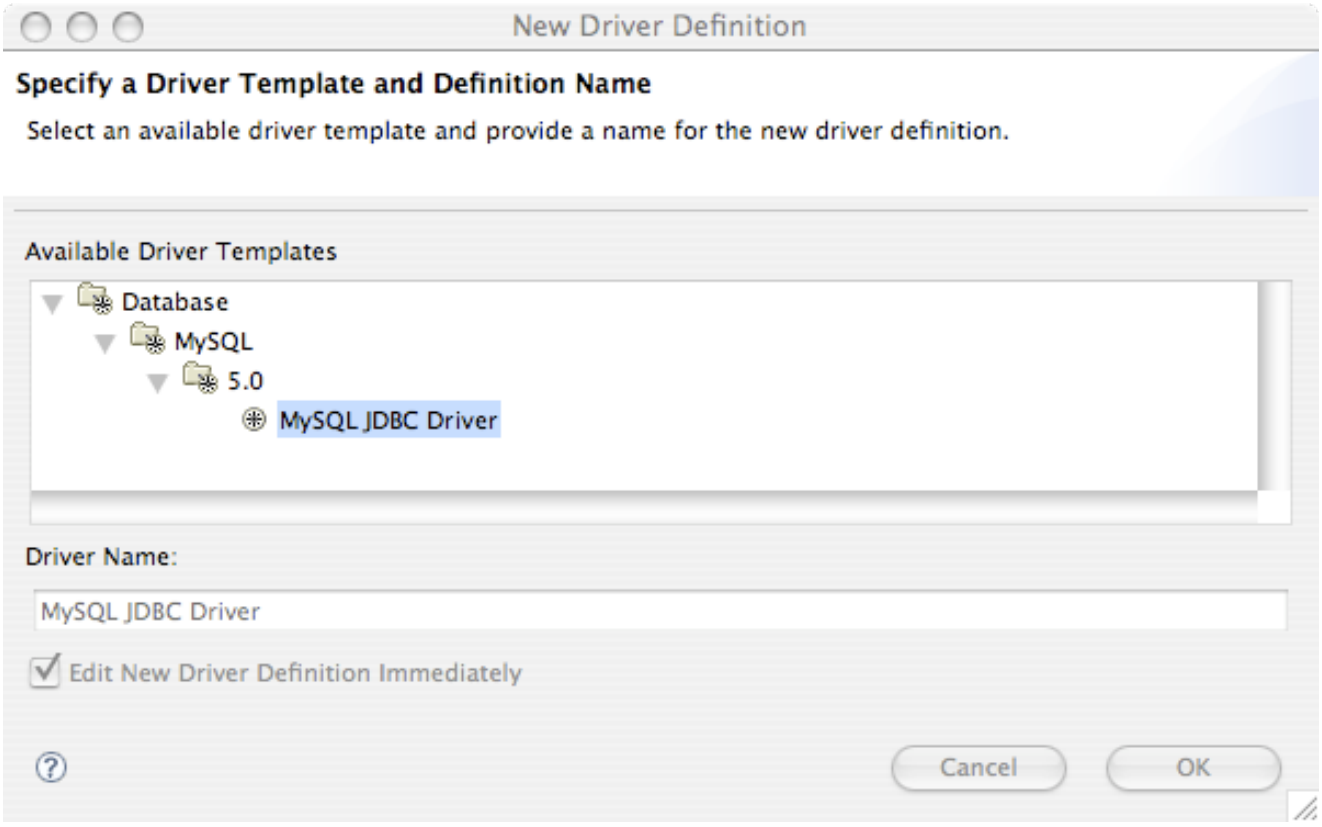

New JDBC Connection Profile
Create connection profile
Please enter detailed information
Name:
MySQL Helloworld
Description(optional):
Auto-connect at startup.
< Back
Next >
Finish
Cancel

JBoss Tools ######################JBoss Tools #### MySQL JDBC ################## ##########################

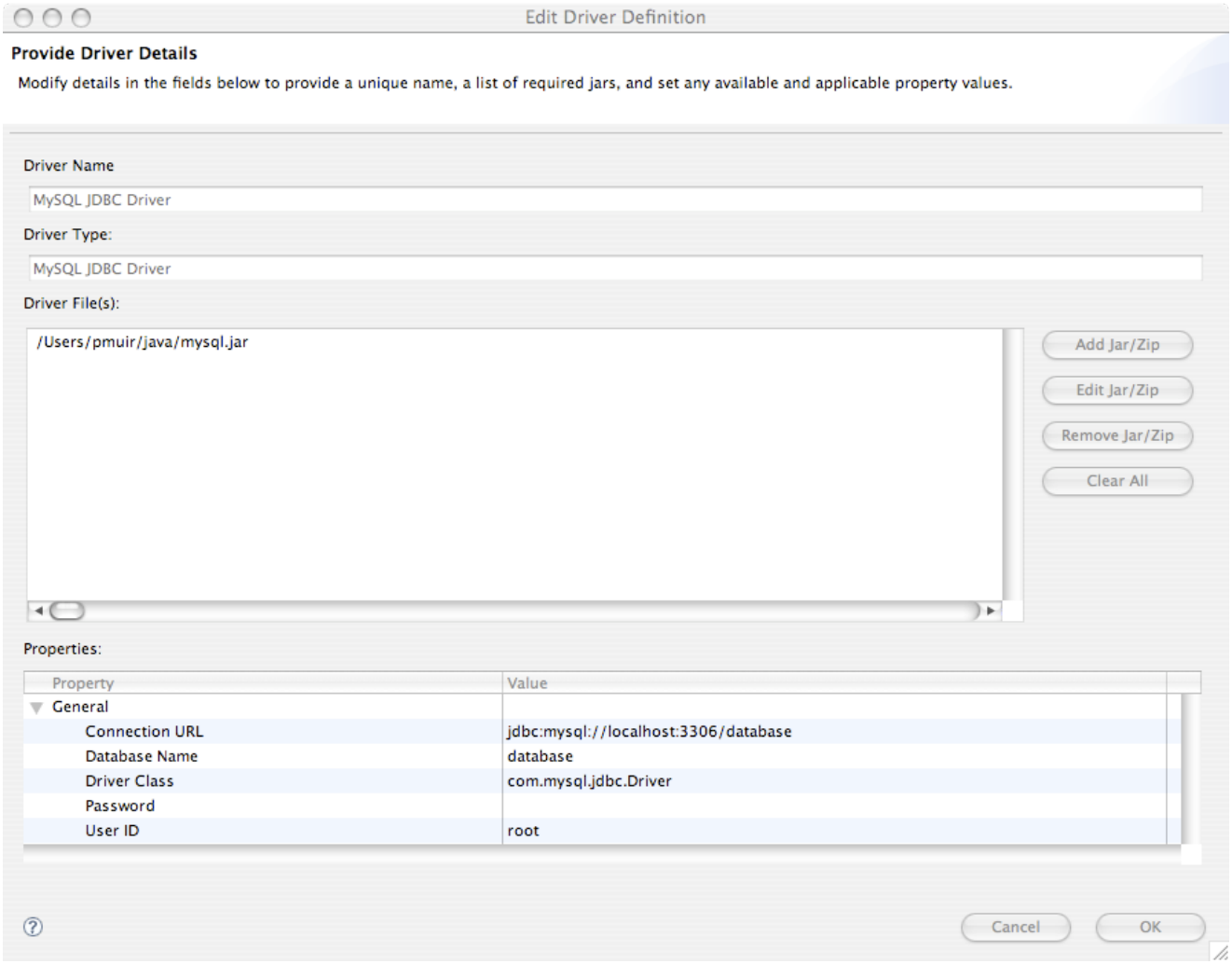
MySQL 5 *#####Add...* #########



MySQL JDBC Driver #################

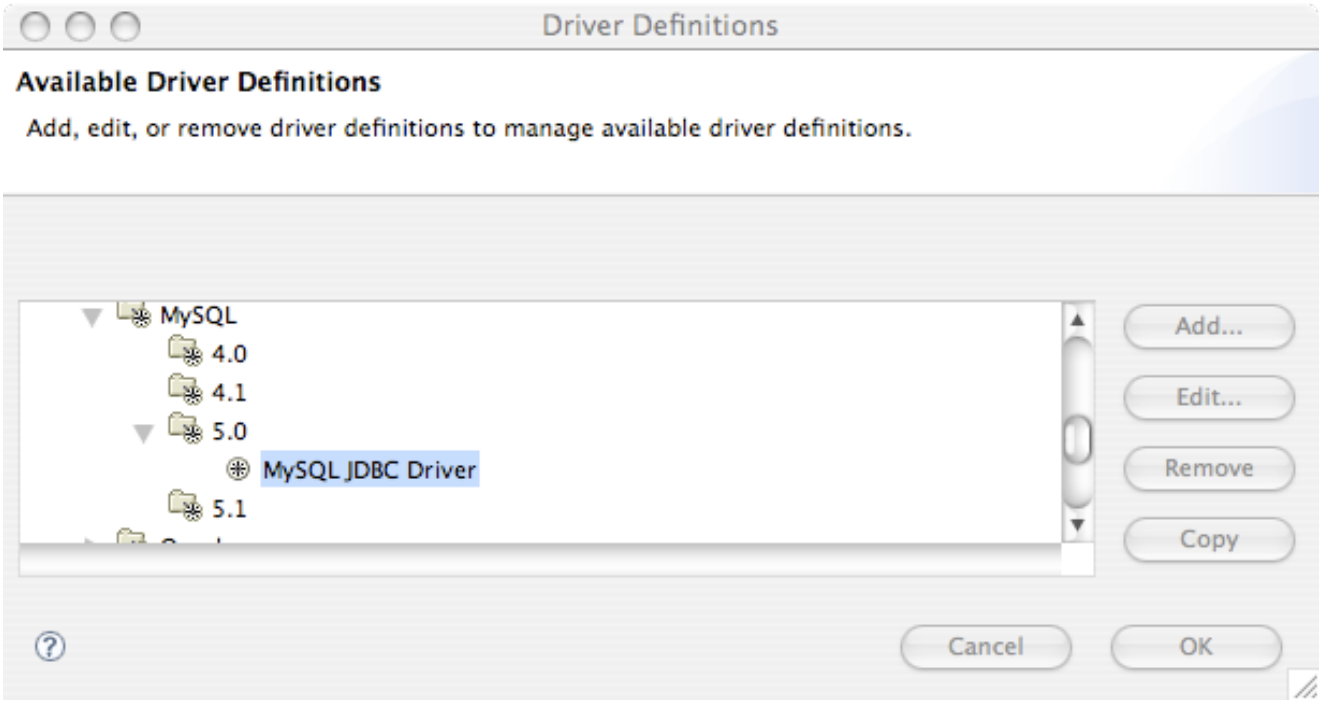


Edit Jar/Zip ################## jar ##############



################################Ok #########

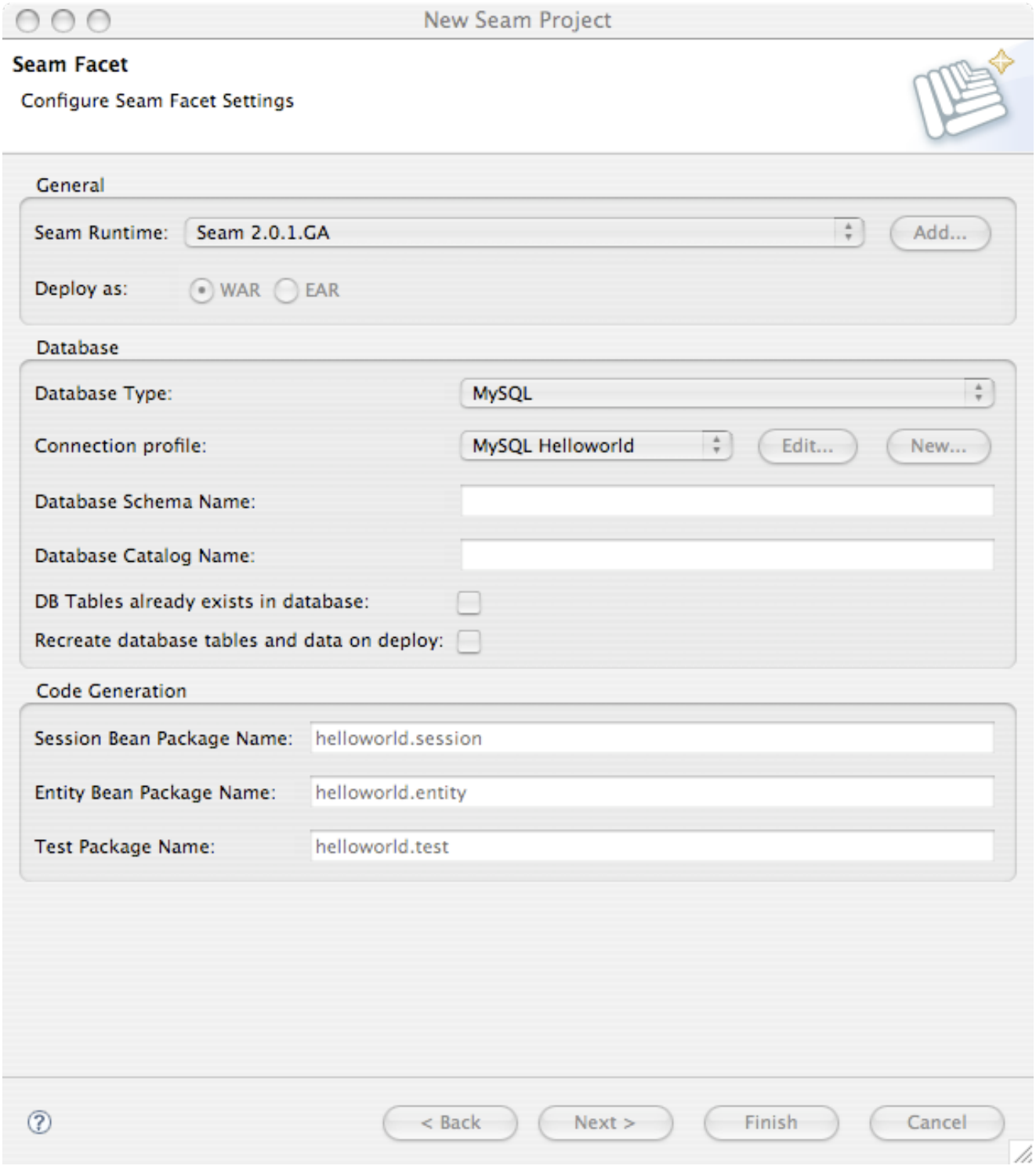
############################



###################### ######################## JBoss Tools #############

########################*Test Connection* ################## ########*Finish* ######

######### Bean ####################*Finish* #########



JBoss # WAR # EAR ########################## ###### JVM ######### — ########## — EAR ##################### JVM # perm gen ######################## ################## perm gen space ######## JVM # JBoss ################ ###############

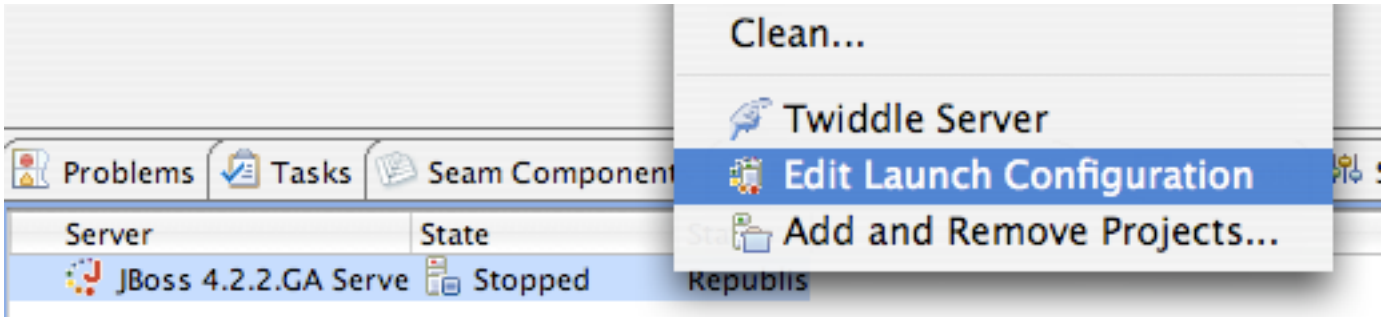
```
-Xms512m -Xmx1024m -XX:PermSize=256m -XX:MaxPermSize=512
```

```
```

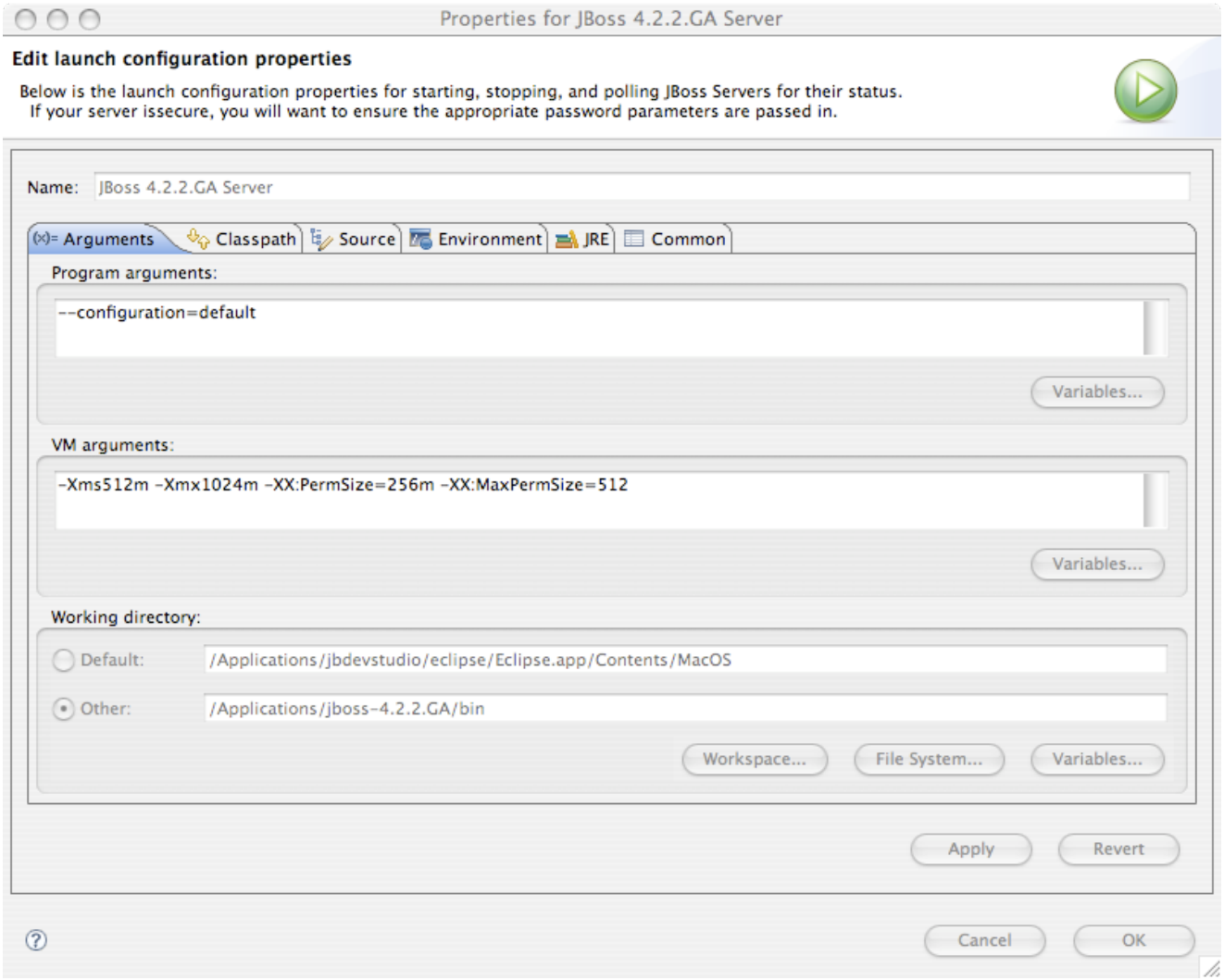
############################

```
-Xms256m -Xmx512m -XX:PermSize=128m -XX:MaxPermSize=256
```

JBoss Server View ######################## Edit Launch Configuration ###########

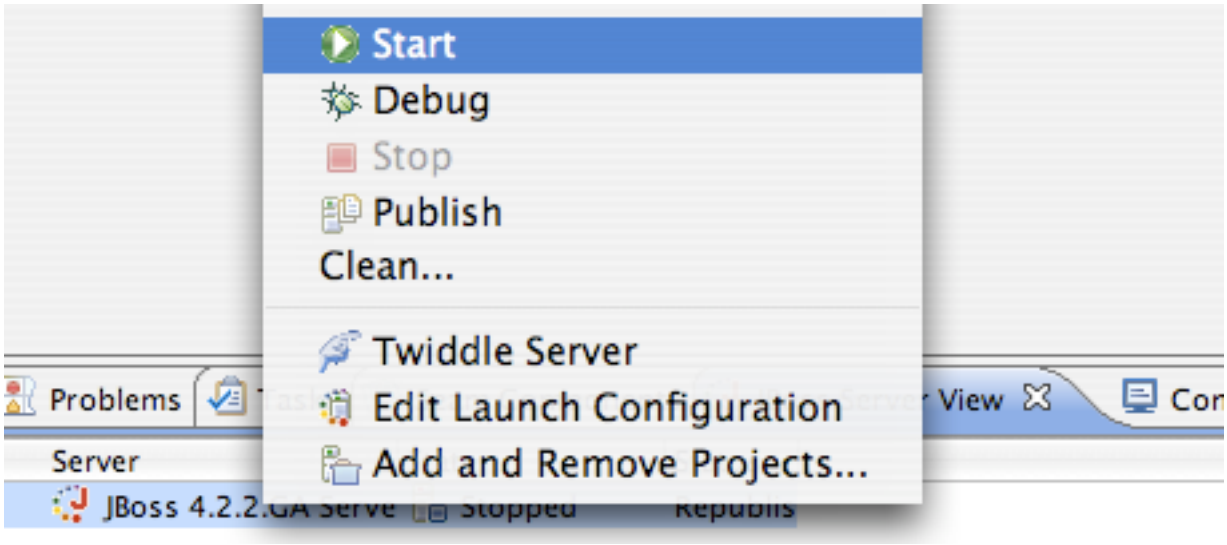


####VM #############



############################## — `OutOfMemoryException` ##################

JBoss ## *Start* ############ ################## *Debug* ############

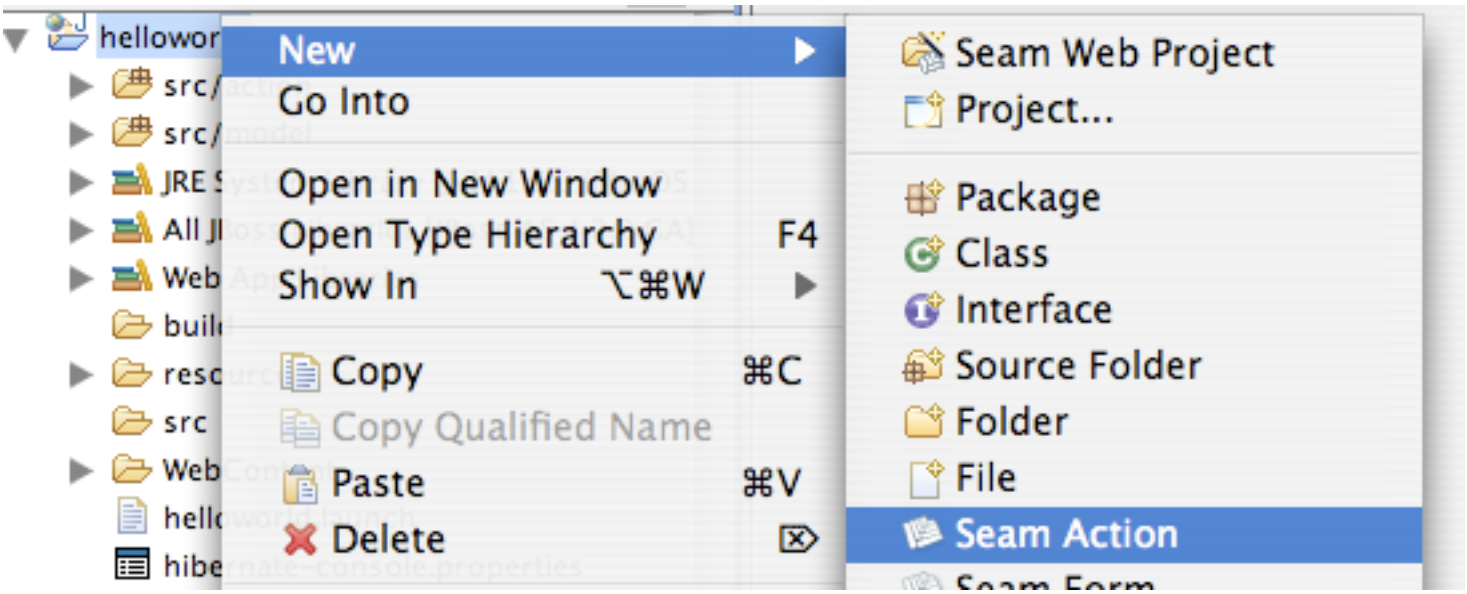


############## XML ############################## ############ Java EE ######## ######## ################ #### Seam ######## 90% #########

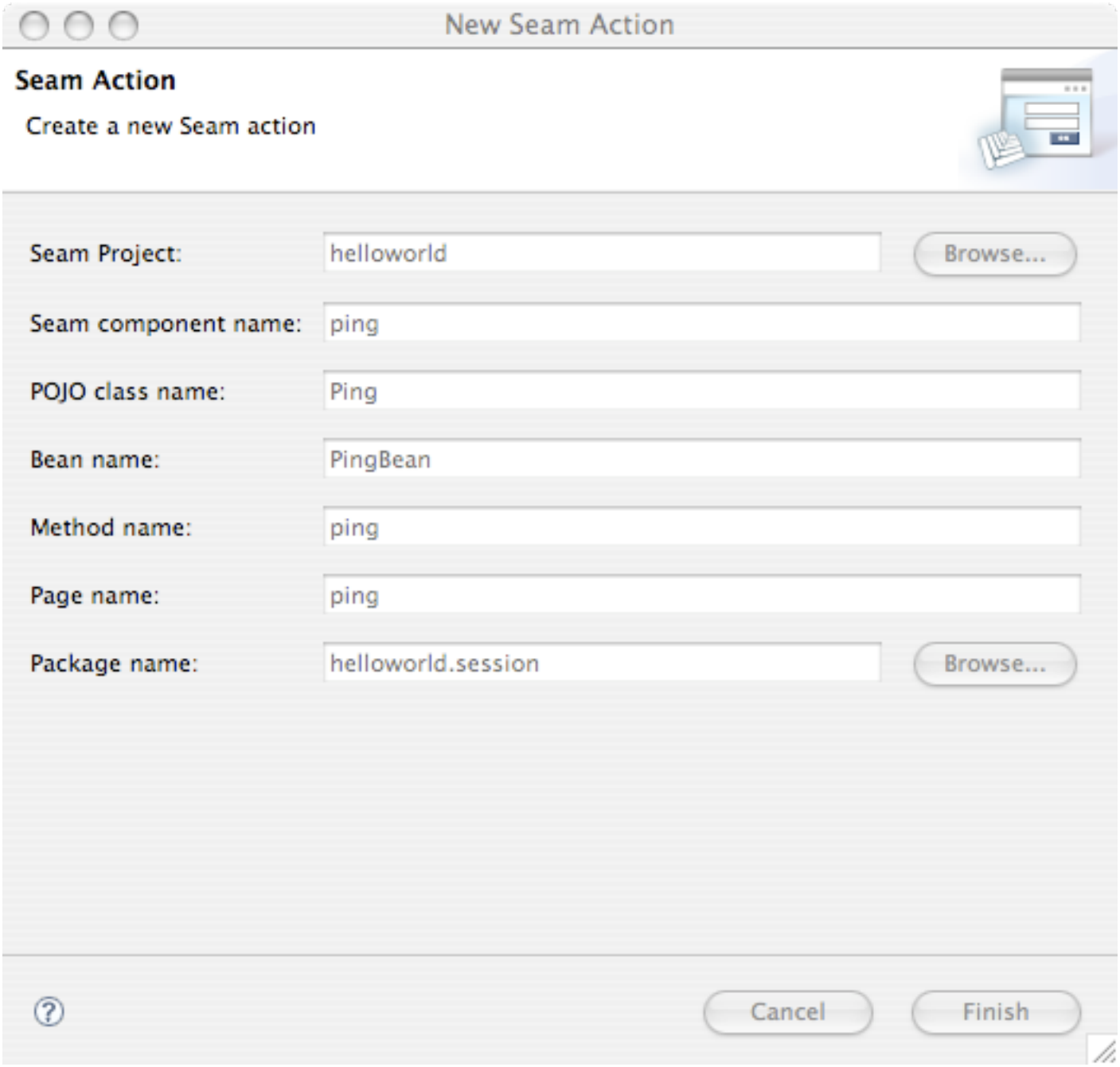
3.3.

############## Web ################## ######### Java ######################### Web ##########################

####New -> Seam Action ###########



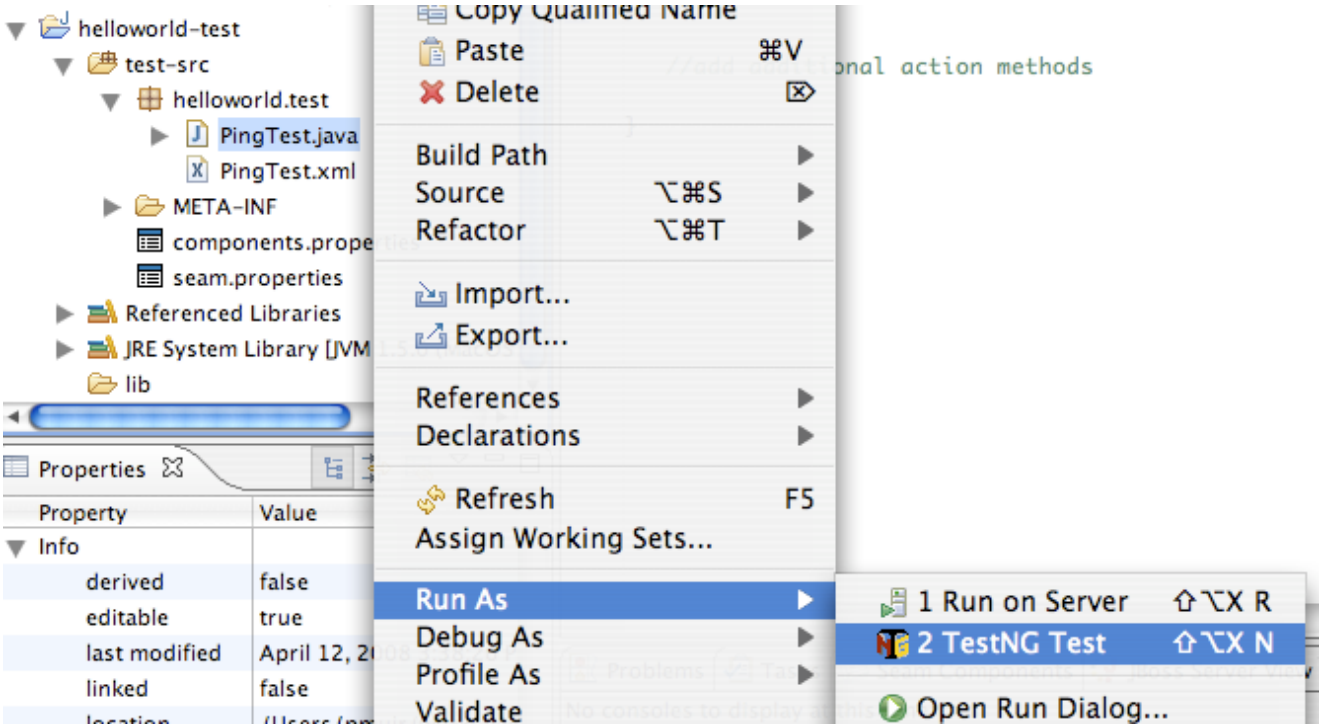
#####Seam ################### JBoss Tools ##############################



Finish

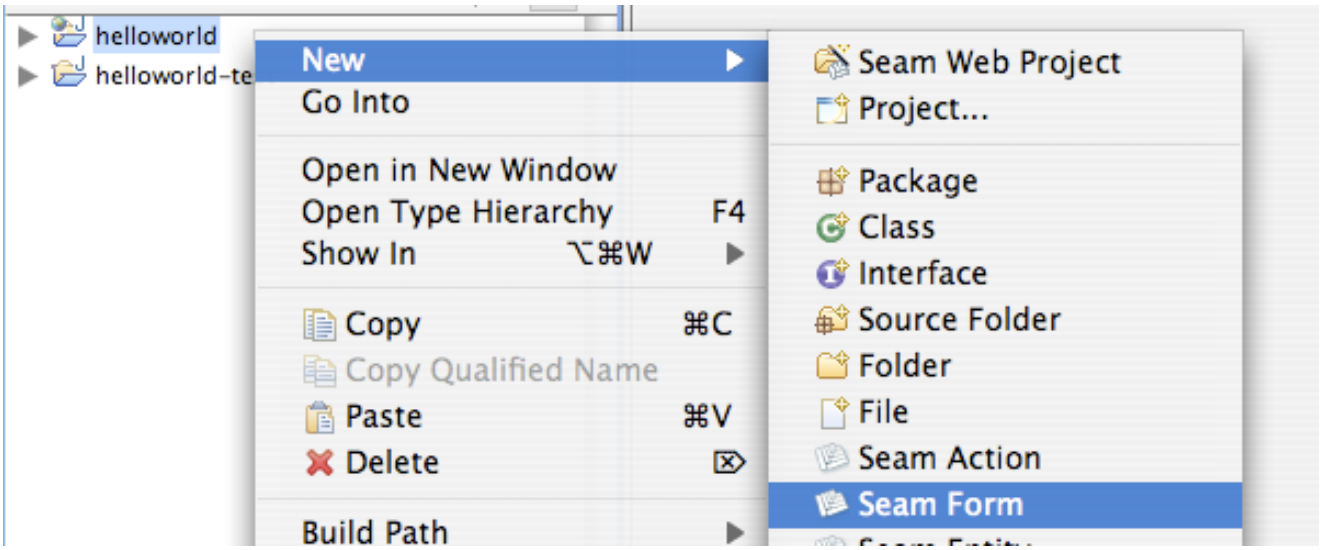
###http://localhost:8080/helloworld/ping.seam ###################### ####### src directory ###################################### ping() ################## ################

####helloworld-test ###########PingTest ######################*Run As -> TestNG Test* ##########

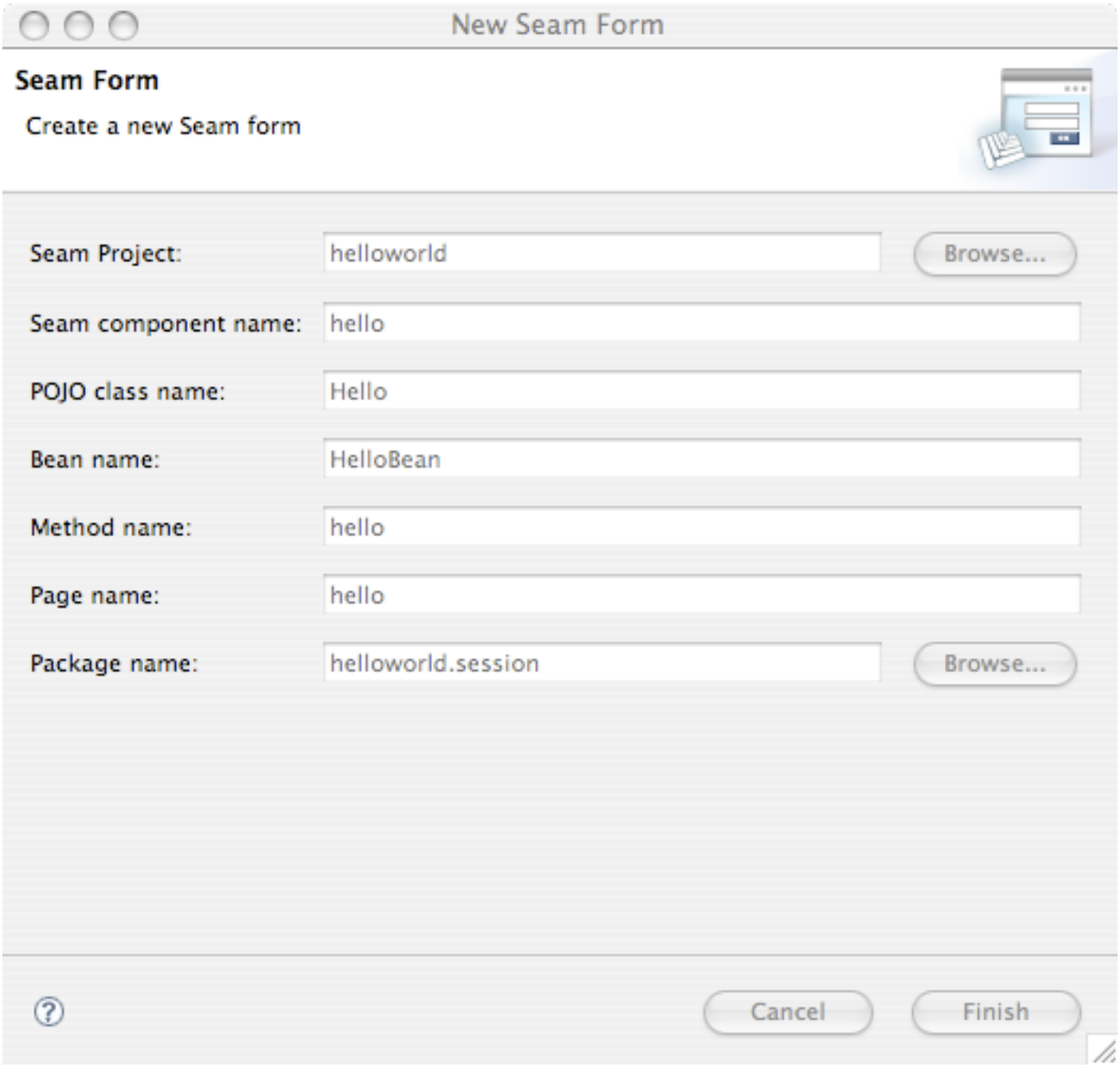


3.4.

######################## New -> Seam Form ###########



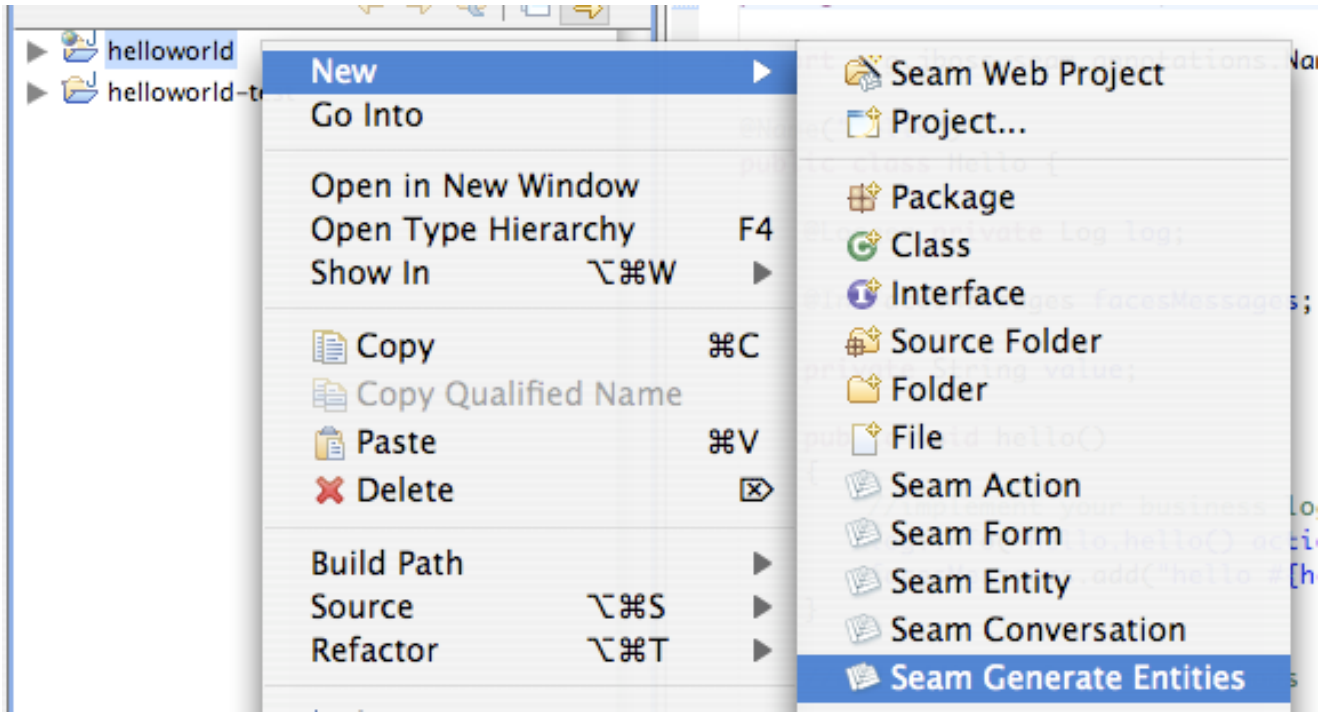
#####Seam ################### JBoss Tools ###############################



`http://localhost:8080/helloworld/hello.seam` ######### ################## ############# ##### Seam ##############################(Seam ################ *#3.6. #Seam # JBoss Tools ################* ##### `src/hot` ##)

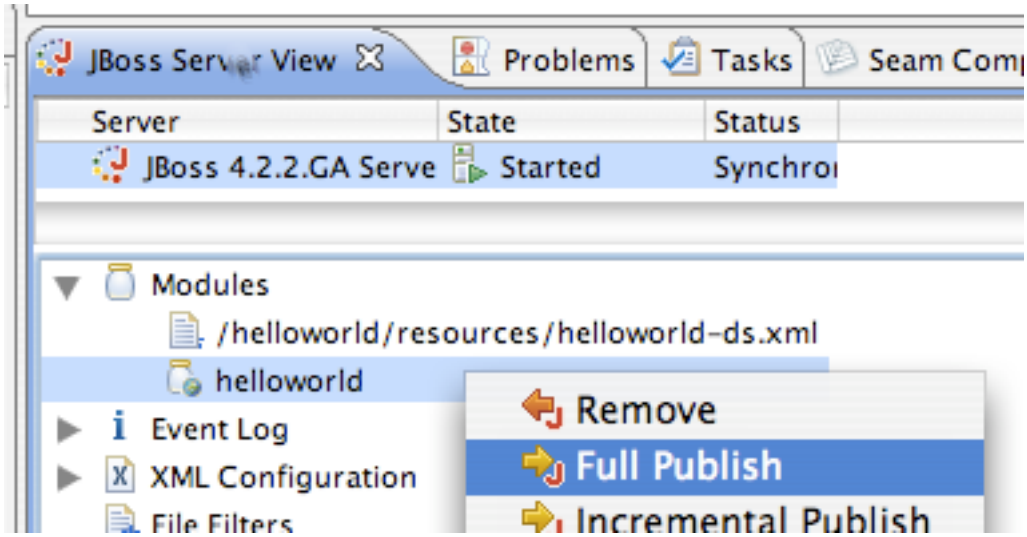
3.5.

######################
(##) *####New -> Seam Generate Entities #######*



JBoss Tools ##
JPA ##
##################################

#############



`http://localhost:8080/helloworld` ##########
##################################
Seam ################################## ##################################

3.6. Seam # JBoss Tools

JBoss Tools ############################

- facelets ###
- `pages.xml` ####

out of the box.

Java ##############*Full Publish*

########/#####/################ Seam # JavaBean ############################## ############### JavaBean ######## `WEB-INF/dev` ###################### ############# WAR #### EAR ########### ### Seam

###################

- ######## JavaBean ################ EJB3 Bean ###### (###########)
- ####################
- `components.xml` ###################################
- ################### `WEB-INF/dev` ######################
- Seam ########### `jboss-seam-debug.jar` # `WEB-INF/lib` ###########
- web.xml # Seam ######################
- ##

JBoss Tools ##### WAR ############## ##############`src/hot` ############################ #### JBoss Tools # EAR ############################

##################

Seam #### 2 ######### ###### #### ####### ###### ########################## EJB ### ## ######## (Bijection) ############## (########) ################### Seam ##

Seam ########################

4.1. Seam

Seam ############################ ######### Java API ################# (demarcation) ################## ##

Seam

- #############
- Event (i.e., request) context
- #########
- ########
- ############
- ###############
- ###############

2 ############### ######## *(conversation context)* ################## Web ################################## 1 ###### 3 ############ (#################) ############################## ## ########## Seam ##

########################

4.1.1.

Components which are truly stateless (stateless session beans, primarily) always live in the stateless context (which is basically the absense of a context since the instance Seam resolves is not stored). Stateless components are not very interesting, and are arguably not very object-

oriented. Nevertheless, they do get developed and used and are thus an important part of any Seam application.

4.1.2.

############################## ############## Web ################## ##########JSF ############################## #################################### ###################################### ####################################

RMI ###### Seam Remoting ### Seam ############### ##################################

4.1.3.

#####################################

4.1.4.

######### Seam ################# ## *(conversation)* ################### ## #################1 ########### ################################## ######## 1 ##

1 ##

##

####### 1 ############# #############Seam #####################################

Seam

####### "##" ############ ### ###### ##########

########################### Seam ############### Seam ###### ######## *(conversation timeout)* ##### ############### ###

Seam ##

#####Seam ##

4.1.5.

################################## ################################## ## ############################

JSR-168 ##

4.1.6.

BPM #### (JBoss jBPM) ################# ################################## ################################ ################################ ################# ######## *(process definition language)* #################

4.1.7.

####Seam ################# Seam

4.1.8.

#################### ########## ################################ ############################ ###Seam ##############################

############## ######################## (######################) ######## `Contexts` ## ### `Context` ####################################

```
User user = (User) Contexts.getSessionContext().get("user");
```

############################

```
Contexts.getSessionContext().set("user", user);
```

##

4.1.9.

##########

- ##########
- #########
- ########
- ############
- ###############
- ###############

`Contexts.lookupInStatefulContexts()` ######################## JSF ######################################

4.1.10.

######### EJB ## ################################## ############################## EJB ### ######################## Bean ####################

################################ Web ######################### ################## (AJAX) ###################### ## Seam ############################

Seam ################################### Seam ################################## ############################## ################################ ## Seam ############## ################## ##########

################################## ##########Seam###################Seam ## ##Seam ########## ############## Bean # JavaBean ########## (############################) ## ## ################## ### ############ #### `@Synchronized` ################ ##### Bean ### JavaBean ################################

########## ########################## AJAX ######################################

4.2. Seam

Seam ######## POJO (Plain Old Java Objects) ### ######Seam ######## JavaBean #### EJB 3.0 ######## Bean ### Seam # ######## EJB ############### EJB 3.0

#################### Seam # EJB 3.0 ##########EJB 3.0 ########## Seam ########## #########

- EJB 3.0 ########## Bean
- EJB 3.0 ########## Bean
- EJB 3.0 entity beans (i.e., JPA entity classes)
- JavaBeans
- EJB 3.0 ######## Bean
- Spring beans (see *# 27. Spring Framework ##*)

4.2.1. ########## Bean

########## Bean ################################## ########## Seam ################################ #### JSF ########################## ###### JSF ##########################

########## Bean ######################

################################### Bean ###################### ################## EJB3 ########## (############################ ####Bean ##)

########## Bean ############ Seam ##########

Seam ########## Bean ######## `Component.getInstance()` ### `@In(create=true)` ###################### JNDI ####### ## new ##########################

4.2.2. ########## Bean

########## Bean ######### Bean ############################ ######################### ############################ ############## Bean ################ ### Seam ###### Web ######################### ########## `HttpSession` ########## ######################### Bean ###################### ####Seam ##################### ##################################

########## Bean ##### JSF ############# ############################## JSF ############ Bean #########

################## Bean ################# ############################

###################### Bean ########## ##Bean##Seam#################### Seam ###############

Seam ########## Bean ######## `Component.getInstance()` ### `@In(create=true)` ###################### JNDI ####### ## new ##########################

4.2.3. ###### Bean

Bean ################Seam#####################
################################## ###############Seam ###################
Java

Bean ################################## ######### Bean
##############################

Bean #### JSF #####################
JSF ###################### Bean ######### ######### Bean ###### Bean
################## ## / ## / ############################ Bean ###################

############## Bean ################## ########## Bean ##########

############### ########## Bean ######## Bean ############ ######
Bean ##################### Seam ######################################
############ Seam ######### Seam ################### Bean ###############

Seam ###### Bean ######## `Component.getInstance()` ### `@In(create=true)`
######################### `new` ##################

4.2.4. JavaBeans

JavaBean ###################### Bean ################ ############ Bean
############ (############ ########## ###################### EJB 3.0 ####
###########)

#####EJB ####### Seam # Hibernate ################ ######################## Bean
JavaBean ### ##################### ########## Bean
################ Seam JavaBean ######################################

#######JavaBean ###################

########## JavaBean ########### Seam ##############

Seam JavaBean ######## `Component.getInstance()` ### `@In(create=true)`
######################### `new` ########################

4.2.5. ######## Bean

######## Bean # Seam ##################### ########### Bean
Seam ######################### ######################## JMS
##############################

######## Bean ##Seam ################ ###################################
############ Bean ############ Seam ############

######## Bean ### EJB
###################

4.2.6.

Seam##### (############################) ######### Seam ################################ JavaBean ###Seam ## ###### Bean ## ##### Bean ###EJB ############ Bean ######################### ####################

```
@Stateless
@Interceptors(SeamInterceptor.class)
public class LoginAction implements Login {
   ...
}
```

##############ejb-jar.xml ##################

```
<interceptors>
  <interceptor>
    <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
  </interceptor>
</interceptors>

<assembly-descriptor>
  <interceptor-binding>
    <ejb-name
>*</ejb-name>
    <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
  </interceptor-binding>
</assembly-descriptor
>
```

4.2.7.

Seam ################ @Name

```
@Name("loginAction")
@Stateless
public class LoginAction implements Login {
   ...
```

```
}
```

######*Seam* ######## ## EJB ######################## ####Seam ############# JSF ## Bean ############ 2 ######################

`@Name` ############################## ######################## ############ Seam ######################

Whenever Seam instantiates a component, it binds the new instance to a variable in the scope configured for the component that matches the component name. This behavior is identical to how JSF managed beans work, except that Seam allows you to configure this mapping using annotations rather than XML. You can also programmatically bind a component to a context variable. This is useful if a particular component serves more than one role in the system. For example, the currently logged in `User` might be bound to the `currentUser` session context variable, while a `User` that is the subject of some administration functionality might be bound to the `user` conversation context variable. Be careful, though, because through a programmatic assignment, it's possible to overwrite a context variable that has a reference to a Seam component, potentially confusing matters.

For very large applications, and for built-in seam components, qualified component names are often used to avoid naming conflicts.

```
@Name("com.jboss.myapp.loginAction")
@Stateless
public class LoginAction implements Login {
   ...
}
```

Java ###### JSF ############################

```
<h:commandButton type="submit" value="Login"
                 action="#{com.jboss.myapp.loginAction.login}"/>
```

#############Seam ############################## ######## `components.xml` ###############

```
<factory name="loginAction" scope="STATELESS" value="#{com.jboss.myapp.loginAction}"/>
```

######## Seam ####################
Seam##Seam JAR##### `components.xml` ###################

```
<components xmlns="http://jboss.com/products/seam/components">

    <import>org.jboss.seam.core</import>
    <import>org.jboss.seam.cache</import>
    <import>org.jboss.seam.transaction</import>
    <import>org.jboss.seam.framework</import>
    <import>org.jboss.seam.web</import>
    <import>org.jboss.seam.faces</import>
    <import>org.jboss.seam.international</import>
    <import>org.jboss.seam.theme</import>
    <import>org.jboss.seam.pageflow</import>
    <import>org.jboss.seam.bpm</import>
    <import>org.jboss.seam.jms</import>
    <import>org.jboss.seam.mail</import>
    <import>org.jboss.seam.security</import>
    <import>org.jboss.seam.security.management</import>
    <import>org.jboss.seam.security.permission</import>
    <import>org.jboss.seam.captcha</import>
    <import>org.jboss.seam.excel.exporter</import>
    <!-- ... --->
</components>
```

################Seam##components.xml##########

4.2.8.

@Scope################################ (#######) ################### ##### Seam ################## ##

```
@Name("user")
@Entity
@Scope(SESSION)
public class User {
    ...
}
```

org.jboss.seam.ScopeType ###################

4.2.9.

Some Seam component classes can fulfill more than one role in the system. For example, we often have a `User` class which is usually used as a session-scoped component representing the current user but is used in user administration screens as a conversation-scoped component. The

`@Role` annotation lets us define an additional named role for a component, with a different scope — it lets us bind the same component class to different context variables. (Any Seam component *instance* may be bound to multiple context variables, but this lets us do it at the class level, and take advantage of auto-instantiation.)

```
@Name("user")
@Entity
@Scope(CONVERSATION)
@Role(name="currentUser", scope=SESSION)
public class User {
   ...
}
```

`@Roles` ###################################

```
@Name("user")
@Entity
@Scope(CONVERSATION)
@Roles({@Role(name="currentUser", scope=SESSION),
      @Role(name="tempUser", scope=EVENT)})
public class User {
   ...
}
```

4.2.10.

################## Seam ############################ (Eat Your Own Dog Food) # ##### Seam ###### (##) # Seam###################### ## ################################### Seam ######################### ############# Seam ##### `org.jboss.seam.core` # ##### Java ################

##############Seam ############################## ######### `instance()` ############

```
FacesMessages.instance().add("Welcome back, #{user.name}!");
```

4.3.

(dependency injection) #### ##### *(inversion of control)* ########## Java ############### ### ############ setter

################################## (################ ######################
############################## ################################
################################
##
###Seam############################## ################################## Seam
########################## *(bijection)* ##########
####################################

- ######## *(contextual)* - ##
  (##### (wider) ######################### (narrow) ########################)
- #### *(bidirectional)* - ####################################
  ############################## *(outject)* ########
  ##
- ## *(dynamic)* - ############################## ###################### Seam
  ####################

##
##
##

`@In` ######################################

```
@Name("loginAction")
@Stateless
public class LoginAction implements Login {
   @In User user;
   ...
}
```

#####setter ################

```
@Name("loginAction")
@Stateless
public class LoginAction implements Login {
   User user;

   @In
   public void setUser(User user) {
      this.user=user;
   }

   ...
```

```
}
```

######## Seam ################################### ###################### ####
@In("currentUser")###################################

Seam ########################## @In(create=true) ############ ######## (null ####) #####@In(required=false)

############### ###################### @In(create=true) ################## ######## ######## @AutoCreate ################## create=true ############ ####################

####################

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

Injected values are disinjected (i.e., set to `null`) immediately after method completion and outjection.

(##)

@Out##

```
@Name("loginAction")
@Stateless
public class LoginAction implements Login {
   @Out User user;
   ...
}
```

getter

```
@Name("loginAction")
@Stateless
public class LoginAction implements Login {
   User user;
```

```
    @Out
    public User getUser() {
        return user;
    }

    ...
}
```

##################################

```
@Name("loginAction")
@Stateless
public class LoginAction implements Login {
    @In @Out User user;
    ...
}
```

####

```
@Name("loginAction")
@Stateless
public class LoginAction implements Login {
    User user;

    @In
    public void setUser(User user) {
        this.user=user;
    }

    @Out
    public User getUser() {
        return user;
    }

    ...
}
```

4.4.

Bean ####### Bean Seam ########### EJB 3.0
(`@PostConstruct#@PreDestroy` ##) ############### ####Seam # JavaBean #######

########################## ################ J2EE ##############Seam # @PostConstruct # @PreDestroy #### 2 ##################################

@Create ##### Seam ########################## ######## 1 ## @Create#############

@Destroy ##### Seam ################################## ######## 1 ## @Destroy ##############

################ Bean ################@Remove######### ######################## Seam #########

################ @Startup ########## ###################################### @Startup ######### ### Seam ############################ @Startup(depends={....}) ######### ###################################

4.5.

@Install ######### ## ###############

- ##################################
- ################################
- ################################## (##############)#

@Install # #### # ### ###############

############### ################################## ######################## Seam ########## Seam ######################## ###################### (##)#

1. BUILT_IN — Seam ##########################
2. FRAMEWORK — ##
3. APPLICATION — ########## ################################
4. DEPLOYMENT — ##############################
5. MOCK — ######################

JMS ######## messageSender ######################

```
@Name("messageSender")
public class MessageSender {
   public void sendMessage() {
      //do something with JMS
   }
```

```
}
```

########## ###JMS ################################
mock

```
@Name("messageSender")
@Install(precedence=MOCK)
public class MockMessageSender extends MessageSender {
    public void sendMessage() {
        //do nothing!
    }
}
```

Seam

############################## ##############
################################### ### Jar ################################
##
################################### `@Install` ##################### Seam
###
################################

4.6.

################################

```
private static final Log log = LogFactory.getLog(CreateOrderAction.class);

public Order createOrder(User user, Product product, int quantity) {
    if ( log.isDebugEnabled() ) {
        log.debug("Creating new order for user: " + user.username() +
            " product: " + product.name()
            + " quantity: " + quantity);
    }
    return new Order(user, product, quantity);
}
```

##
Java ####### 10
################################

Seam ################## API #######

```
@Logger private Log log;

public Order createOrder(User user, Product product, int quantity) {
    log.debug("Creating new order for user: #0 product: #1 quantity: #2", user.username(),
 product.name(), quantity);
   return new Order(user, product, quantity);
}
```

It doesn't matter if you declare the `log` variable static or not — it will work either way, except for entity bean components which require the `log` variable to be static.

`#########debug() ##### ## ####### ###### if ( log.isDebugEnabled() ) ######################## Seam ########### Log ################### ###################################`

`User` # `Product` ## ############# Seam ########################

```
@Logger private Log log;

public Order createOrder(User user, Product product, int quantity) {
   log.debug("Creating new order for user: #{user.username} product: #{product.name} quantity:
 #0", quantity);
   return new Order(user, product, quantity);
}
```

Seam ######### log4j #### JDK logging ################ log4j ##############Seam ########## ########Seam # JDK logging #######

4.7. `Mutable` ######### `@ReadOnly`

`############# setAttribute() ########## ###################################### ############################## HttpSession ################## ############################## ############################## ############################################################ ##############`

`#####EJB ########## Bean ################################### ##### EJB ################################ ######### Seam ##### EJB 3.0 ################################ ################ JavaBean ####### Bean ############ Seam # Web ##############################################`

`############# JavaBean ############Seam ## ################################ setAttribute() ################### ######################################## ######## org.jboss.seam.core.Mutable ##############`

org.jboss.seam.core.AbstractMutable #######
##########

```
@Name("account")
public class Account extends AbstractMutable
{
   private BigDecimal balance;

   public void setBalance(BigDecimal balance)
   {
      setDirty(this.balance, balance);
      this.balance = balance;
   }

   public BigDecimal getBalance()
   {
      return balance;
   }

   ...

}
```

################ @ReadOnly ################

```
@Name("account")
public class Account
{
   private BigDecimal balance;

   public void setBalance(BigDecimal balance)
   {
      this.balance = balance;
   }

   @ReadOnly
   public BigDecimal getBalance()
   {
      return balance;
   }

   ...
```

```
}
```

################## Bean ########## Seam # (#######) ########## *Seam* ## #### `setAttribute()` ########################## ###################### ################## Bean ############## ###### Bean ################# ########## Bean # JavaBean ################# ##########

```
@Stateful
@Name("account")
public class AccountManager extends AbstractMutable
{
   private Account account; // an entity bean

   @Unwrap
   public Account getAccount()
   {
      return account;
   }

   ...

}
```

Seam ################## `EntityHome` #### Seam ##################### Bean ######################################

4.8.

Seam ################################## ####### `@In` ##### Seam ################ ## ####### Seam ############################## (### `@Destroy`)# ####Seam ####### Seam ############################ Seam ################################## 2#3 ############

################ ## Seam ## ######### ## ########################## ################## `@Factory` ############################## ########################## ##################### ######################## #########Seam ########################

```
@Factory(scope=CONVERSATION)
```

```
public List<Customer
> getCustomerList() {
   return ... ;
}
```

########## ###################### `void` ###########

```
@DataModel List<Customer
> customerList;

@Factory("customerList")
public void initCustomerList() {
   customerList = ...  ;
}
```

######## `customerList` ################ null #### ########################## ################ ########## ############ ### ##### ############### Seam ########## ######################## ###################

########## `@Unwrap` ##################### ########################## ## ########################

```
@Name("customerList")
@Scope(CONVERSATION)
public class CustomerListManager
{
   ...

   @Unwrap
   public List<Customer
> getCustomerList() {
      return ... ;
   }
}
```

`@Unwrap` ############### `@Destroy`

```
@Name("hens")
@Scope(APPLICATION)
```

```
public class HenHouse
{
   Set<Hen> hens;

   @In(required=false) Hen hen;

   @Unwrap
   public List<Hen> getHens()
   {
      if (hens == null)
      {
         // Setup our hens
      }
      return hens;
   }

   @Observer({"chickBorn", "chickenBoughtAtMarket"})
   public addHen()
   {
      hens.add(hen);
   }

   @Observer("chickenSoldAtMarket")
   public removeHen()
   {
      hens.remove(hen);
   }

   @Observer("foxGetsIn")
   public removeAllHens()
   {
      hens.clear();
   }
   ...
}
```

##
##

Seam##########

XML#################Seam####################XML####Seam#####################
Java######################## ######################## Seam##################
Seam############################ ############ `web.xml` ################
`components.xml` ########

5.1.

Seam components may be provided with configuration properties either via servlet context parameters, via system properties, or via a properties file named `seam.properties` in the root of the classpath.

The configurable Seam component must expose JavaBeans-style property setter methods for the configurable attributes. If a Seam component named `com.jboss.myapp.settings` has a setter method named `setLocale()`, we can provide a property named `com.jboss.myapp.settings.locale` in the `seam.properties` file, a system property named `org.jboss.seam.properties.com.jboss.myapp.settings.locale` via -D at startup, or as a servlet context parameter, and Seam will set the value of the `locale` attribute whenever it instantiates the component.

The same mechanism is used to configure Seam itself. For example, to set the conversation timeout, we provide a value for `org.jboss.seam.core.manager.conversationTimeout` in `web.xml`, `seam.properties`, or via a system property prefixed with `org.jboss.seam.properties`. (There is a built-in Seam component named `org.jboss.seam.core.manager` with a setter method named `setConversationTimeout()`.)

5.2. `components.xml`##############

`components.xml`###################### ############

- Configure components that have been installed automatically — including both built-in components, and application components that have been annotated with the `@Name` annotation and picked up by Seam's deployment scanner.
- Install classes with no `@Name` annotation as Seam components — this is most useful for certain kinds of infrastructural components which can be installed multiple times with different names (for example Seam-managed persistence contexts).
- `@Name`##`@Install`#####################################
- ################ (override) #####

`components.xml`#########################

- `war#WEB-INF######`
- `jar#META-INF######`

- @Name####################jar

###Seam#####################seam.properties #####META-INF/components.xml########## @Name################################## (####@Install ######################### ########) # components.xml######### ##

components.xml #####jBPM###########

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:bpm="http://jboss.com/products/seam/bpm">
   <bpm:jbpm/>
</components
>
```

###############

```
<components>
   <component class="org.jboss.seam.bpm.Jbpm"/>
</components
>
```

###2######Seam##############################

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:persistence="http://jboss.com/products/seam/persistence"

   <persistence:managed-persistence-context name="customerDatabase"
                persistence-unit-jndi-name="java:/customerEntityManagerFactory"/>

   <persistence:managed-persistence-context name="accountingDatabase"
                persistence-unit-jndi-name="java:/accountingEntityManagerFactory"/>

</components
>
```

###########

```
<components>
   <component name="customerDatabase"
```

```
            class="org.jboss.seam.persistence.ManagedPersistenceContext">
        <property name="persistenceUnitJndiName"
>java:/customerEntityManagerFactory</property>
    </component>

    <component name="accountingDatabase"
            class="org.jboss.seam.persistence.ManagedPersistenceContext">
        <property name="persistenceUnitJndiName"
>java:/accountingEntityManagerFactory</property>
    </component>
</components
>
```

###############Seam################### ######################

```
<components xmlns="http://jboss.com/products/seam/components"
          xmlns:persistence="http://jboss.com/products/seam/persistence"

  <persistence:managed-persistence-context name="productDatabase"
                                    scope="session"
                  persistence-unit-jndi-name="java:/productEntityManagerFactory"/>

</components
>
```

```
<components>

    <component name="productDatabase"
            scope="session"
            class="org.jboss.seam.persistence.ManagedPersistenceContext">
        <property name="persistenceUnitJndiName"
>java:/productEntityManagerFactory</property>
    </component>

</components
>
```

############################auto-create ##############################@In
################create=true########## ########

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:persistence="http://jboss.com/products/seam/persistence"

 <persistence:managed-persistence-context name="productDatabase"
                            auto-create="true"
                persistence-unit-jndi-name="java:/productEntityManagerFactory"/>

</components
>
```

```
<components>

   <component name="productDatabase"
      auto-create="true"
          class="org.jboss.seam.persistence.ManagedPersistenceContext">
      <property name="persistenceUnitJndiName"
>java:/productEntityManagerFactory</property>
   </component>

</components
>
```

`<factory>`############################ ######################################

```
<components>

                    <factory      name="contact"      method="#{contactManager.loadContact}"
 scope="CONVERSATION"/>

</components
>
```

Seam################ (##) ########

```
<components>

   <factory name="user" value="#{actor}" scope="STATELESS"/>

</components
```

```
>
```

##################################

```
<components>

   <factory name="contact" value="#{contactManager.contact}" scope="STATELESS"/>

</components
>
```

<factory>###auto-create="true"###### ###############

```
<components>

    <factory name="session" value="#{entityManager.delegate}" scope="STATELESS" auto-
create="true"/>

</components
>
```

############### components.xml ##
Seam# components.xml ######@wildcard@ ######################Ant########
(#####) ####### components.properties ############ (###)
############################Seam####################

5.3.

###XML########################components.xml
################################### Seam###### com.helloworld.Hello#### com/
helloworld/Hello.component.xml#####################
(################################## Hibernate##################)
#############<components> ### <component>#############

##################################

```
<components>
   <component class="com.helloworld.Hello" name="hello">
      <property name="name"
>#{user.name}</property>
   </component>
```

```
    <factory name="message" value="#{hello.message}"/>
</components
>
```

################################### ##########

```
<component name="hello">
    <property name="name"
>#{user.name}</property>
</component
>
```

##

#####com/helloworld/components.xml# com.helloworld############################

5.4.

##

org.jboss.seam.core.manager.conversationTimeout 60000

```
<core:manager conversation-timeout="60000"/>
```

```
<component name="org.jboss.seam.core.manager">
    <property name="conversationTimeout"
>60000</property>
</component
>
```

##

org.jboss.seam.bpm.jbpm.processDefinitions order.jpdl.xml, return.jpdl.xml, inventory.jpdl.xml

```
<bpm:jbpm>
    <bpm:process-definitions>
        <value
```

```
>order.jpdl.xml</value>
      <value
>return.jpdl.xml</value>
      <value
>inventory.jpdl.xml</value>
   </bpm:process-definitions>
</bpm:jbpm
>
```

```
<component name="org.jboss.seam.bpm.jbpm">
   <property name="processDefinitions">
      <value
>order.jpdl.xml</value>
      <value
>return.jpdl.xml</value>
      <value
>inventory.jpdl.xml</value>
   </property>
</component
>
```

##

```
<component name="issueEditor">
   <property name="issueStatuses">
      <key
>open</key
> <value
>open issue</value>
      <key
>resolved</key
> <value
>issue resolved by developer</value>
      <key
>closed</key
> <value
>resolution accepted by user</value>
   </property>
</component
>
```

When configuring multi-valued properties, by default, Seam will preserve the order in which you place the attributes in `components.xml` (unless you use a `SortedSet`/`SortedMap` then Seam will use `TreeMap`/`TreeSet`). If the property has a concrete type (for example `LinkedList`) Seam will use that type.

##################################

```
<component name="issueEditor">
  <property name="issueStatusOptions" type="java.util.LinkedHashMap">
    <key
>open</key
> <value
>open issue</value>
    <key
>resolved</key
> <value
>issue resolved by developer</value>
    <key
>closed</key
> <value
>resolution accepted by user</value>
  </property>
</component
>
```

############## (value-binding expression) ##########################
###@In############################ ##
######JSF#Spring#######IoC#############################

```
<drools:managed-working-memory name="policyPricingWorkingMemory"
   rule-base="#{policyPricingRules}"/>
```

```
<component name="policyPricingWorkingMemory"
   class="org.jboss.seam.drools.ManagedWorkingMemory">
   <property name="ruleBase"
>#{policyPricingRules}</property>
</component
>
```

Seam#########Bean#################EL###

```
<component name="greeter" class="com.example.action.Greeter">
   <property name="message"
>Nice to see you, #{identity.username}!</property>
</component
>
```

##Seam#ValueExpression ### MethodExpression########EL##Seam#############

```
<framework:entity-home name="myEntityHome"
   class="com.example.action.MyEntityHome" entity-class="com.example.model.MyEntity"
   created-message="'#{myEntityHome.instance.name}' has been successfully added."/>
```

############ ValueExpression ### MethodExpression ## getExpressionString() ###################################### ValueExpression########## getValue() ########################MethodExpression###### invoke(Object args...)########################MethodExpression##### ########################EL############

5.5. XML#######

######################XML###components.xml#####

```
<?xml version="1.0" encoding="UTF-8"?>
<components xmlns="http://jboss.com/products/seam/components"
            xsi:schemaLocation="http://jboss.com/products/seam/components http://jboss.com/products/seam/components-2.2.xsd">

   <component class="org.jboss.seam.core.init">
      <property name="debug">true</property>
      <property name="jndiPattern">@jndiPattern@</property>
   </component>

</components>
```

################ ##

########################

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:core="http://jboss.com/products/seam/core"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation=
          "http://jboss.com/products/seam/core http://jboss.com/products/seam/core-2.2.xsd
                http://jboss.com/products/seam/components  http://jboss.com/products/seam/
components-2.2.xsd">

   <core:init debug="true" jndi-pattern="@jndiPattern@"/>

</components>
```

####################XML##################
XML##############################
#################components.xml#######################

##########Seam###
####Seam################### ######################<component> ###
################################
Seam##

###Java########@Namespace#########################XML#################
(################################package-info.java##################)
###seapay#########

```
@Namespace(value="http://jboss.com/products/seam/examples/seampay")
package org.jboss.seam.example.seampay;

import org.jboss.seam.annotations.Namespace;
```

###############components.xml################## ######################

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:pay="http://jboss.com/products/seam/examples/seampay"
        ... >

   <pay:payment-home new-instance="#{newPayment}"
            created-message="Created a new payment to #{newPayment.payee}" />

   <pay:payment name="newPayment"
           payee="Somebody"
           account="#{selectedAccount}"
           payment-date="#{currentDatetime}"
```

```
            created-date="#{currentDatetime}" />
    ...
</components
>
```

####

```
<components xmlns="http://jboss.com/products/seam/components"
          xmlns:pay="http://jboss.com/products/seam/examples/seampay"
          ... >

   <pay:payment-home>
      <pay:new-instance
>"#{newPayment}"</pay:new-instance>
      <pay:created-message
>Created a new payment to #{newPayment.payee}</pay:created-message>
   </pay:payment-home>

   <pay:payment name="newPayment">
      <pay:payee
>Somebody"</pay:payee>
      <pay:account
>#{selectedAccount}</pay:account>
      <pay:payment-date
>#{currentDatetime}</pay:payment-date>
      <pay:created-date
>#{currentDatetime}</pay:created-date>
    </pay:payment>
    ...
</components
>
```

################################## #######<pay:payment-home># paymentHome################

```
package org.jboss.seam.example.seampay;
...
@Name("paymentHome")
public class PaymentController
   extends EntityHome<Payment>
{
   ...
```

```
}
```

################################## ##################################

#########<pay:payment>###org.jboss.seam.example.seampay#######Payment##########
Payment #################Seam#######################

```
package org.jboss.seam.example.seampay;
...
@Entity
public class Payment
   implements Serializable
{
   ...
}
```

##
#############Seam###Seam#######

##Seam################

- components — `http://jboss.com/products/seam/components`
- core — `http://jboss.com/products/seam/core`
- drools — `http://jboss.com/products/seam/drools`
- framework — `http://jboss.com/products/seam/framework`
- jms — `http://jboss.com/products/seam/jms`
- remoting — `http://jboss.com/products/seam/remoting`
- theme — `http://jboss.com/products/seam/theme`
- security — `http://jboss.com/products/seam/security`
- mail — `http://jboss.com/products/seam/mail`
- web — `http://jboss.com/products/seam/web`
- pdf — `http://jboss.com/products/seam/pdf`
- spring — `http://jboss.com/products/seam/spring`

################

##############################Seam##
############JSF#################method binding expression)
#################################### ####################################
(cross-cutting concerns) ##

6.1. Seam####

Seam########################## ########################## (fine-grained eventing model) ########################## Seam#################################

- JSF####
- jBPM########
- Seam########
- Seam##############
- Seam#############

###############JSF EL################Seam################JSF######JSF##################

```
<h:commandButton value="Click me!" action="#{helloWorld.sayHello}"/>
```

jBPM########jBPM##########################

```
<start-page name="hello" view-id="/hello.jsp">
   <transition to="hello">
      <action expression="#{helloWorld.sayHello}"/>
   </transition>
</start-page
>
```

JSF ##### jPBM ################################ #### Seam ##############################

6.2.

Seam################################## #########WEB-INF/pages.xml####### ###JSF###id#########################

```
<pages>
   <page view-id="/hello.jsp" action="#{helloWorld.sayHello}"/>
</pages
>
```

* ############################ ID##############

```
<pages>
   <page view-id="/hello/*" action="#{helloWorld.sayHello}"/>
</pages
>
```

Keep in mind that if the <page> element is defined in a fine-grained page descriptor, the view-id attribute can be left off since it is implied.

################################id######## Seam################# (least-specific to most-specific) ## ####################

###############JSF outcome##################outcome #null######Seam#####################################

Furthermore, the view id mentioned in the <page> element need not correspond to a real JSP or Facelets page! So, we can reproduce the functionality of a traditional action-oriented framework like Struts or WebWork using page actions. This is quite useful if you want to do complex things in response to non-faces requests (for example, HTTP GET requests).

####################<action>##############

```
<pages>
   <page view-id="/hello.jsp">
      <action execute="#{helloWorld.sayHello}" if="#{not validation.failed}"/>
      <action execute="#{hitCount.increment}"/>
   </page>
</pages
>
```

Page actions are executed on both an initial (non-faces) request and a postback (faces) request. If you are using the page action to load data, this operation may conflict with the standard JSF action(s) being executed on a postback. One way to disable the page action is to setup a condition that resolves to true only on an initial request.

```
<pages>
   <page view-id="/dashboard.xhtml">
      <action execute="#{dashboard.loadData}"
         if="#{not facesContext.renderKit.responseStateManager.isPostback(facesContext)}"/>
   </page>
</pages>
```

This condition consults the `ResponseStateManager#isPostback(FacesContext)` to determine if the request is a postback. The ResponseStateManager is accessed using `FacesContext.getCurrentInstance().getRenderKit().getResponseStateManager()`.

To save you from the verbosity of JSF's API, Seam offers a built-in condition that allows you to accomplish the same result with a heck of a lot less typing. You can disable a page action on postback by simply setting the `on-postback` to `false`:

```
<pages>
   <page view-id="/dashboard.xhtml">
      <action execute="#{dashboard.loadData}" on-postback="false"/>
   </page>
</pages>
```

For backwards compatibility reasons, the default value of the `on-postback` attribute is true, though likely you will end up using the opposite setting more often.

6.3.

JSF faces ## (######) ######## (############) ######## (############) ############## ############################

GET ################ ################################## (JSF ##############)#

##

6.3.1.

Seam##

```
<pages>
    <page view-id="/hello.jsp" action="#{helloWorld.sayHello}">
       <param name="firstName" value="#{person.firstName}"/>
       <param name="lastName" value="#{person.lastName}"/>
    </page>
 </pages
```

```
>
```

<param>###########JSF################

- ########id####non-faces (GET) ########## Seam##
- ### <s:link> # <s:button> ################## ######## ############# (<s:link> ############) ############################
- ###id####<redirect/ >###
- ###########id###########JSF############# ############faces######PAGE##############################

########################## /hello.jsp ## (### /hello.jsp ## /hello.jsp ######) ############# ######################### (############) ########################

6.4.

name ##################### PAGE

```
<pages>
    <page view-id="/hello.jsp" action="#{helloWorld.sayHello}">
      <param name="firstName" />
      <param name="lastName" />
    </page>
 </pages
>
```

##CRUD###

- ### <s:link> # <s:button> ##
- ###########id###########JSF############# ############faces######PAGE#########################

###
faces###

6.5. ###########URL####

#####pages.xml##############################Seam#URL##############################URL#######

```
<page view-id="/home.xhtml">
   <rewrite pattern="/home" />
</page>
```

/home ############ /home.xhtml ###################### /home.seam ############ /home ################################URL################# / home.seam?conversationId=13 # /home.seam?color=red

##

```
<page view-id="/home.xhtml">
   <rewrite pattern="/home/{color}" />
   <rewrite pattern="/home" />
</page>
```

/home/red ########## /home.seam?color=red ##################color ########### /home.seam?color=blue ##########URL###### /home/blue

Seam###################################Seam##########URL####################/ search.seam?conversationId=13#/search-13##########

```
<page view-id="/search.xhtml">
   <rewrite pattern="/search-{conversationId}" />
   <rewrite pattern="/search" />
</page>
```

Seam URL################################Seam################################Seam############## org.tuckey URLRewriteFilter ########Web#######################

URL#####Seam##*#30.1.4.3. #URL* ##############

6.6.

################JSF#################

```
<pages>
```

```
  <page view-id="/calculator.jsp" action="#{calculator.calculate}">
    <param name="x" value="#{calculator.lhs}"/>
    <param name="y" value="#{calculator.rhs}"/>
                    <param    name="op"    converterId="com.my.calculator.OperatorConverter"
 value="#{calculator.op}"/>
  </page>
</pages
>
```

########################

```
<pages>
  <page view-id="/calculator.jsp" action="#{calculator.calculate}">
    <param name="x" value="#{calculator.lhs}"/>
    <param name="y" value="#{calculator.rhs}"/>
    <param name="op" converter="#{operatorConverter}" value="#{calculator.op}"/>
  </page>
</pages
>
```

######JSF###### `required="true"` ########

```
<pages>
   <page view-id="/blog.xhtml">
      <param name="date"
           value="#{blog.date}"
           validatorId="com.my.blog.PastDate"
           required="true"/>
   </page>
</pages
>
```

########################

```
<pages>
   <page view-id="/blog.xhtml">
      <param name="date"
           value="#{blog.date}"
           validator="#{pastDateValidator}"
           required="true"/>
```

```
  </page>
</pages
>
```

Even better, model-based Hibernate validator annotations are automatically recognized and validated. Seam also provides a default date converter to convert a string parameter value to a date and back.

########################`FacesMessage`#`FacesContext`########

6.7.

Seam##########`faces-config.xml`#########JSF#####################JSF#####################

- ###############################
- ###### (conversation) ##############
- ############################### ########EL################

`###pages.xml # faces-config.xml` ##`pages.xml` ###############

##JSF############

```
<navigation-rule>
  <from-view-id
>/editDocument.xhtml</from-view-id>

  <navigation-case>
    <from-action
>#{documentEditor.update}</from-action>
    <from-outcome
>success</from-outcome>
    <to-view-id
>/viewDocument.xhtml</to-view-id>
    <redirect/>
  </navigation-case>

</navigation-rule
>
```

#################

```
<page view-id="/editDocument.xhtml">

   <navigation from-action="#{documentEditor.update}">
      <rule if-outcome="success">
         <redirect view-id="/viewDocument.xhtml"/>
      </rule>
   </navigation>

</page
>
```

`####DocumentEditor`
##################JSF###Seam##################

```
<page view-id="/editDocument.xhtml">

   <navigation from-action="#{documentEditor.update}"
               evaluate="#{documentEditor.errors.size}">
      <rule if-outcome="0">
         <redirect view-id="/viewDocument.xhtml"/>
      </rule>
   </navigation>

</page
>
```

######################

```
<page view-id="/editDocument.xhtml">

   <navigation from-action="#{documentEditor.update}">
      <rule if="#{documentEditor.errors.empty}">
         <redirect view-id="/viewDocument.xhtml"/>
      </rule>
   </navigation>

</page
>
```

########################coutcome######################
##########outcome############################

#################### (conversation) ############## ##############

```
<page view-id="/editDocument.xhtml">

   <navigation from-action="#{documentEditor.update}">
      <rule if="#{documentEditor.errors.empty}">
         <end-conversation/>
         <redirect view-id="/viewDocument.xhtml"/>
      </rule>
   </navigation>

</page
>
```

###ID#########################

```
<page view-id="/editDocument.xhtml">

   <navigation from-action="#{documentEditor.update}">
      <rule if="#{documentEditor.errors.empty}">
         <end-conversation/>
         <redirect view-id="/viewDocument.xhtml">
            <param name="documentId" value="#{documentEditor.documentId}"/>
         </redirect>
      </rule>
   </navigation>

</page
>
```

outcome#null#####JSF###########coucome#null###############
##########################null####outcome########outcome#null############

```
<page view-id="/editDocument.xhtml">

   <navigation from-action="#{documentEditor.update}">
      <rule>
         <render view-id="/viewDocument.xhtml"/>
```

```
        </rule>
    </navigation>

</page
>
```

outcome#null###################### ###############

```
<page view-id="/editDocument.xhtml">

    <navigation from-action="#{documentEditor.update}">
        <render view-id="/viewDocument.xhtml"/>
    </navigation>

</page
>
```

##

In case you are using JSF RI 2, you have to define navigation rule for each of the possible non-null outcome values from a page action, or else implicit navigation is going to render. It is annoying, hopefully will be fixed in the next maintenance version release of JSF 2.

###id#JSF EL################

```
<page view-id="/editDocument.xhtml">

    <navigation>
        <rule if-outcome="success">
            <redirect view-id="/#{userAgent}/displayDocument.xhtml"/>
        </rule>
    </navigation>

</page
>
```

6.8.

################################## ################################ ###id#/calc/calculator.jsp############ calc/calculator.page.xml ################## ###########<page>#######id##########

```
<page action="#{calculator.calculate}">
   <param name="x" value="#{calculator.lhs}"/>
   <param name="y" value="#{calculator.rhs}"/>
   <param name="op" converter="#{operatorConverter}" value="#{calculator.op}"/>
</page
>
```

6.9.

Seam################################## ##############observer/observable##################
##Seam####################

######## (observers) #components.xml#######

```
<components>
   <event type="hello">
      <action execute="#{helloListener.sayHelloBack}"/>
      <action execute="#{logger.logHello}"/>
   </event>
</components
>
```

###event type #############

################################components.xml
##Seam#########################

```
@Name("helloWorld")
public class HelloWorld {
   public void sayHello() {
      FacesMessages.instance().add("Hello World!");
      Events.instance().raiseEvent("hello");
   }
}
```

########################

```
@Name("helloWorld")
public class HelloWorld {
   @RaiseEvent("hello")
   public void sayHello() {
      FacesMessages.instance().add("Hello World!");
   }
}
```

##
##

```
@Name("helloListener")
public class HelloListener {
   public void sayHelloBack() {
      FacesMessages.instance().add("Hello to you too!");
   }
}
```

components.xml ## components.xml
##

```
@Name("helloListener")
public class HelloListener {
   @Observer("hello")
   public void sayHelloBack() {
      FacesMessages.instance().add("Hello to you too!");
   }
}
```

##Seam###

```
@Name("helloWorld")
public class HelloWorld {
   private String name;
   public void sayHello() {
      FacesMessages.instance().add("Hello World, my name is #0.", name);
      Events.instance().raiseEvent("hello", name);
   }
```

```
}
```

```
@Name("helloListener")
public class HelloListener {
   @Observer("hello")
   public void sayHelloBack(String name) {
      FacesMessages.instance().add("Hello #0!", name);
   }
}
```

6.10.

Seam###

- `org.jboss.seam.validationFailed` — JSF####################
- `org.jboss.seam.noConversation` — ################################
- `org.jboss.seam.preSetVariable.<name>` — ######## <name> ##############
- `org.jboss.seam.postSetVariable.<name>` — ######## <name> ###############
- `org.jboss.seam.preRemoveVariable.<name>` — ######## <name> #################
- `org.jboss.seam.postRemoveVariable.<name>` — ######## <name> #################
- `org.jboss.seam.preDestroyContext.<SCOPE>` — <SCOPE> ###################
- `org.jboss.seam.postDestroyContext.<SCOPE>` — <SCOPE> ###################
- `org.jboss.seam.beginConversation` — called whenever a long-running conversation begins
- `org.jboss.seam.endConversation` — called whenever a long-running conversation ends
- `org.jboss.seam.conversationTimeout` — called when a conversation timeout occurs. The conversation id is passed as a parameter.
- `org.jboss.seam.beginPageflow` — called when a pageflow begins
- `org.jboss.seam.beginPageflow.<name>` — called when the pageflow <name> begins
- `org.jboss.seam.endPageflow` — called when a pageflow ends
- `org.jboss.seam.endPageflow.<name>` — called when the pageflow <name> ends

- `org.jboss.seam.createProcess.<name>` — called when the process <name> is created
- `org.jboss.seam.endProcess.<name>` — called when the process <name> ends
- `org.jboss.seam.initProcess.<name>` — called when the process <name> is associated with the conversation
- `org.jboss.seam.initTask.<name>` — called when the task <name> is associated with the conversation
- `org.jboss.seam.startTask.<name>` — called when the task <name> is started
- `org.jboss.seam.endTask.<name>` — called when the task <name> is ended
- `org.jboss.seam.postCreate.<name>` — called when the component <name> is created
- `org.jboss.seam.preDestroy.<name>` — called when the component <name> is destroyed
- `org.jboss.seam.beforePhase` — called before the start of a JSF phase
- `org.jboss.seam.afterPhase` — called after the end of a JSF phase
- `org.jboss.seam.postInitialization` — called when Seam has initialized and started up all components
- `org.jboss.seam.postReInitialization` — called when Seam has re-initialized and started up all components after a redeploy
- `org.jboss.seam.exceptionHandled.<type>` — ##############Seam##################
- `org.jboss.seam.exceptionHandled` — ##############Seam##################
- `org.jboss.seam.exceptionNotHandled` — ####################################
- `org.jboss.seam.afterTransactionSuccess` — Seam####################################
- `org.jboss.seam.afterTransactionSuccess.<name>` — `<name>`##################Seam##
- `org.jboss.seam.security.loggedOut` — called when a user logs out
- `org.jboss.seam.security.loginFailed` — called when a user authentication attempt fails
- `org.jboss.seam.security.loginSuccessful` — called when a user is successfully authenticated
- `org.jboss.seam.security.notAuthorized` — called when an authorization check fails
- `org.jboss.seam.security.notLoggedIn` — called there is no authenticated user and authentication is required

- `org.jboss.seam.security.postAuthenticate.` — called after a user is authenticated
- `org.jboss.seam.security.preAuthenticate` — called before attempting to authenticate a user

Seam############################## (observe) ######### ########################

6.11. Seam######

EJB 3.0 ###### Bean ################################ Bean ############### `@AroundInvoke` ################################ ## Bean ################### `@Interceptors` ####################

```
public class LoggedInInterceptor {

  @AroundInvoke
  public Object checkLoggedIn(InvocationContext invocation) throws Exception {

    boolean isLoggedIn = Contexts.getSessionContext().get("loggedIn")!=null;
    if (isLoggedIn) {
      //the user is already logged in
      return invocation.proceed();
    }
    else {
      //the user is not logged in, fwd to login page
      return "login";
    }
  }

}
```

##############################Bean####################Bean# `@Interceptors(LoggedInInterceptor.class)` ##Seam############################## `@Interceptors` #######EJB3################################## `@LoggedIn` ##############

```
@Target(TYPE)
@Retention(RUNTIME)
@Interceptors(LoggedInInterceptor.class)
public @interface LoggedIn {}
```

Bean #`@LoggedIn`

```
@Stateless
@Name("changePasswordAction")
@LoggedIn
@Interceptors(SeamInterceptor.class)
public class ChangePasswordAction implements ChangePassword {

   ...

   public String changePassword() { ... }

}
```

################ (########)# ############# @Interceptor
##################################

```
@Interceptor(around={BijectionInterceptor.class,
                    ValidationInterceptor.class,
                    ConversationInterceptor.class},
             within=RemoveInterceptor.class)
public class LoggedInInterceptor
{
   ...
}
```

########################## EJB3 ##########################

```
@Interceptor(type=CLIENT)
public class LoggedInInterceptor
{
   ...
}
```

EJB ################ ######################################
####################### Seam ## @Interceptor(stateless=true)
################################

Seam################################Seam##

Seam ####### EJB3 Bean ##### JavaBean ##################

EJB ## ######### (@AroundInvoke ####) ############## ###########
@PostConstruct# @PreDestroy# @PrePassivate ### @PostActive ###########

Seam ## ###################################### EJB3 Bean ##### JavaBean ################### (JavaBean ################ `@PreDestroy` #####)#

6.12.

JSF ###################### ################ Seam ################## XML ## ###### ## EJB 3.0 ### `@ApplicationException` ############################

6.12.1.

Bean ######################## ## EJB ######### ###### ###################### ########## ###################### `@ApplicationException(rollback=true)` ################## (############ ########## ### `@ApplicationException` ###################### ######### `@ApplicationException` ########################)

########################## ###################### ## ##

Seam # EJB 3.0 ########## Seam JavaBean #################

Seam ################## ### ############## Seam ################ ### JSF ################## ###################################### ###### Seam ################## Seam

6.12.2. Seam

Seam##############################`web.xml`####################

```
<filter>
   <filter-name
>Seam Filter</filter-name>
   <filter-class
>org.jboss.seam.servlet.SeamFilter</filter-class>
</filter>

<filter-mapping>
   <filter-name
>Seam Filter</filter-name>
   <url-pattern
>*.seam</url-pattern>
</filter-mapping
```

```
>
```

################ web.xml # Facelets ######## components.xml # Seam ######################

6.12.3.

Seam ################## HTTP 404 ######### ################################## ## Seam

```
@HttpError(errorCode=404)
public class ApplicationException extends Exception { ... }
```

Seam ################################### ### ############

```
@Redirect(viewId="/failure.xhtml", end=true)
@ApplicationException(rollback=true)
public class UnrecoverableApplicationException extends RuntimeException { ... }
```

##
Seam#JSF###

EL############ ###id #############

Seam ## ###

```
@Redirect(viewId="/error.xhtml", message="Unexpected error")
public class SystemException extends RuntimeException { ... }
```

6.12.4. ##### XML

#####################################Seam###### pages.xml ################

```
<pages>
```

```
  <exception class="javax.persistence.EntityNotFoundException">
    <http-error error-code="404"/>
  </exception>

  <exception class="javax.persistence.PersistenceException">
    <end-conversation/>
    <redirect view-id="/error.xhtml">
      <message
>Database access failed</message>
    </redirect>
  </exception>

  <exception>
    <end-conversation/>
    <redirect view-id="/error.xhtml">
      <message
>Unexpected failure</message>
    </redirect>
  </exception>

</pages
>
```

`<exception>` ########## ########## `pages.xml`

EL########## `view-id` ##########

EL ##########
Seam##########

```
...
throw new AuthorizationException("You are not allowed to do this!");

<pages>

   <exception class="org.jboss.seam.security.AuthorizationException">
      <end-conversation/>
      <redirect view-id="/error.xhtml">
        <message severity="WARN"
>#{org.jboss.seam.handledException.message}</message>
      </redirect>
   </exception>
```

```
</pages
>
```

`org.jboss.seam.handledException`
`#####################################################org.jboss.seam.caughtException###########`

6.12.4.1.

For the exception handlers defined in `pages.xml`, it is possible to declare the logging level at which the exception will be logged, or to even suppress the exception being logged altogether. The attributes `log` and `log-level` can be used to control exception logging. By setting `log="false"` as per the following example, then no log message will be generated when the specified exception occurs:

```
  <exception class="org.jboss.seam.security.NotLoggedInException" log="false">
    <redirect view-id="/register.xhtml">
      <message severity="warn"
>You must be a member to use this feature</message>
    </redirect>
  </exception
>
```

If the `log` attribute is not specified, then it defaults to `true` (i.e. the exception will be logged). Alternatively, you can specify the `log-level` to control at which log level the exception will be logged:

```
  <exception class="org.jboss.seam.security.NotLoggedInException" log-level="info">
    <redirect view-id="/register.xhtml">
      <message severity="warn">You must be a member to use this feature</message>
    </redirect>
  </exception>
```

The acceptable values for `log-level` are: `fatal`, `error`, `warn`, `info`, `debug` or `trace`. If the `log-level` is not specified, or if an invalid value is configured, then it will default to `error`.

6.12.5.

##JPA########

```
<exception class="javax.persistence.EntityNotFoundException">
  <redirect view-id="/error.xhtml">
    <message
```

```
>Not found</message>
  </redirect>
</exception>

<exception class="javax.persistence.OptimisticLockException">
  <end-conversation/>
  <redirect view-id="/error.xhtml">
    <message
>Another user changed the same data, please try again</message>
  </redirect>
</exception
>
```

##Seam#########################

```
<exception class="org.jboss.seam.framework.EntityNotFoundException">
  <redirect view-id="/error.xhtml">
    <message
>Not found</message>
  </redirect>
</exception
>
```

##Seam###############

```
<exception class="org.jboss.seam.security.AuthorizationException">
  <redirect>
    <message
>You don't have permission to do this</message>
  </redirect>
</exception>

<exception class="org.jboss.seam.security.NotLoggedInException">
  <redirect view-id="/login.xhtml">
    <message
>Please log in first</message>
  </redirect>
</exception
>
```

####JSF###

```
<exception class="javax.faces.application.ViewExpiredException">
  <redirect view-id="/error.xhtml">
    <message
>Your session has timed out, please try again</message>
  </redirect>
</exception
>
```

A `ViewExpiredException` occurs if the user posts back to a page once their session has expired. The `conversation-required` and `no-conversation-view-id` settings in the Seam page descriptor, discussed in *#7.4. #Requiring a long-running conversation#*, give you finer-grained control over session expiration if you are accessing a page used within a conversation.

##########

Seam

######## 3 ############## Seam ################

- ####### ######## ### 2002 ########## (#######) ################ ########### Struts ############################2 ######################
- ## ## (Hibernate #### `LazyInitializationException` ################## ####### Hibernate ########### ### Spring ######## J2EE ####### *stateless session facade* (anti) ##)
- ################

##################################### ##

7.1. Seam

###################################

- JSF ############ ####### ########## ######## ############## ################## ####################
- JSF ############################ Seam ######################### ######## Seam ######################
- `@Begin` ########## ########################
- `@End` ########## ################################
- JSF ################################## Seam ##################### ######################
- ##### faces ## (JSF ######) ################ ######## non-faces ## (#### GET ##) ################## ####################
- If the JSF request lifecycle is foreshortened by a redirect, Seam transparently stores and restores the current conversation context — unless the conversation was already ended via `@End(beforeRedirect=true)`.

Seam ############ (###############) # JSF ######################## ######### # *faces* ## (#### GET ####) ################################## ########## ################## non-faces ##### Seam ########### ########## Seam ## *ID (conversation id)* ##################

```
<a href="main.jsf?#{manager.conversationIdParameter}=#{conversation.id}"
>Continue</a
>
```

JSF ####################

```
<h:outputLink value="main.jsf">
    <f:param name="#{manager.conversationIdParameter}" value="#{conversation.id}"/>
    <h:outputText value="Continue"/>
</h:outputLink
>
```

Seam ################ ########

```
<h:outputLink value="main.jsf">
    <s:conversationId/>
    <h:outputText value="Continue"/>
</h:outputLink
>
```

################################ ##############

```
<h:commandLink action="main" value="Exit">
    <f:param name="conversationPropagation" value="none"/>
</h:commandLink
>
```

Seam ################ ########

```
<h:commandLink action="main" value="Exit">
    <s:conversationPropagation type="none"/>
</h:commandLink
>
```

###################### ################################

The `conversationPropagation` request parameter, or the `<s:conversationPropagation>` tag may even be used to begin a conversation, end the current conversation, destroy the entire conversation stack, or begin a nested conversation.

```
<h:commandLink action="main" value="Exit">
   <s:conversationPropagation type="end"/>
</h:commandLink
>
```

```
<h:commandLink action="main" value="Exit">
   <s:conversationPropagation type="endRoot"/>
</h:commandLink>
```

```
<h:commandLink action="main" value="Select Child">
   <s:conversationPropagation type="nested"/>
</h:commandLink
>
```

```
<h:commandLink action="main" value="Select Hotel">
   <s:conversationPropagation type="begin"/>
</h:commandLink
>
```

```
<h:commandLink action="main" value="Select Hotel">
   <s:conversationPropagation type="join"/>
</h:commandLink
>
```

######### ## ############################## ##

- *##### ## ##### ######## (nested conversation) ################ ####################*
- *## ########################*

7.2.

#################### @Begin(nested=true) ############################
################################ ############################
##
################################

- ## ###############
- ###
- ##
  ###################################### ########## ############

@End ####### ############(Pop)########
######################

############## (################) #### ################################
Seam
#####################################

############################ ############################ @End(root=true)
######################

###################### ############
####
####################################

######## ################################### ################################
##
@PerNestedConversation
########################

7.3. GET

non-faces ## (## HTTP GET ##) ############ JSF
######################## <h:outputLink> ######################

################################ JSF ############## ###### @Begin
####################################

################################ ############ ####2 ############### Seam
#################### @Create ################## ########## ##########
@Factory ################

################### Seam ## pages.xml ##### ######## ############

```
<pages>
  <page view-id="/messageList.jsp" action="#{messageManager.list}"/>
```

```
  ...
</pages
>
```

########## ########## ########
null ########## Seam #### JSF ### Seam ##########
##########

########## ##########

```
<pages>
  <page view-id="/messageList.jsp" action="#{conversation.begin}"/>
  ...
</pages
>
```

JSF ########## ### #{conversation.end}
##########

########## ########## <begin-conversation> ##########

```
<pages>
  <page view-id="/messageList.jsp">
    <begin-conversation nested="true" pageflow="AddItem"/>
  <page>
  ...
</pages
>
```

<end-conversation>

```
<pages>
  <page view-id="/home.jsp">
    <end-conversation/>
  <page>
  ...
</pages
>
```

1 ########## ## 5 ##########

- `@Create` ##### `@Begin` ############
- `@Factory` ##### `@Begin` ############
- Seam ############## `@Begin` ############
- `pages.xml` # `<begin-conversation>` #####
- `#{conversation.begin}` # Seam ###################

7.4. Requiring a long-running conversation

Certain pages are only relevant in the context of a long-running conversation. One way to "protect" such a page is to require a long-running conversation as a prerequisite to rendering the page. Fortunately, Seam has a built-in mechanism for enforcing this requirement.

In the Seam page descriptor, you can indicate that the current conversation must be long-running (or nested) in order for a page to be rendered using the `conversation-required` attribute as follows:

```
<page view-id="/book.xhtml" conversation-required="true"/>
```

##

The only downside is there's no built-in way to indicate *which* long-running conversation is required. You can build on this basic authorization by dually checking if a specific value is present in the conversation within a page action.

When Seam determines that this page is requested outside of a long-running conversation, the following actions are taken:

- A contextual event named `org.jboss.seam.noConversation` is raised
- A warning status message is registered using the bundle key `org.jboss.seam.NoConversation`
- The user is redirected to an alternate page, if defined

The alternate page is defined in the `no-conversation-view-id` attribute on a `<pages>` element in the Seam page descriptor as follows:

```
<pages no-conversation-view-id="/main.xhtml"/>
```

At the moment, you can only define one such page for the entire application.

7.5. <s:link> # <s:button>

JSF ########## JavaScript ################ ## ##### JFS ############## <h:outputLink> ############# #### <h:outputLink> ######## 2 ######

- JSF ########### <h:outputLink> #################
- ################## JSF ###### DataModel ##########

Seam provides the notion of a *page action* to help solve the first problem, but this does nothing to help us with the second problem. We *could* work around this by using the RESTful approach of passing a request parameter and requerying for the selected object on the server side. In some cases — such as the Seam blog example application — this is indeed the best approach. The RESTful style supports bookmarking, since it does not require server-side state. In other cases, where we don't care about bookmarks, the use of `@DataModel` and `@DataModelSelection` is just so convenient and transparent!

############################## Seam # <s:link> JSF ###########

JSF ### ID

```
<s:link view="/login.xhtml" value="Login"/>
```

(##### #####################)#

```
<s:link action="#{login.logout}" value="Logout"/>
```

JSF ### ID ################### ########## *null* ##########################

```
<s:link view="/loggedOut.xhtml"  action="#{login.logout}" value="Logout"/>
```

<h:dataTable> ###### DataModel

```
<s:link view="/hotel.xhtml" action="#{hotelSearch.selectHotel}" value="#{hotel.name}"/>
```

########################

```
<s:link view="/main.xhtml" propagation="none"/>
```

################

```
<s:link action="#{issueEditor.viewComment}" propagation="nested"/>
```

########## ########################

```
<s:link action="#{documentEditor.getDocument}" propagation="begin"
        pageflow="EditDocument"/>
```

The `taskInstance` attribute is for use in jBPM task lists:

```
<s:link action="#{documentApproval.approveOrReject}" taskInstance="#{task}"/>
```

(##### DVD ########################)

`<s:button>`

```
<s:button action="#{login.logout}" value="Logout"/>
```

7.6.

JSF `FacesMessage` ######### ###### ############################## JSF ########## faces ############## ######## JSF

########### Seam ########## `facesMessages` ################ (Seam ##############################)

```
@Name("editDocumentAction")
@Stateless
public class EditDocumentBean implements EditDocument {
   @In EntityManager em;
   @In Document document;
   @In FacesMessages facesMessages;

   public String update() {
      em.merge(document);
      facesMessages.add("Document updated");
   }
```

```
}
```

`facesMessages` ## ### Seam ##

JSF EL ## faces ######################

```
facesMessages.add("Document #{document.title} was updated");
```

########################

```
<h:messages globalOnly="true"/>
```

7.7. ######## ID

################################## ID ##

###############

########## 2 ################################## ######### # #ebay ## ################################## ################################## ## ############################ #

With a natural conversation it's really easy to have the user rejoin the existing conversation, and pick up where they left off - just have them to rejoin the payForItem conversation with the itemId as the conversation id.

########### *URL*

############## (URL ###########) ##### URL (## Wiki ############## ID #########) ################## ######################### URL ######################

With a natural conversation, when you are building your hotel booking system (or, of course, whatever your app is) you can generate a URL like `http://seam-hotels/book.seam?hotel=BestWesternAntwerpen` (of course, whatever parameter `hotel` maps to on your domain model must be unique) and with URLRewrite easily transform this to http://seam-hotels/book/BestWesternAntwerpen.

##############

7.8.

######## `pages.xml` ########

```
<conversation name="PlaceBid"
              parameter-name="auctionId"
              parameter-value="#{auction.auctionId}"/>
```

########################## #### PlaceBid ###### ################### page ########################

parameter-name ######### ID ################## ID ################ ##### parameter-name # auctionId ###### #### #### URL ###### cid=123 ################ auctionId=765432

############ parameter-value ### ID ############################## EL ######## ######## ID ######## auction ################

###################### page ### conversation ###############

```
<page view-id="/bid.xhtml" conversation="PlaceBid" login-required="true">
    <navigation from-action="#{bidAction.confirmBid}"
>
        <rule if-outcome="success">
            <redirect view-id="/auction.xhtml">
                <param name="id" value="#{bidAction.bid.auction.auctionId}"/>
            </redirect>
        </rule
>
    </navigation>
</page
>
```

7.9.

######################### ############################## ####################

```
<page view-id="/auction.xhtml">
  <param name="id" value="#{auctionDetail.selectedAuctionId}"/>

  <navigation from-action="#{bidAction.placeBid}">
    <redirect view-id="/bid.xhtml"/>
  </navigation>
</page
>
```

auction ########## #{bidAction.placeBid} ########## (############ Seam ### seamBay ########)# ############## PlaceBid ###### /bid.xhtml ##################### ##########################

```
@Begin(join = true)
public void placeBid()
```

############ <page/> ############## ## ## ################################### s:conversationName ########################

```
<h:commandButton id="placeBidWithAmount" styleClass="placeBid"
 action="#{bidAction.placeBid}">
  <s:conversationName value="PlaceBid"/>
 </h:commandButton
>
```

######### s:link ### s:button ############## conversationName ##########

```
<s:link value="Place Bid" action="#{bidAction.placeBid}" conversationName="PlaceBid"/>
```

7.10.

##########1 ########################## Seam ########## Java ################### ###################################

- ######## ID (JSF ### Seam #################) ######### (jPDL ######) ################ ######################################
- ###### 1 ################## JSF ### facelets ########## ################## ####### ####### (breadcrumbs) ######################

7.10.1. ########## JSF

JSF ### Seam ################## Seam ####### view-id ################ ################## pages.xml ################ Seam ######## WEB-INF ######## faces-config.xml ####################

```
<pages>
```

```
    <page view-id="/main.xhtml">
      <description
>Search hotels: #{hotelBooking.searchString}</description>
    </page>
    <page view-id="/hotel.xhtml">
      <description
>View hotel: #{hotel.name}</description>
    </page>
    <page view-id="/book.xhtml">
      <description
>Book hotel: #{hotel.name}</description>
    </page>
    <page view-id="/confirm.xhtml">
      <description
>Confirm: #{booking.description}</description>
    </page>
    </pages
>
```

################## Seam ###################### ############################

7.10.2. ########## jPDL

jPDL ############## Seam #### jBPM ############################## ## `view-id` #### `<page>` ### ######## `<page>` ###########

```
<pageflow-definition name="shopping">

  <start-state name="start">
    <transition to="browse"/>
  </start-state>

  <page name="browse" view-id="/browse.xhtml">
    <description
>DVD Search: #{search.searchPattern}</description>
    <transition to="browse"/>
    <transition name="checkout" to="checkout"/>
  </page>

  <page name="checkout" view-id="/checkout.xhtml">
    <description
>Purchase: $#{cart.total}</description>
    <transition to="checkout"/>
    <transition name="complete" to="complete"/>
```

```
  </page>

  <page name="complete" view-id="/complete.xhtml">
    <end-conversation />
  </page>

</pageflow-definition
>
```

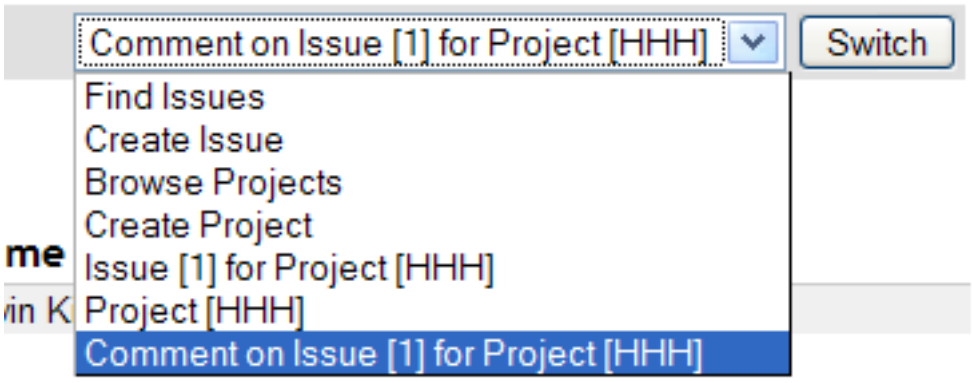
7.10.3.

JSP ### facelets
##

```
<h:selectOneMenu value="#{switcher.conversationIdOrOutcome}">
   <f:selectItem itemLabel="Find Issues" itemValue="findIssue"/>
   <f:selectItem itemLabel="Create Issue" itemValue="editIssue"/>
   <f:selectItems value="#{switcher.selectItems}"/>
</h:selectOneMenu>
<h:commandButton action="#{switcher.select}" value="Switch"/>
```

2 ##############

######## (`pages.xml` ###) ######################



7.10.4.

######################## #################

```
<h:dataTable value="#{conversationList}" var="entry"
```

```
        rendered="#{not empty conversationList}">
    <h:column>
        <f:facet name="header"
>Workspace</f:facet>
        <h:commandLink action="#{entry.select}" value="#{entry.description}"/>
        <h:outputText value="[current]" rendered="#{entry.current}"/>
    </h:column>
    <h:column>
        <f:facet name="header"
>Activity</f:facet>
        <h:outputText value="#{entry.startDatetime}">
            <f:convertDateTime type="time" pattern="hh:mm a"/>
        </h:outputText>
        <h:outputText value=" - "/>
        <h:outputText value="#{entry.lastDatetime}">
            <f:convertDateTime type="time" pattern="hh:mm a"/>
        </h:outputText>
    </h:column>
    <h:column>
        <f:facet name="header"
>Action</f:facet>
        <h:commandButton action="#{entry.select}" value="#{msg.Switch}"/>
        <h:commandButton action="#{entry.destroy}" value="#{msg.Destroy}"/>
    </h:column>
</h:dataTable
>
```

##

Workspace	Workspace activity	Action
Comment on Issue [1] for Project [HHH]	01:18 PM - 01:18 PM	Switch Destroy
Issue [1] for Project [HHH]	01:18 PM - 01:18 PM	Switch Destroy
Project [HHH]	01:18 PM - 01:18 PM	Switch Destroy

######################

##

7.10.5. ####### (Breadcrumbs)

######### ############################## ######### ############################

```
<ui:repeat value="#{conversationStack}" var="entry">
```

```
    <h:outputText value=" | "/>
    <h:commandLink value="#{entry.description}" action="#{entry.select}"/>
</ui:repeat
```

7.11. ########## JSF

########## JSF ## (###### ###################################### ################## JSF ######################) postback #### Seam ################ ###################################

########## ## ################################

```
@Name("grid")
@Scope(ScopeType.EVENT)
public class Grid
{
    private HtmlPanelGrid htmlPanelGrid;

    // getters and setters
    ...
}
```

```
@Name("gridEditor")
@Scope(ScopeType.CONVERSATION)
public class GridEditor
{
    @In(required=false)
    private Grid grid;

    ...
}
```

JSF ## #### `facesMessages` ############## Seam

Alternatively, you can access the JSF component tree through the implicit `uiComponent` handle. The following example accesses `getRowIndex()` of the `UIData` component which backs the data table during iteration, it prints the current row number:

```
<h:dataTable id="lineItemTable" var="lineItem" value="#{orderHome.lineItems}">
  <h:column>
    Row: #{uiComponent['lineItemTable'].rowIndex}
  </h:column>
  ...
</h:dataTable
>
```

JSF UI ####################################

7.12.

Seam ###################### *#4.1.10.* ########## ######## ################################ (AJAX ######################)# ################### Ajax ################################## RichFaces ######################

################################ Seam ############################## ############################ #### ################### ### ############################## ## (##########)# ### ######################## AJAX ##

Seam ######################### (##########)# ## ##### AJAX

components.xml ##

```
<core:manager concurrent-request-timeout="500" />
```

################################

```
<page view-id="/book.xhtml"
      conversation-required="true"
      login-required="true"
      concurrent-request-timeout="2000" />
```

###################### AJAX ################ ##########################
###################### ##### AJAX ################# (1
#####################################)# #### ### 1 ##########################

################################ ############ AJAX ####### ##########################
########################## (################)# ##########################
################## (#############) ######################################

7.12.1. ### AJAX

################################

################################### ######################################
############################### (#####)#
(######## keypress
onblur ##) ##
#####################################

##

###################################### ######### @Asynchronous
########################

```
int total;

// This method is called when an event occurs on the client
// It takes a really long time to execute
@Asynchronous
public void calculateTotal() {
  total = someReallyComplicatedCalculation();
}

// This method is called as the result of the poll
// It's very quick to execute
public int getTotal() {
  return total;
}
```

7.12.2.

##
############################ concurrent-request-timeout
############################ ####### Seam # pages.xml #######
ConcurrentRequestTimeoutException ####### HTTP 503 ##############

```
<exception class="org.jboss.seam.ConcurrentRequestTimeoutException" log-level="trace">
  <http-error error-code="503" />
</exception>
```

503 Service Unavailable (HTTP/1.1 RFC)

######## ##
###################################

#############################

```
<exception class="org.jboss.seam.ConcurrentRequestTimeoutException" log-level="trace">
  <end-conversation/>
  <redirect view-id="/error.xhtml">
    <message>The server is too busy to process your request, please try again later</message>
  </redirect>
</exception>
```

ICEfaces, RichFaces Ajax and Seam Remoting can all handle HTTP error codes. Seam Remoting will pop up a dialog box showing the HTTP error. ICEfaces will indicate the error in its connection status component. RichFaces provides the most complete support for handling HTTP errors by providing a user definable callback. For example, to show the error message to the user:

```
<script type="text/javascript">
  A4J.AJAX.onError = function(req,status,message) {
    alert("An error occurred");
  };
</script>
```

If instead of an error code, the server reports that the view has expired, perhaps because the session timed out, you use a separate callback function in RichFaces to handle this scenario.

```
<script type="text/javascript">
  A4J.AJAX.onExpired = function(loc,message) {
    alert("View expired");
  };
</script>
```

Alternatively, you can allow RichFaces handle the error, in which case the user will be presented with a prompt that reads "View state could't be restored - reload page?" You can customize this message globally by setting the following message key in an application resource bundle.

```
AJAX_VIEW_EXPIRED=View expired. Please reload the page.
```

7.12.3. RichFaces (Ajax4jsf)

RichFaces (Ajax4jsf) is the Ajax library most commonly used with Seam, and provides all the controls discussed above:

- `eventsQueue` — provides a queue in which events are placed. All events are queued and requests are sent to the server serially. This is useful if the request to the server can take some time to execute (e.g. heavy computation, retrieving information from a slow source) as the server isn't flooded.
- `ignoreDupResponses` — ignores the response produced by the request if a more recent 'similar' request is already in the queue. ignoreDupResponses="true" does *not cancel* the processing of the request on the server side — just prevents unnecessary updates on the client side.

  ################################ Seam ##################
- `requestDelay` — ################### (#####)# ############################ (###################) ## (####################) #####

  ############################### Seam ################### ####################### (##############) ####################
- `<a:poll reRender="total" interval="1000" />` — ############################

###############

JBoss jBPM #Java SE # EE ###################### jBPM ######################## ###############WEB######################## ############## jPDL ##### XML ############### Eclipse ################################ jPDL ###########WEB ##SOA ##############################

Seam ######### jBPM #2 ##############

- ################################## jPDL ############################ Seam##
- ################### ###################################### ##### jBPM ################################ ################### ### ####jBPM ##

##*Pageflow*, *conversation* # *task* ##############################

Seam######JPDL#########################Seam###

8.1. Seam#######

Seam ############# 2 ##########

- JSF####Seam ############ - #################
- jPDL ### - #################

################################### ############################## ##########################

8.1.1.

########### ################### (outcome) ## ########################## ## ################################## ## ##

JSF

```
<navigation-rule>
   <from-view-id
>/numberGuess.jsp</from-view-id>

   <navigation-case>
      <from-outcome
>guess</from-outcome>
```

```
        <to-view-id
>/numberGuess.jsp</to-view-id>
        <redirect/>
    </navigation-case>

    <navigation-case>
        <from-outcome
>win</from-outcome>
        <to-view-id
>/win.jsp</to-view-id>
        <redirect/>
    </navigation-case>

    <navigation-case>
        <from-outcome
>lose</from-outcome>
        <to-view-id
>/lose.jsp</to-view-id>
        <redirect/>
    </navigation-case>

</navigation-rule
>
```

Seam

```
<page view-id="/numberGuess.jsp">

    <navigation>
        <rule if-outcome="guess">
            <redirect view-id="/numberGuess.jsp"/>
        </rule>
        <rule if-outcome="win">
            <redirect view-id="/win.jsp"/>
        </rule>
        <rule if-outcome="lose">
            <redirect view-id="/lose.jsp"/>
        </rule>
    </navigation>

</page
>
```

######################## ######################ID############

```
public String guess() {
   if (guess==randomNumber) return "/win.jsp";
   if (++guessCount==maxGuesses) return "/lose.jsp";
   return null;
}
```

########################## ###############################

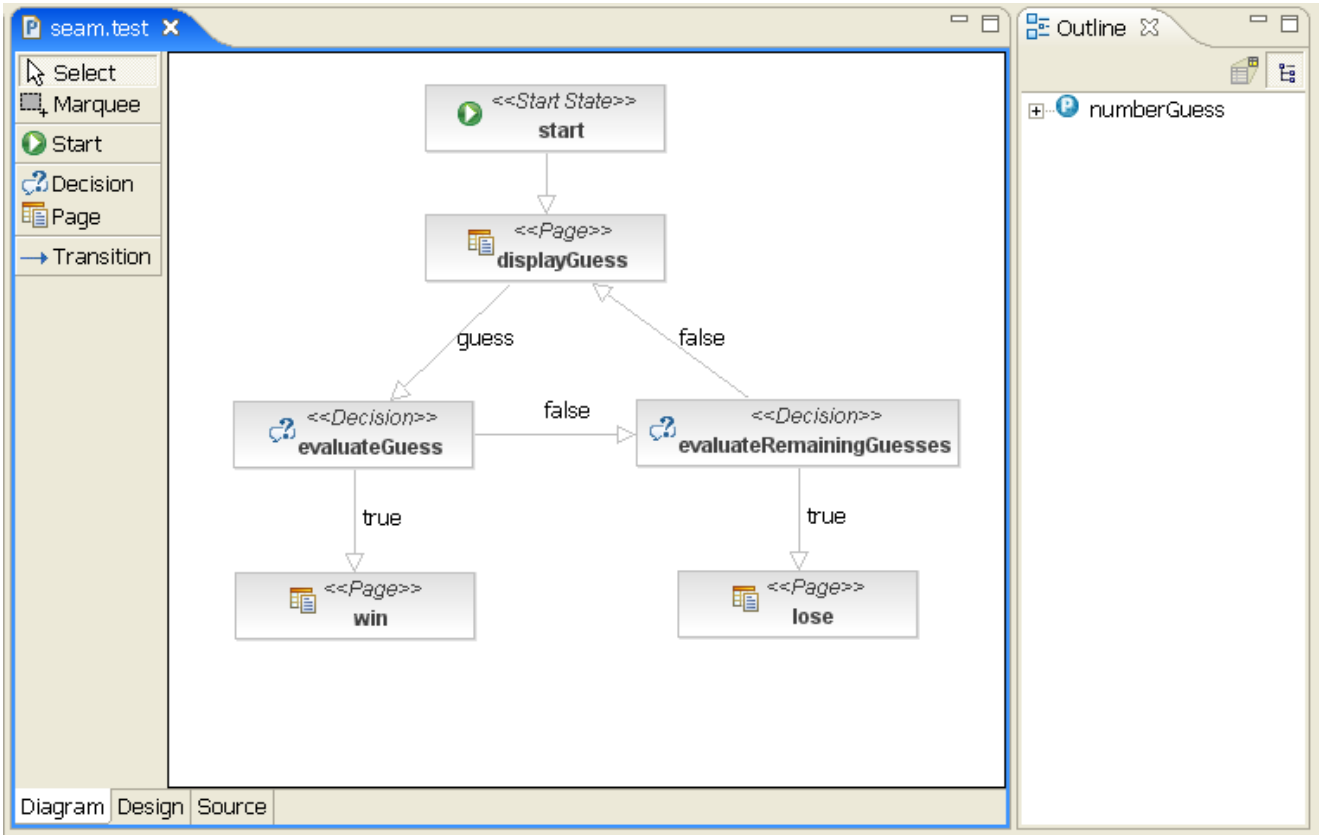
```
public String search() {
   return "/searchResults.jsp?searchPattern=#{searchAction.searchPattern}";
}
```

#######jPDL
##
##

jPDL

```
<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
>
```



######### ###########

- JSF/Seam ############# ##### (############# Java ##############################)
- jPDL #JSP # Java ############ ########################

#################### ### ### ####### (############)
################################ ################ ###### ################
##

/ ################## ######## /
################################ ##### 1 ########## Seam ##########

###################### ####Seam ######### ############################ ########## ###################### ############################ ##################################

############## ############

8.1.2. Seam #

JSF #### Seam ################## Seam ################################### ########## ##################### ############## Struts # WebWork ################# WEB ################ ####EJB ##########Bean # Spring framework ############################### ## ####Seam ############## ########## Bean ######################### ############ ######################### no-conversation-view-id ######## null ######################## ######################### ####################

#####no-conversation-view-id#### pages.xml###### ######################## ### Seam ################################

```
<page view-id="/checkout.xhtml"
      no-conversation-view-id="/main.xhtml"/>
```

On the other hand, in the stateful model, using the back button is interpreted as an undefined transition back to a previous state. Since the stateful model enforces a defined set of transitions from the current state, the back button is not permitted by default in the stateful model! Seam transparently detects the use of the back button, and blocks any attempt to perform an action from a previous, "stale" page, and simply redirects the user to the "current" page (and displays a faces message). Whether you consider this a feature or a limitation of the stateful model depends upon your point of view: as an application developer, it is a feature; as a user, it might be frustrating! You can enable backbutton navigation from a particular page node by setting `back="enabled"`.

```
<page name="checkout"
      view-id="/checkout.xhtml"
      back="enabled">
   <redirect/>
   <transition to="checkout"/>
   <transition name="complete" to="complete"/>
</page
>
```

This allows navigation via the back button *from* the `checkout` state to *any previous state!*

##

If a page is set to redirect after a transition, it is not possible to use the back button to return to that page even when back is enabled on a page later in the flow. The reason is because Seam stores information about the pageflow in the page scope and the back button must result in a POST for that information to be restored (i.e., a Faces request). A redirect severs this linkage.

##
##################no-conversation-view-id ##################:

```
<page name="checkout"
      view-id="/checkout.xhtml"
      back="enabled"
      no-conversation-view-id="/main.xhtml">
   <redirect/>
   <transition to="checkout"/>
   <transition name="complete" to="complete"/>
</page
>
```

###

8.2. jPDL

8.2.1.

Seam # jBPM ###################Seam ##### # seam.properties ########### # ############### (### jpdl.xml #######) ############ ## components.xml # Seam ###############

```
<bpm:jbpm />
```

Seam # jBPM ### ## components.xml # Seam ###############

```
<bpm:jbpm>
   <bpm:pageflow-definitions>
      <value
>pageflow.jpdl.xml</value>
   </bpm:pageflow-definitions>
```

```
</bpm:jbpm
>
```

8.2.2.

`@Begin#@BeginTask` ##### `@StartTask` ############# ##################### jPDL ################:

```
@Begin(pageflow="numberguess")
public void begin() { ... }
```

#####pages.xml###################

```
<page>
      <begin-conversation pageflow="numberguess"/>
   </page
>
```

If we are beginning the pageflow during the `RENDER_RESPONSE` phase — during a `@Factory` or `@Create` method, for example — we consider ourselves to be already at the page being rendered, and use a `<start-page>` node as the first node in the pageflow, as in the example above.

(outcome) ######################## ###################### `<start-state>` ##### ########## (outcome)

```
<pageflow-definition name="viewEditDocument">

   <start-state name="start">
      <transition name="documentFound" to="displayDocument"/>
      <transition name="documentNotFound" to="notFound"/>
   </start-state>

   <page name="displayDocument" view-id="/document.jsp">
      <transition name="edit" to="editDocument"/>
      <transition name="done" to="main"/>
   </page>

   ...

   <page name="notFound" view-id="/404.jsp">
```

```
        <end-conversation/>
    </page>

</pageflow-definition
>
```

8.2.3.

<page>

```
<page name="displayGuess" view-id="/numberGuess.jsp">
    <redirect/>
    <transition name="guess" to="evaluateGuess">
        <action expression="#{numberGuess.guess}" />
    </transition>
</page
>
```

view-id #JSF####ID### <redirect/>###JSF##############<redirect/>############################post-then-redirect#####################(Seam#######################################Ruby on Rails # "flash"#######Seam############

#####numberGuess.jsp ###### ### ############################ JSF ## (outcome) ######

```
<h:commandButton type="submit" value="Guess" action="guess"/>
```

################################ numberGuess ######## guess () ################ jBPM ################ jPDL ############################ JSF EL ########## ################## ######## Seam ########## Seam ########################## ####JSF ########################## jBPM ############### (###### (The One Kind of Stuff principle))

null##outcome ### (####action ################)# ##################Seam ################## ## ################################

```
<h:commandButton type="submit" value="Guess"/>
```

##########################

```
<page name="displayGuess" view-id="/numberGuess.jsp">
   <redirect/>
   <transition to="evaluateGuess">
      <action expression="#{numberGuess.guess}" />
   </transition>
</page
>
```

########################## ########### (outcome) ##########

```
<h:commandButton type="submit" value="Guess" action="#{numberGuess.guess}"/>
```

```
<page name="displayGuess" view-id="/numberGuess.jsp">
   <transition name="correctGuess" to="win"/>
   <transition name="incorrectGuess" to="evaluateGuess"/>
</page
>
```

######################## ##
##################################

8.2.4.

###################jPDL#####################<decision>#########

```
<decision name="evaluateGuess" expression="#{numberGuess.correctGuess}">
   <transition name="true" to="win"/>
   <transition name="false" to="evaluateRemainingGuesses"/>
</decision
>
```

decision (####) # Seam ######### JSF EL #############

8.2.5.

<end-conversation>####@End ############### (############ #########)

```
<page name="win" view-id="/win.jsp">
   <redirect/>
```

```
    <end-conversation/>
</page
>
```

########transition ###################### ####Seam ##################################

```
<page name="win" view-id="/win.jsp">
    <redirect/>
    <end-task transition="success"/>
</page
>
```

8.2.6.

##<process-state> ###################################

```
<process-state name="cheat">
    <sub-process name="cheat"/>
    <transition to="displayGuess"/>
</process-state
>
```

<start-state> ################# <end-state> ############################<process-state> ##################

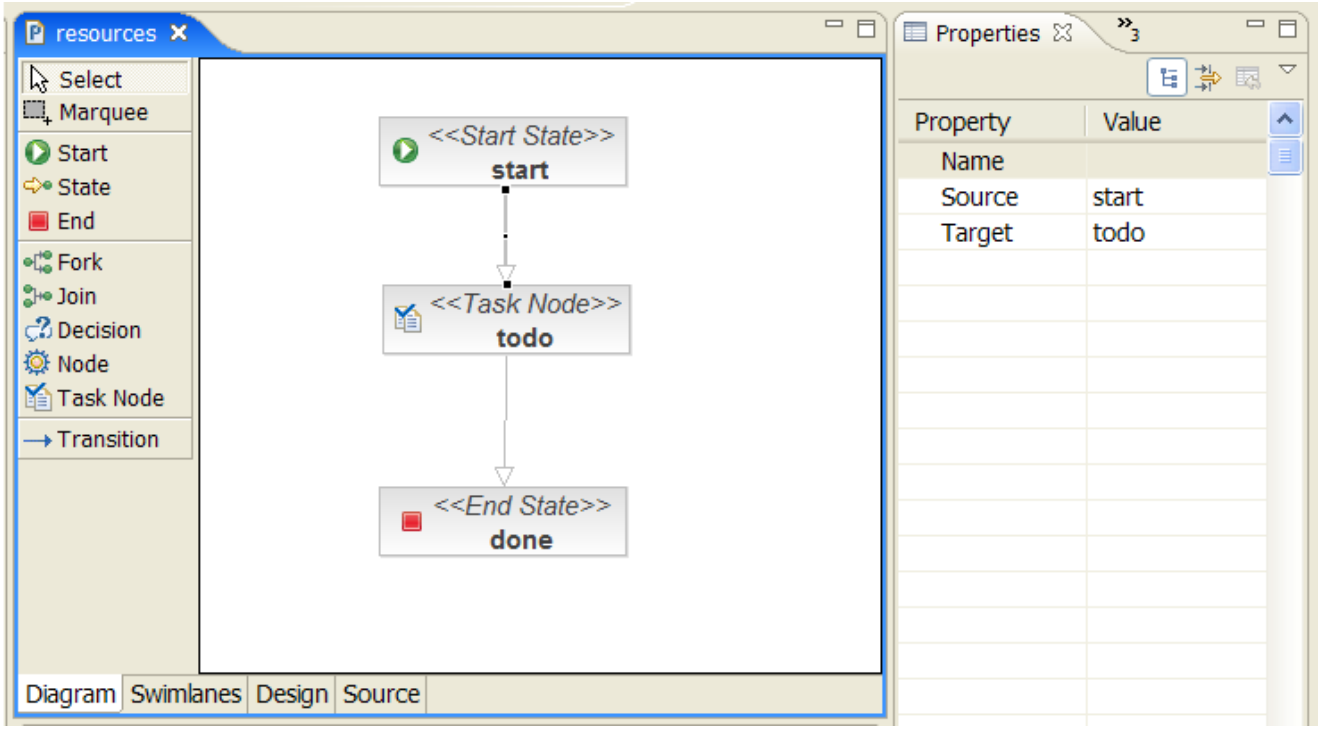
8.3. Seam

Seam # jBPM ### Seam

<page> #####<task-node> ######### ## (###### (The One Kind of Stuff principle)) ################ ##

```
<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
>
```



###########jPDL ########### jPDL ################### ######2 ##### ########## <task># ###### <process-definition>#########

8.4. jPDL

8.4.1.

jBPM #######jBPM############################

```
<bpm:jbpm>
```

```
   <bpm:process-definitions>
      <value
>todo.jpdl.xml</value>
   </bpm:process-definitions>
</bpm:jbpm
>
```

jBPM ########################## Seam ##Seam##############jBPM############ ####################

8.4.2. ####ID####

################################## jBPM *#actor id # group actor id*################ `actor` ######### Seam ##################### actor id #######

```
@In Actor actor;

public String login() {
   ...
   actor.setId( user.getUserName() );
   actor.getGroupActorIds().addAll( user.getGroupNames() );
   ...
}
```

8.4.3.

######################### `@CreateProcess` ###############

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

pages.xml#####################:

```
<page>
   <create-process definition="todo" />
</page
>
```

8.4.4.

##ID###

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

####################

```
<task name="todo" description="#{todoList.description}">
   <assignment pooled-actors="employees"/>
</task
>
```

8.4.5.

######### Seam ############################## pooledTaskInstanceList #
#####################################

```
<h:dataTable value="#{pooledTaskInstanceList}" var="task">
   <h:column>
      <f:facet name="header"
>Description</f:facet>
      <h:outputText value="#{task.description}"/>
   </h:column>
   <h:column>
     <s:link action="#{pooledTask.assignToCurrentActor}" value="Assign" taskInstance="#{task}"/
>
   </h:column
>
</h:dataTable
>
```

<s:link> ######## JSF <h:commandLink> ##############

```
<h:commandLink action="#{pooledTask.assignToCurrentActor}"
>
   <f:param name="taskId" value="#{task.id}"/>
```

```
</h:commandLink
>
```

pooledTask ##

taskInstanceListForType ######### ################################

```
<h:dataTable value="#{taskInstanceListForType['todo']}" var="task">
   <h:column>
      <f:facet name="header"
>Description</f:facet>
      <h:outputText value="#{task.description}"/>
   </h:column>
   <h:column>
      <s:link action="#{todoList.start}" value="Start Work" taskInstance="#{task}"/>
   </h:column
>
</h:dataTable
>
```

8.4.6.

########################## @StartTask #### @BeginTask#######

```
@StartTask
public String start() { ... }
```

pages.xml ###############:

```
<page>
   <start-task />
</page
>
```

##
##

@EndTask ################Seam #################

```
@EndTask(transition="completed")
```

```
public String completed() { ... }
```

pages.xml#######:

```
<page>
   <end-task transition="completed" />
</page
>
```

pages.xml####EL############

#####jBPM ########################## (##)

jBPM

Seam

Seam # EJB 3.0 ###### Java Persistence API ### Hibernate3 ########## Java ############################## Seam ################## ############################## ORM #########

9.1.

Seam grew out of the frustration of the Hibernate team with the statelessness typical of the previous generation of Java application architectures. The state management architecture of Seam was originally designed to solve problems relating to persistence — in particular problems associated with *optimistic transaction processing*. Scalable online applications always use optimistic transactions. An atomic (database/JTA) level transaction should not span a user interaction unless the application is designed to support only a very small number of concurrent clients. But almost all interesting work involves first displaying data to a user, and then, slightly later, updating the same data. So Hibernate was designed to support the idea of a persistence context which spanned an optimistic transaction.

Seam # EJB 3.0 ## ##### ## ### ###################### ## Hibernate ################## `LazyInitializationException` ##########

EJB 3.0 ########## ##################### ########## ################## (########### Bean) ############## ############################ (############) # ###########

- ########### Bean ################################## (################################)#
- ##

Seam ## ############################ Bean ################## 1 ############ (####################################) ###################################### (Seam #########) ########## ## ###################################### ##### Seam # EJB 3.0 ##

9.2. Seam

EJB ##### Bean ######################### EJB ##### Bean ######################### ################################ JSF ###################### Bean ############ ################## 1 ##################

Seam

1 ###### Seam ########################## Bean

- The request might require processing by several loosely-coupled components, each of which is called independently from the web layer. It is common to see several or even many calls per request from the web layer to EJB components in Seam.
- ################################

1 ##################### ## ########### ### ############################

Hibernate ################ *#open session in view#* ############ Spring ### ############################## `LazyInitializationException` ######### Hibernate #########open session in view##############

This pattern is usually implemented as a single transaction which spans the entire request. There are several problems with this implementation, the most serious being that we can never be sure that a transaction is successful until we commit it — but by the time the "open session in view" transaction is committed, the view is fully rendered, and the rendered response may already have been flushed to the client. How can we notify the user that their transaction was unsuccessful?

Seam ################################### #open session in view############## ##############

- ######################### ############################
- ############################ 1 ################ (##) ############################ 2 ######################

########## ################################### ##### Seam ################################## Seam ### ## Seam ###################### Seam ## #### ############################ ##########

Seam ########## EJB 3.0 ################################## Java EE 5 ###### Seam ######## ######### Seam ##############################

9.2.1. Seam

Seam ###################### JSF ############ ##### ###### ### `components.xml` ##########

```
<core:init transaction-management-enabled="false"/>

<transaction:no-transaction />
```

9.2.2. Seam##################

Seam ############## ##### ####### ############################## ####### Seam ############## EJB ############ JTA ###################### Java EE 5 ############ `components.xml` # EJB ########################

```
<transaction:ejb-transaction />
```

EE 5 ################ Seam ################################## Seam

- JPA RESOURCE_LOCAL ######### `javax.persistence.EntityTransaction` ############## `EntityTransaction` ##############################
- Hibernate ########### `org.hibernate.Transaction` ############# `HibernateTransaction` ##############################
- Spring ########### `org.springframework.transaction.PlatformTransactionManager` ############## Spring # `PlatformTransactionManagement` ###### `userConversationContext` ##
- Seam ###################

components.xml ####### JPA RESOURCE_LOCAL ################# `#{em}` # `persistence:managed-persistence-context` ########### ############ `entityManager` ## `entity-manager` ################ (*Seam* *###########* ###)

```
<transaction:entity-transaction entity-manager="#{em}"/>
```

Hibernate ################# ## components.xml ####### `#{hibernateSession}` ######## `persistence:managed-hibernate-session` ########### ## Hibernate ####### `session` ## `session` ################ (*Seam* *###########* ###)

```
<transaction:hibernate-transaction session="#{hibernateSession}"/>
```

Seam ######################## components.xml #######

```
<transaction:no-transaction />
```

Spring ############## *Spring # PlatformTransactionManagement #####* ##########

9.2.3.

############ `beforeCompletion()` # `afterCompletion()` ################################ ####### Seam ################################ ## Seam ################################ Java EE 5 #### `<transaction:ejb-transaction/>` ######## `components.xml` ##### ##### Seam ########################## Seam ##############################

9.3. Seam

Seam # Java EE 5 ############## ################################ EE 5 ######## ## ##

######### ######## ########## (JPA #) ### ####### (Hibernate #) ################## Seam ################## Seam ########## ######### `EntityManager` ### `Session` ############## `@In` ##################

Seam ################################ EJB 3.0 ## Seam ############ ## (################## ### EJB ######################)

9.3.1. JPA # Seam

################## `components.xml` ############

```
<persistence:managed-persistence-context name="bookingDatabase"
                                         auto-create="true"
                                         persistence-unit-jndi-name="java:/EntityManagerFactories/bookingData"/>
```

############## `bookingDatabase` ###### Seam ############ JNDI # `java:/EntityManagerFactories/bookingData` ######### (`EntityManagerFactory` ######) # `EntityManager` ################

`EntityManagerFactory` # JNDI ###################### JBoss ### ########## `persistence.xml`

```
<property name="jboss.entity.manager.factory.jndi.name"
```

```
        value="java:/EntityManagerFactories/bookingData"/>
```

######## `EntityManager` ############

```
@In EntityManager bookingDatabase;
```

EJB 3 ################## `@TransactionAttribute(REQUIRES_NEW)` ## #### Seam ################################### `REQUIRES_NEW` ################### ###### ##### `REQUIRES_NEW` ######### @PersistenceContext ############################

9.3.2. Seam ### Hibernate

Seam ## Hibernate #########`components.xml` ##################

```
<persistence:hibernate-session-factory name="hibernateSessionFactory"/>

<persistence:managed-hibernate-session name="bookingDatabase"
                        auto-create="true"
                session-factory-jndi-name="java:/bookingSessionFactory"/>
```

`java:/bookingSessionFactory` # `hibernate.cfg.xml` ######################

```
<session-factory name="java:/bookingSessionFactory">
   <property name="transaction.flush_before_completion"
>true</property>
   <property name="connection.release_mode"
>after_statement</property>
   <property name="transaction.manager_lookup_class"
>org.hibernate.transaction.JBossTransactionManagerLookup</property>
   <property name="transaction.factory_class"
>org.hibernate.transaction.JTATransactionFactory</property>
   <property name="connection.datasource"
>java:/bookingDatasource</property>
   ...
</session-factory
>
```

Seam ################## `hibernate.transaction.flush_before_completion` ############## JTA ##

JavaBean ########## Hibernate `Session`

```
@In Session bookingDatabase;
```

9.3.3. Seam

`merge()` ######### ################## `LazyInitializationException` # `NonUniqueObjectException` ######### ##

Hibernate # EJB 3.0 #### `@Version`

######## ################################ (############)# ################## ## ######################## EJB 3.0 ############ JBoss# Sun# Sybase ## EJB 3.0 ## #### Hibernate ############ `FlushModeType` ########################## ### ##################################

Seam ######### `FlushModeType.MANUAL` ############ ### Hibernate ############################ ##########################

```
@In EntityManager em; //a Seam-managed persistence context

@Begin(flushMode=MANUAL)
public void beginClaimWizard() {
   claim = em.find(Claim.class, claimId);
}
```

`claim` ############## ################### ## claim

```
public void addPartyToClaim() {
   Party party = ....;
   claim.addParty(party);
}
```

##

```
@End
```

```
public void commitClaim() {
   em.flush();
}
```

pages.xml ## `flushMode` # `MANUAL` ##############

```
<begin-conversation flush-mode="MANUAL" />
```

Seam

```
<components xmlns="http://jboss.com/products/seam/components"
  xmlns:core="http://jboss.com/products/seam/core">
  <core:manager conversation-timeout="120000" default-flush-mode="manual" />
</components
>
```

9.4. JPA

`EntityManager` ########## `getDelegate()` ############### API ############### ##### Hibernate ################ `org.hibernate.Session` ########################## ########################## ############ ## JPA ################### *### JPA ##########* ########

Hibernate ########################## #### Seam ################################

```
@In EntityManager entityManager;

@Create
public void init() {
   ( (Session) entityManager.getDelegate() ).enableFilter("currentVersions");
}
```

Java ###################### ################ ###################### ### ##### `components.xml`

```
<factory name="session"
      scope="STATELESS"
      auto-create="true"
```

```
        value="#{entityManager.delegate}"/>
```

############################

```
@In Session session;

@Create
public void init() {
   session.enableFilter("currentVersions");
}
```

9.5. EJB-QL/HQL # EL

Seam ################## @PersistenceContext ################################ Seam # EntityManager ### Session ############### ###### EL ###################################### ##### #########

```
User user = em.createQuery("from User where username=#{user.username}")
        .getSingleResult();
```

##############

```
User user = em.createQuery("from User where username=:username")
        .setParameter("username", user.getUsername())
        .getSingleResult();
```

####################

```
User user = em.createQuery("from User where username=" + user.getUsername()) //BAD!
        .getSingleResult();
```

(###### SQL ######################)

9.6. Hibernate

Hibernate ######### #### ###### ## ######### Hibernate ################## ##### ##### Seam ###################################### ## Seam Application Framework ################

Seam ################################ #### `EntityManager` # Hibernate `Session` ###################### (### Hibernate ############################)

```
<persistence:filter name="regionFilter">
   <persistence:name
>region</persistence:name>
   <persistence:parameters>
      <key
>regionCode</key>
      <value
>#{region.code}</value>
   </persistence:parameters>
</persistence:filter>

<persistence:filter name="currentFilter">
   <persistence:name
>current</persistence:name>
   <persistence:parameters>
      <key
>date</key>
      <value
>#{currentDate}</value>
   </persistence:parameters>
</persistence:filter>

<persistence:managed-persistence-context name="personDatabase"
   persistence-unit-jndi-name="java:/EntityManagerFactories/personDatabase">
   <persistence:filters>
      <value
>#{regionFilter}</value>
      <value
>#{currentFilter}</value>
   </persistence:filters>
</persistence:managed-persistence-context
>
```

Seam ## JSF

JSF ###

```
<h:form>
   <h:messages/>

   <div>
      Country:
      <h:inputText value="#{location.country}" required="true">
         <my:validateCountry/>
      </h:inputText>
   </div>

   <div>
      Zip code:
      <h:inputText value="#{location.zip}" required="true">
         <my:validateZip/>
      </h:inputText>
   </div>

   <h:commandButton/>
</h:form
>
```

########
DRY ########### Seam # Hibernate Validator ####################################

`Location` ######################

```
public class Location {
   private String country;
   private String zip;

   @NotNull
   @Length(max=30)
   public String getCountry() { return country; }
   public void setCountry(String c) { country = c; }

   @NotNull
   @Length(max=6)
   @Pattern("^\d*$")
   public String getZip() { return zip; }
```

```
    public void setZip(String z) { zip = z; }
}
```

############## #### Hibernate Validator ######################################

```
public class Location {
    private String country;
    private String zip;

    @NotNull
    @Country
    public String getCountry() { return country; }
    public void setCountry(String c) { country = c; }

    @NotNull
    @ZipCode
    public String getZip() { return zip; }
    public void setZip(String z) { zip = z; }
}
```

############### JSF ################################ ##### <s:validate> ##

```
<h:form>
    <h:messages/>

    <div>
        Country:
        <h:inputText value="#{location.country}" required="true">
            <s:validate/>
        </h:inputText>
    </div>

    <div>
        Zip code:
        <h:inputText value="#{location.zip}" required="true">
            <s:validate/>
        </h:inputText>
    </div>

    <h:commandButton/>
```

```
</h:form
>
```

##: ###### `@NotNull` ###################### `required="true"` ######################## JSF ####################

This approach *defines* constraints on the model, and *presents* constraint violations in the view — a significantly better design.

`<s:validateAll>`

```
<h:form>

   <h:messages/>

   <s:validateAll>

      <div>
         Country:
         <h:inputText value="#{location.country}" required="true"/>
      </div>

      <div>
         Zip code:
         <h:inputText value="#{location.zip}" required="true"/>
      </div>

      <h:commandButton/>

   </s:validateAll>

</h:form
>
```

######## `<s:validate>` ###################### ############### ############################

`label`

```
<h:inputText value="#{location.zip}" required="true" label="Zip:">
   <s:validate/>
```

```
</h:inputText
>
```

########### {0} ################################## (Hiberate Validator ##### JSF ######################)# ######################### (Internationalization) ###############

```
validator.length={0} ### {min} # {max} ############
```

########## ########################## (#### JSF ###)# ################## (######)# ############################ (######) ##### ### ## ########################

################################## ## #### facelets ######################

```
<ui:composition xmlns="http://www.w3.org/1999/xhtml"
                xmlns:ui="http://java.sun.com/jsf/facelets"
                xmlns:h="http://java.sun.com/jsf/html"
                xmlns:f="http://java.sun.com/jsf/core"
                xmlns:s="http://jboss.com/products/seam/taglib">

   <div>

      <s:label styleClass="#{invalid?'error':''}">
         <ui:insert name="label"/>
         <s:span styleClass="required" rendered="#{required}"
>*</s:span>
      </s:label>

      <span class="#{invalid?'error':''}">
         <h:graphicImage value="/img/error.gif" rendered="#{invalid}"/>
         <s:validateAll>
            <ui:insert/>
         </s:validateAll>
      </span>

      <s:message styleClass="error"/>

   </div>
```

```
</ui:composition
>
```

`<s:decorate>` ######################################

```
<h:form>

   <h:messages globalOnly="true"/>

   <s:decorate template="edit.xhtml">
      <ui:define name="label"
>Country:</ui:define>
      <h:inputText value="#{location.country}" required="true"/>
   </s:decorate>

   <s:decorate template="edit.xhtml">
      <ui:define name="label"
>Zip code:</ui:define>
      <h:inputText value="#{location.zip}" required="true"/>
   </s:decorate>

   <h:commandButton/>

</h:form
>
```

RichFaces Ajax

```
<h:form>

   <h:messages globalOnly="true"/>

   <s:decorate id="countryDecoration" template="edit.xhtml">
      <ui:define name="label"
>Country:</ui:define>
      <h:inputText value="#{location.country}" required="true">
         <a:support event="onblur" reRender="countryDecoration" bypassUpdates="true"/>
      </h:inputText>
   </s:decorate>

   <s:decorate id="zipDecoration" template="edit.xhtml">
      <ui:define name="label"
```

```
>Zip code:</ui:define>
      <h:inputText value="#{location.zip}" required="true">
         <a:support event="onblur" reRender="zipDecoration" bypassUpdates="true"/>
      </h:inputText>
   </s:decorate>

   <h:commandButton/>

</h:form
>
```

#################### ID #################### ## UI ######## Selenium ############################## #### ID ######## JSF ####################################

```
<h:form id="form">

   <h:messages globalOnly="true"/>

   <s:decorate id="countryDecoration" template="edit.xhtml">
      <ui:define name="label"
>Country:</ui:define>
      <h:inputText id="country" value="#{location.country}" required="true">
         <a:support event="onblur" reRender="countryDecoration" bypassUpdates="true"/>
      </h:inputText>
   </s:decorate>

   <s:decorate id="zipDecoration" template="edit.xhtml">
      <ui:define name="label"
>Zip code:</ui:define>
      <h:inputText id="zip" value="#{location.zip}" required="true">
         <a:support event="onblur" reRender="zipDecoration" bypassUpdates="true"/>
      </h:inputText>
   </s:decorate>

   <h:commandButton/>

</h:form
>
```

################################### Seam ######### (########## EL ############################) # Hibernate Validator ##############

```
public class Location {
    private String name;
    private String zip;

    // Getters and setters for name

    @NotNull
    @Length(max=6)
    @ZipCode(message="#{messages['location.zipCode.invalid']}")
    public String getZip() { return zip; }
    public void setZip(String z) { zip = z; }
}
```

```
location.zipCode.invalid = #{location.name} ###################
```

Groovy

JBoss Seam############RAD######################RAD##
API##API####################################
Java #############################*Groovy* [http://groovy.codehaus.org]#######################

JBoss Seam #########################Java EE ######################JBoss Seam ###Seam #################Seam ##################################API##########################

11.1.

Groovy ##Java######################Python#Ruby#Smalltalk######################Groovy #########2####

- Java ##################Groovy#####Java####################
- Groovy ############Java##########Java###########################

TODO:Groovy ################

11.2. Groovy ### Seam

########################Groovy###### # Java ############Seam ###################Groovy###############################Groovy#####Java##################

11.2.1. Groovy

##############Seam######################################Groovy 1.1 ####################Seam###############Groovy########

11.2.1.1.

```
@Entity
  @Name("hotel")
  class Hotel implements Serializable
  {
    @Id @GeneratedValue
    Long id

    @Length(max=50) @NotNull
    String name
```

```
    @Length(max=100) @NotNull
    String address

    @Length(max=40) @NotNull
    String city

    @Length(min=2, max=10) @NotNull
    String state

    @Length(min=4, max=6) @NotNull
    String zip

    @Length(min=2, max=40) @NotNull
    String country

    @Column(precision=6, scale=2)
    BigDecimal price

    @Override
    String toString()
    {
        return "Hotel(${name},${address},${city},${zip})"
    }
}
```

Groovy###########(getter/
setter)#######################getter#setter#########################hotel####Java##hotel.getCity()######

11.2.1.2. Seam

Seam########Groovy########Java############Seam#####################################

```
@Scope(ScopeType.SESSION)
@Name("bookingList")
class BookingListAction implements Serializable
{
    @In EntityManager em
    @In User user
    @DataModel List<Booking> bookings
    @DataModelSelection Booking booking
    @Logger Log log

    @Factory public void getBookings()
```

```
    {
        bookings = em.createQuery("'
                select b from Booking b
                where b.user.username = :username
                order by b.checkinDate"')
            .setParameter("username", user.username)
            .getResultList()
    }

    public void cancel()
    {
        log.info("Cancel booking: #{bookingList.booking.id} for #{user.username}")
        Booking cancelled = em.find(Booking.class, booking.id)
        if (cancelled != null) em.remove( cancelled )
        getBookings()
                    FacesMessages.instance().add("Booking cancelled for confirmation number
 #{bookingList.booking.id}", new Object[0])
    }
}
```

11.2.2. seam-gen

seam-gen #Groovy###########seam-gen
################################Groovy###########Groovy################.groovy#####src/
main###################################.groovy#####src/hot########

11.3.

Groovy##########Java###3###########

###################JavaBeans#Seam####################JBoss
Seam#################################/
##########################.groovy################GroovyBeans#Seam####################

11.3.1. Groovy

Groovy########Java###################Java##########Groovy#######Groovy#####bean#Groovy
Seam##groovyc
ant####################Groovy####Java###Groov

11.3.2. ######### .groovy

JBoss
Seam##############################.groovy###

#Seam ######################Groovy###(.groovy ####)#WEB-INF/dev###################################GroovyBean##

#####.groovy###############Seam############################

- ##########JavaBeans #### GroovyBeans###EJB3 bean############
- ####################
- ################### WEB-INF/dev ######################
- Seam ####################

11.3.3. seam-gen

Seam-gen #Groovy###.groovy################src/hot#WAR, Java, Groovy#########seam-gen###Groovy###############

examples/groovybooking###Groovy#######################################

######################Apache Wicket#####

Seam#JSF######################Wicket#########Seam######wicket#########################Wick

##

Wicket#####Seam#################JSF#################Wicket#####################

12.1. Seam##Wicket###############

Wicket###

Wicket###Seam#######

##this()#super()#####################################

##

######################

##

12.1.1.

Seam######Wicket###################Seam#######EVENT#CONVERSATION#SESSION#APPLICATION#BUSINES

Wicket##Seam######################@In######################

```
@In(create=true)
private HotelBooking hotelBooking;
```

####

Wicket#######Seam######################@Name##############

Wicket##########Seam###################################

```
@Out(scope=ScopeType.EVENT, required=false)
private String verify;
```

TODO ###################

12.1.2.

Wicket############@Restrict###15.############@Rest
###############

#:

```
@Restrict
public class Main extends WebPage {

  ...
```

####

Seam#################################

Wicket############@Begin#@End###Seam################

##

ifOutcome##############

#:

```
item.add(new Link("viewHotel") {

  @Override
  @Begin
  public void onClick() {
    hotelBooking.selectHotel(hotel);
    setResponsePage(org.jboss.seam.example.wicket.Hotel.class);
  }
};
```

###@NoConversationPage########

```
@Restrict
@NoConversationPage(Main.class)
```

```
public class Hotel extends WebPage {
```

############################Seam####################Events.instance().raiseEvent("foo")####
(outcome)############################

Wicket######@CreateProcess#@ResumeTask#@BeginTask#@EndTask#@StartTask#@Transition############

TODO - BPM####### - JBSEAM-3194

12.2.

Seam needs to instrument the bytecode of your Wicket classes to be able to intercept the annotations you use. The first decision to make is: do you want your code instrumented at runtime as your app is running, or at compile time? The former requires no integration with your build environment, but has a performance penalty when loading each instrumented class for the first time. The latter is faster, but requires you to integrate this instrumentation into your build environment.

12.2.1. Runtime instrumentation

There are two ways to achieve runtime instrumentation. One relies on placing wicket components to be instrumented in a special folder in your WAR deployment. If this is not acceptable or possible, you can also use an instrumentation "agent," which you specify in the command line for launching your container.

12.2.1.1. Location-specific instrumentation

Any classes placed in the `WEB-INF/wicket` folder within your WAR deployment will be automatically instrumented by the seam-wicket runtime. You can arrange to place your wicket pages and components here by specifying a separate output folder for those classes in your IDE, or through the use of ant scripts.

12.2.1.2. Runtime instrumentation agent

The jar file `jboss-seam-wicket.jar` can be used as an iinstrumentation agent through the Java Instrumentation api. This is accomplished through the following steps:

- Arrange for the `jboss-seam-wicket.jar` file to live in a location for which you have an absolute path, as the Java Instrumentation API does not allow relative paths when specifying the location of an agent lib.
- Add `javaagent:/path/to/jboss-seam-wicket.jar` to the command line options when launching your webapp container:
- In addition, you will need to add an environment variable that specifies packages that the agent should instrument. This is accomplished by a comma separated list of package names:

```
-Dorg.jboss.seam.wicket.instrumented-packages=my.package.one,my.other.package
```

Note that if a package A is specified, classes in subpackages of A are also examined. The classes chosen for instrumentation can be further limited by specifying:

```
-Dorg.jboss.seam.wicket.scanAnnotations=true
```

and then marking instrumentable classes with the `@SeamWicketComponent` annotation, see *#12.2.3. #The `@SeamWicketComponent` annotation#*.

12.2.2. Compile-time instrumentation

Seam supports instrumentation at compile time through either Apache Ant or Apache Maven.

12.2.2.1. Instrumenting with ant

Seam provides an ant task in the `jboss-seam-wicket-ant.jar` . This is used in the following manner:

```
<taskdef name="instrumentWicket"
  classname="org.jboss.seam.wicket.ioc.WicketInstrumentationTask">
 <classpath>
  <pathelement location="lib/jboss-seam-wicket-ant.jar"/>
  <pathelement location="web/WEB-INF/lib/jboss-seam-wicket.jar"/>
  <pathelement location="lib/javassist.jar"/>
  <pathelement location="lib/jboss-seam.jar"/>
 </classpath>
</taskdef>

<instrumentWicket outputDirectory="${build.instrumented}" useAnnotations="true">
 <classpath refid="build.classpath"/>
 <fileset dir="${build.classes}" includes="**/*.class"/>
</instrumentWicket>
```

This results in the instrumented classes being placed in the directory specified by `${build.instrumented}`. You will then need to instruct ant to copy these classes into `WEB-INF/classes`. If you want to hot deploy the Wicket components, you can copy the instrumented classes to `WEB-INF/dev`; if you use hot deploy, make sure that your `WicketApplication` class is also hot-deployed. Upon a reload of hot-deployed classes, the entire WicketApplication instance has to be re-initialized, in order to pick up new references to the classes of mounted pages.

The useAnnotations attribute is used to make the ant task only include classes that have been marked with the @SeamWicketComponent annotation, see *#12.2.3. #The @SeamWicketComponent annotation#*.

12.2.2.2. Instrumenting with maven

The jboss maven repository repository.jboss.org provides a plugin named seam-instrument-wicket with a process-classes mojo. An example configuration in your pom.xml might look like:

```
<build>
        <plugins>
         <plugin>
            <groupId>org.jboss.seam</groupId>
            <artifactId>seam-instrument-wicket</artifactId>
            <version>2.2.0</version>
            <configuration>
                <scanAnnotations>true</scanAnnotations>
                <includes>
                    <include>your.package.name</include>
                </includes>
            </configuration>
            <executions>
                <execution>
                    <id>instrument</id>
                    <phase>process-classes</phase>
                    <goals>
                        <goal>instrument</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

The above example illustrates that the instrumentation is limited to classes specified by the includes element. In this example, the scanAnnotations is specified, see *#12.2.3. #The @SeamWicketComponent annotation#*.

12.2.3. The @SeamWicketComponent annotation

Classes placed in WEB-INF/wicket will unconditionally be instrumented. The other instrumentation mechanisms all allow you to specify that instrumentation should only be applied to classes

annotated with the `@SeamWicketComponent` annotation. This annotation is inherited, which means all subclasses of an annotated class will also be instrumented. An example usage is:

```
import org.jboss.seam.wicket.ioc.SeamWicketComponent;
@SeamWicketComponent
public class MyPage extends WebPage{
 ...
}
```

12.2.4.

Seam############Wicket#WebApplication####`SeamWebApplication`####################Wicket#########

#:

`SeamAuthorizationStrategy`##Wicket#########`@Restrict`#########Seam####################`SeamWeb`

#######`getHomePage()`####################################

```
public class WicketBookingApplication extends SeamWebApplication {

  @Override
  public Class getHomePage() {
    return Home.class;
  }

  @Override
  protected Class getLoginPage() {
    return Home.class;
  }

}
```

Seam automatically installs the Wicket filter for you (ensuring that it is inserted in the correct place for you). But you still need to tell Wicket which `WebApplication` class to use.

```
<components xmlns="http://jboss.com/products/seam/components"
 xmlns:wicket="http://jboss.com/products/seam/wicket"
 xsi:schemaLocation=
  "http://jboss.com/products/seam/wicket
   http://jboss.com/products/seam/wicket-2.2.xsd">
```

```
  <wicket:web-application
    application-class="org.jboss.seam.example.wicket.WicketBookingApplication" />
</components
```

In addition, if you plan to use JSF-based pages in the same application as wicket pages, you'll need to ensure that the jsf exception filter is only enabled for jsf urls:

```
<components xmlns="http://jboss.com/products/seam/components"
 xmlns:web="http://jboss.com/products/seam/web"
 xmlns:wicket="http://jboss.com/products/seam/wicket"
 xsi:schemaLocation=
  "http://jboss.com/products/seam/web
   http://jboss.com/products/seam/web-2.2.xsd">

   <!-- Only map the seam jsf exception filter to jsf paths, which we identify with the *.seam path -->
       <web:exception-filter url-pattern="*.seam"/>
</components
```

####

##################Application##############################Wicket#####

Seam################

Seam ############################## ### Java ### #### `components.xml`### (#######) ## ##

Seam################ JPA #Hibernae ##

We should emphasize that the framework is extremely simple, just a handful of simple classes that are easy to understand and extend. The "magic" is in Seam itself — the same magic you use when creating any Seam application even without using this framework.

13.1.

Seam ############################## ######################## ####### ## Seam ###################### `components.xml` ################################ ##### ### `components.xml` ####### `Person` ############## CRUD ######## 1 ####################

```
<framework:entity-home name="personHome"
                       entity-class="eg.Person"
                       entity-manager="#{personDatabase}">
   <framework:id
>#{param.personId}</framework:id>
</framework:entity-home
>
```

####XML ########################## ########################

```
@Name("personHome")
public class PersonHome extends EntityHome<Person
> {

   @In EntityManager personDatabase;

   public EntityManager getEntityManager() {
     return personDatabase;
   }

}
```

(########)
(##)

EJB ############ Bean
(##EJB##################JavaBean###########) JBoss AS
#########4.2.2.GA####################

```
@Stateful
@Name("personHome")
public class PersonHome extends EntityHome<Person
> implements LocalPersonHome {

}
```

################ Bean ############ ##### ##### `entityManager`
##

```
@Stateless
@Name("personHome")
public class PersonHome extends EntityHome<Person
> implements LocalPersonHome {

  @In EntityManager entityManager;

  public EntityManager getPersistenceContext() {
    entityManager;
  }

}
```

#####Seam#################CRUD ##`EntityHome`
`#HibernateEntityHome`####Query####`EntityQuery` #
`HibernateEntityQuery`#4#########################

Home#Query#####################################
####################################

Seam################Seam#########################
#######`entityManager`####################

13.2. Home######

Home####################################`Person`################

```
@Entity
public class Person {
   @Id private Long id;
   private String firstName;
   private String lastName;
   private Country nationality;

   //getters and setters...
}
```

##############personHome####################

```
<framework:entity-home name="personHome" entity-class="eg.Person" />
```

############################

```
@Name("personHome")
public class PersonHome extends EntityHome<Person
> {}
```

Home ####### persist()# remove()# update()# getInstance() # ##############
remove()##### update() ######### setId() ###################################

Home # JSF ##########################

```
<h1
>Create Person</h1>
<h:form>
   <div
>First name: <h:inputText value="#{personHome.instance.firstName}"/></div>
   <div
>Last name: <h:inputText value="#{personHome.instance.lastName}"/></div>
   <div>
      <h:commandButton value="Create Person" action="#{personHome.persist}"/>
   </div>
</h:form
>
```

###Person#person###############components.xml#######################

```
<factory name="person"
        value="#{personHome.instance}"/>

<framework:entity-home name="personHome"
                    entity-class="eg.Person" />
```

(################) `PersonHome` # `@Factory` #######

```
@Name("personHome")
public class PersonHome extends EntityHome<Person
> {

   @Factory("person")
   public Person initPerson() { return getInstance(); }

}
```

(###########) ######### JSF ###############

```
<h1
>Create Person</h1>
<h:form>
   <div
>First name: <h:inputText value="#{person.firstName}"/></div>
   <div
>Last name: <h:inputText value="#{person.lastName}"/></div>
   <div>
      <h:commandButton value="Create Person" action="#{personHome.persist}"/>
   </div>
</h:form
>
```

`####Person############################` ############
############################
`Person################PersonHome############################`
############################

```
<pages>
   <page view-id="/editPerson.jsp">
```

```
    <param name="personId" value="#{personHome.id}"/>
  </page>
</pages
>
```

####JSF#######################

```
<h1>
  <h:outputText rendered="#{!personHome.managed}" value="Create Person"/>
  <h:outputText rendered="#{personHome.managed}" value="Edit Person"/>
</h1>
<h:form>
  <div
>First name: <h:inputText value="#{person.firstName}"/></div>
  <div
>Last name: <h:inputText value="#{person.lastName}"/></div>
  <div>
                <h:commandButton value="Create Person" action="#{personHome.persist}"
 rendered="#{!personHome.managed}"/>
                <h:commandButton value="Update Person" action="#{personHome.update}"
 rendered="#{personHome.managed}"/>
                <h:commandButton value="Delete Person" action="#{personHome.remove}"
 rendered="#{personHome.managed}"/>
  </div>
</h:form
>
```

########################Person###############`personId`
####################Person#################

`Person` ##### nationality ### ##############

```
<factory name="person"
      value="#{personHome.instance}"/>

<framework:entity-home name="personHome"
                entity-class="eg.Person"
                new-instance="#{newPerson}"/>

<component name="newPerson"
       class="eg.Person">
  <property name="nationality"
```

```
>#{country}</property>
</component
>
```

########################

```
@Name("personHome")
public class PersonHome extends EntityHome<Person
> {

   @In Country country;

   @Factory("person")
   public Person initPerson() { return getInstance(); }

   protected Person createInstance() {
      return new Person(country);
   }

}
```

#####Country##########CountryHome##### Home ############################

############## ################ PersonHome ########################

```
@Name("personHome")
public class PersonHome extends EntityHome<Person
> {

   @In Country country;

   @Factory("person")
   public Person initPerson() { return getInstance(); }

   protected Person createInstance() {
      return new Person(country);
   }

   public void migrate()
   {
      getInstance().setCountry(country);
      update();
```

```
   }

}
```

The Home object raises an `org.jboss.seam.afterTransactionSuccess` event when a transaction succeeds (a call to `persist()`, `update()` or `remove()` succeeds). By observing this event we can refresh our queries when the underlying entities are changed. If we only want to refresh certain queries when a particular entity is persisted, updated or removed we can observe the `org.jboss.seam.afterTransactionSuccess.<name>` event (where `<name>` is the simple name of the entity, e.g. an entity called "org.foo.myEntity" has "myEntity" as simple name).

Home###################################### ##############################

```
<factory name="person"
        value="#{personHome.instance}"/>

<framework:entity-home name="personHome"
                    entity-class="eg.Person"
                    new-instance="#{newPerson}">
   <framework:created-message
>New person #{person.firstName} #{person.lastName} created</framework:created-message>
   <framework:deleted-message
>Person #{person.firstName} #{person.lastName} deleted</framework:deleted-message>
   <framework:updated-message
>Person #{person.firstName} #{person.lastName} updated</framework:updated-message>
</framework:entity-home>

<component name="newPerson"
         class="eg.Person">
   <property name="nationality"
>#{country}</property>
</component
>
```

############################

```
@Name("personHome")
public class PersonHome extends EntityHome<Person
> {

   @In Country country;
```

```
   @Factory("person")
   public Person initPerson() { return getInstance(); }

   protected Person createInstance() {
      return new Person(country);
   }

      protected String getCreatedMessage() { return createValueExpression("New person
#{person.firstName} #{person.lastName} created"); }
         protected String getUpdatedMessage() { return createValueExpression("Person
#{person.firstName} #{person.lastName} updated"); }
         protected String getDeletedMessage() { return createValueExpression("Person
#{person.firstName} #{person.lastName} deleted"); }

}
```

###################### (###### `messages` ######) Seam ############################

```
Person_created=New person #{person.firstName} #{person.lastName} created
Person_deleted=Person #{person.firstName} #{person.lastName} deleted
Person_updated=Person #{person.firstName} #{person.lastName} updated
```

###

########<s:validateAll>
#<s:decorate>###

13.3. Query######

########`Person`#####################Query##########################

```
<framework:entity-query name="people"
                        ejbql="select p from Person p"/>
```

############JSF###############

```
<h1
>List of people</h1>
<h:dataTable value="#{people.resultList}" var="person">
   <h:column>
      <s:link view="/editPerson.jsp" value="#{person.firstName} #{person.lastName}">
```

```
        <f:param name="personId" value="#{person.id}"/>
      </s:link>
   </h:column>
</h:dataTable
>
```

##############################

```
<framework:entity-query name="people"
                        ejbql="select p from Person p"
                        order="lastName"
                        max-results="20"/>
```

##############################

```
<pages>
   <page view-id="/searchPerson.jsp">
      <param name="firstResult" value="#{people.firstResult}"/>
   </page>
</pages
>
```

##########JSF######################

```
<h1
>Search for people</h1>
<h:dataTable value="#{people.resultList}" var="person">
   <h:column>
      <s:link view="/editPerson.jsp" value="#{person.firstName} #{person.lastName}">
         <f:param name="personId" value="#{person.id}"/>
      </s:link>
   </h:column>
</h:dataTable>

<s:link view="/search.xhtml" rendered="#{people.previousExists}" value="First Page">
   <f:param name="firstResult" value="0"/>
</s:link>

<s:link view="/search.xhtml" rendered="#{people.previousExists}" value="Previous Page">
   <f:param name="firstResult" value="#{people.previousFirstResult}"/>
```

```
</s:link>

<s:link view="/search.xhtml" rendered="#{people.nextExists}" value="Next Page">
   <f:param name="firstResult" value="#{people.nextFirstResult}"/>
</s:link>

<s:link view="/search.xhtml" rendered="#{people.nextExists}" value="Last Page">
   <f:param name="firstResult" value="#{people.lastFirstResult}"/>
</s:link
>
```

##
#######################Query################################

```
<component name="examplePerson" class="Person"/>

<framework:entity-query name="people"
                        ejbql="select p from Person p"
                        order="lastName"
                        max-results="20">
   <framework:restrictions>
      <value
>lower(firstName) like lower( concat(#{examplePerson.firstName},'%') )</value>
      <value
>lower(lastName) like lower( concat(#{examplePerson.lastName},'%') )</value>
   </framework:restrictions>
</framework:entity-query
>
```

######example#######################

```
<h1
>Search for people</h1>
<h:form>
   <div
>First name: <h:inputText value="#{examplePerson.firstName}"/></div>
   <div
>Last name: <h:inputText value="#{examplePerson.lastName}"/></div>
   <div
><h:commandButton value="Search" action="/search.jsp"/></div>
</h:form>
```

```
<h:dataTable value="#{people.resultList}" var="person">
   <h:column>
      <s:link view="/editPerson.jsp" value="#{person.firstName} #{person.lastName}">
         <f:param name="personId" value="#{person.id}"/>
      </s:link>
   </h:column>
</h:dataTable
>
```

############################## `org.jboss.seam.afterTransactionSuccess` ###########

```
<event type="org.jboss.seam.afterTransactionSuccess">
   <action execute="#{people.refresh}" />
</event
>
```

`PersonHome` # person ########### ###

```
<event type="org.jboss.seam.afterTransactionSuccess.Person">
   <action execute="#{people.refresh}" />
   </event
>
```

Unfortunately Query objects don't work well with *join fetch* queries - the use of pagination with these queries is not recommended, and you'll have to implement your own method of calculating the total number of results (by overriding `getCountEjbql()`.

##############################Query##############################

13.4. Controller######

Seam##############################`Controller`############## `EntityController`# `HibernateEntityController` #`BusinessProcessController`###### ##Seam###############

#####Seam#Registration###`RegisterAction`#Seam##################################

```
@Stateless
@Name("register")
public class RegisterAction extends EntityController implements Register
{
```

```
  @In private User user;

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

    if ( existing.size()==0 )
    {
      persist(user);
      info("Registered new user #{user.username}");
      return "/registered.jspx";
    }
    else
    {
      addFacesMessage("User #{user.username} already exists");
      return null;
    }
  }

}
```

############################

Seam # JBoss Rules

Seam ###Seam ########## jBPM ######## JBoss Rules (Drools) # RuleBase ##########

14.1.

#########Seam ######### `org.drools.RuleBase` ################### #######Seam ## ########## `components.xml` ####################

```
<drools:rule-base name="policyPricingRules">
   <drools:rule-files>
      <value
>policyPricingRules.drl</value>
   </drools:rule-files>
</drools:rule-base
>
```

This component compiles rules from a set of DRL (`.drl`) or decision table (`.xls`) files and caches an instance of `org.drools.RuleBase` in the Seam `APPLICATION` context. Note that it is quite likely that you will need to install multiple rule bases in a rule-driven application.

Drools DSL########## DSL ################

```
<drools:rule-base name="policyPricingRules" dsl-file="policyPricing.dsl">
   <drools:rule-files>
      <value
>policyPricingRules.drl</value>
   </drools:rule-files>
</drools:rule-base
>
```

Support for Drools RuleFlow is also available and you can simply add a `.rf` or a `.rfm` as part of your rule files as:

```
            <drools:rule-base  name="policyPricingRules"  rule-files="policyPricingRules.drl,
policyPricingRulesFlow.rf"/>
```

Note that when using the Drools 4.x RuleFlow (`.rfm`) format, you need to specify the -Ddrools.ruleflow.port=true system property on server startup. This is however still an experimental feature and we advise to use the Drools5 (`.rf`) format if possible.

If you want to register a custom consequence exception handler through the RuleBaseConfiguration, you need to write the handler, for example:

```
@Scope(ScopeType.APPLICATION)
@Startup
@Name("myConsequenceExceptionHandler")
public class MyConsequenceExceptionHandler implements ConsequenceExceptionHandler,
 Externalizable {

  public void readExternal(ObjectInput in) throws IOException, ClassNotFoundException {
  }

  public void writeExternal(ObjectOutput out) throws IOException {
  }

  public void handleException(Activation activation,
                              WorkingMemory workingMemory,
                              Exception exception) {
    throw new ConsequenceException( exception,
                                    activation.getRule() );
  }

}
```

and register it:

```
<drools:rule-base name="policyPricingRules" dsl-file="policyPricing.dsl" consequence-exception-handler="#{myConsequenceExceptionHandler}">
   <drools:rule-files>
      <value>policyPricingRules.drl</value>
   </drools:rule-files>
</drools:rule-base>
```

########## ########## RuleBase #### Drools RuleAgent ######### RuleAgent # Drools ######## (BRMS) ########## RulesAgent ### RuleBase # `components.xml` #########

```
<drools:rule-agent name="insuranceRules"
                   configurationFile="/WEB-INF/deployedrules.properties" />
```

############ RulesAgent ################## Drools ##################################

```
newInstance=true
url=http://localhost:8080/drools-jbrms/org.drools.brms.JBRMS/package/org.acme.insurance/fmeyer
localCacheDir=/Users/fernandomeyer/projects/jbossrules/drools-examples/drools-examples-brms/cache
poll=30
name=insuranceconfig
```

######################################

```
<drools:rule-agent name="insuranceRules"
    url="http://localhost:8080/drools-jbrms/org.drools.brms.JBRMS/package/org.acme.insurance/fmeyer"
                   local-cache-dir="/Users/fernandomeyer/projects/jbossrules/drools-examples/drools-examples-brms/cache"
   poll="30"
   configuration-name="insuranceconfig" />
```

########## `org.drools.WorkingMemory` ##################### (# `WorkingMemory` ############ fact #######)

```
<drools:managed-working-memory name="policyPricingWorkingMemory" auto-create="true"
 rule-base="#{policyPricingRules}"/>
```

`policyPricingWorkingMemory` ## `ruleBase` ########### RuleBase ######################

We can also add means to be notified of rule engine events, including rules firing, objects being asserted, etc. by adding event listeners to WorkingMemory.

```
<drools:managed-working-memory name="policyPricingWorkingMemory" auto-create="true"
 rule-base="#{policyPricingRules}">
      <drools:event-listeners>
```

```
        <value>org.drools.event.DebugWorkingMemoryEventListener</value>
      <value>org.drools.event.DebugAgendaEventListener</value>
      </drools:event-listeners>
</drools:managed-working-memory>
```

14.2. Seam

`WorkingMemory` ## ### Seam ################ fact ##########################

```
@In WorkingMemory policyPricingWorkingMemory;

@In Policy policy;
@In Customer customer;

public void pricePolicy() throws FactException
{
      policyPricingWorkingMemory.insert(policy);
      policyPricingWorkingMemory.insert(customer);
      // if we have a ruleflow, start the process
      policyPricingWorkingMemory.startProcess(startProcessId)
   policyPricingWorkingMemory.fireAllRules();
}
```

14.3. jBPM

You can even allow a rule base to act as a jBPM action handler, decision handler, or assignment handler — in either a pageflow or business process definition.

```
<decision name="approval">

   <handler class="org.jboss.seam.drools.DroolsDecisionHandler">
      <workingMemoryName>orderApprovalRulesWorkingMemory</workingMemoryName>
      <!-- if a ruleflow was added -->
      <startProcessId>approvalruleflowid</startProcessId>
      <assertObjects>
         <element>#{customer}</element>
         <element>#{order}</element>
         <element>#{order.lineItems}</element>
      </assertObjects>
   </handler>
```

```
    <transition name="approved" to="ship">
      <action class="org.jboss.seam.drools.DroolsActionHandler">
        <workingMemoryName>shippingRulesWorkingMemory</workingMemoryName>
        <assertObjects>
          <element>#{customer}</element>
          <element>#{order}</element>
          <element>#{order.lineItems}</element>
        </assertObjects>
      </action>
    </transition>

    <transition name="rejected" to="cancelled"/>

</decision>
```

`<assertObjects>` ###### `WorkingMemory` # fact ######################## 1 ######### EL ########

The `<retractObjects>` element on the other hand specifies EL expressions that return an object or collection of objects to be retracted from the `WorkingMemory`.

jBPM ############ Drools ###############

```
<task-node name="review">
    <task name="review" description="Review Order">
      <assignment handler="org.jboss.seam.drools.DroolsAssignmentHandler">
        <workingMemoryName
>orderApprovalRulesWorkingMemory</workingMemoryName>
        <assertObjects>
          <element
>#{actor}</element>
          <element
>#{customer}</element>
          <element
>#{order}</element>
          <element
>#{order.lineItems}</element>
        </assertObjects>
      </assignment>
    </task>
    <transition name="rejected" to="cancelled"/>
    <transition name="approved" to="approved"/>
</task-node
```

```
>
```

################ Drools ################ jBPM Assignable # assignable #### Seam Decision ####### decision ########## decision ######### decision.setOutcome"result") ################ assignment ######### Assignable ##########ID##########

```
package org.jboss.seam.examples.shop

import org.jboss.seam.drools.Decision

global Decision decision

rule "Approve Order For Loyal Customer"
  when
    Customer( loyaltyStatus == "GOLD" )
    Order( totalAmount <= 10000 )
  then
    decision.setOutcome("approved");
end
```

```
package org.jboss.seam.examples.shop

import org.jbpm.taskmgmt.exe.Assignable

global Assignable assignable

rule "Assign Review For Small Order"
  when
    Order( totalAmount <= 100 )
  then
    assignable.setPooledActors( new String[] {"reviewers"} );
end
```

##

Drools ##### http://www.drools.org ##########

##

Seam ###################### Drools ############## Drools ##

######

15.1.

Seam#######API#Seam#################################
########################

- ######################################JAAS##############
- ID###Seam##########################API######
- ##
- ##Seam############
- CAPCHA######Seam###
- ##

########################

15.2.

In some situations it may be necessary to disable Seam Security, for instances during unit tests or because you are using a different approach to security, such as native JAAS. Simply call the static method `Identity.setSecurityEnabled(false)` to disable the security infrastructure. Of course, it's not very convenient to have to call a static method when you want to configure the application, so as an alternative you can control this setting in components.xml:

- ##############
- Hibernate#############
- Seam#############
- ########
- Servlet API security integration

Assuming you are planning to take advantage of what Seam Security has to offer, the rest of this chapter documents the plethora of options you have for giving your user an identity in the eyes of the security model (authentication) and locking down the application by establishing constraints (authorization). Let's begin with the task of authentication since that's the foundation of any security model.

15.3.

Seam ################ JAAS (Java Authentication and Authorization Service) ############ ######################## API ######### ####### Seam# # JAAS ################################

15.3.1.

##

########Seam#ID################authenticator#####################################

#########Seam#######################SeamLoginModule#####Seam########JAAS##################
############Seam###
############################
####################components.xml#######identity####################

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:core="http://jboss.com/products/seam/core"
        xmlns:security="http://jboss.com/products/seam/security"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation=
                "http://jboss.com/products/seam/components  http://jboss.com/products/seam/
components-2.2.xsd
                http://jboss.com/products/seam/security  http://jboss.com/products/seam/security-
2.2.xsd">

   <security:identity authenticate-method="#{authenticator.authenticate}"/>

</components>
```

EL#   #{authenticator.authenticate}
#authenticator########authenticate###

15.3.2.

components.xml   ##identity#authenticate-
method######SeamLoginModule#####################
##############################boolean#######
#####username#password#Credentials.getUsername()   #
Credentials.getPassword()################Identity.instance().getCredentials()##credentials#####
#####################Identity.addRole()################
###POJO##########################

```
@Name("authenticator")
public class Authenticator {
   @In EntityManager entityManager;
```

```
  @In Credentials credentials;
  @In Identity identity;

  public boolean authenticate() {
    try {
      User user = (User) entityManager.createQuery(
        "from User where username = :username and password = :password")
        .setParameter("username", credentials.getUsername())
        .setParameter("password", credentials.getPassword())
        .getSingleResult();

      if (user.getRoles() != null) {
        for (UserRole mr : user.getRoles())
          identity.addRole(mr.getName());
      }

      return true;
    }
    catch (NoResultException ex) {
      return false;
    }

  }

}
```

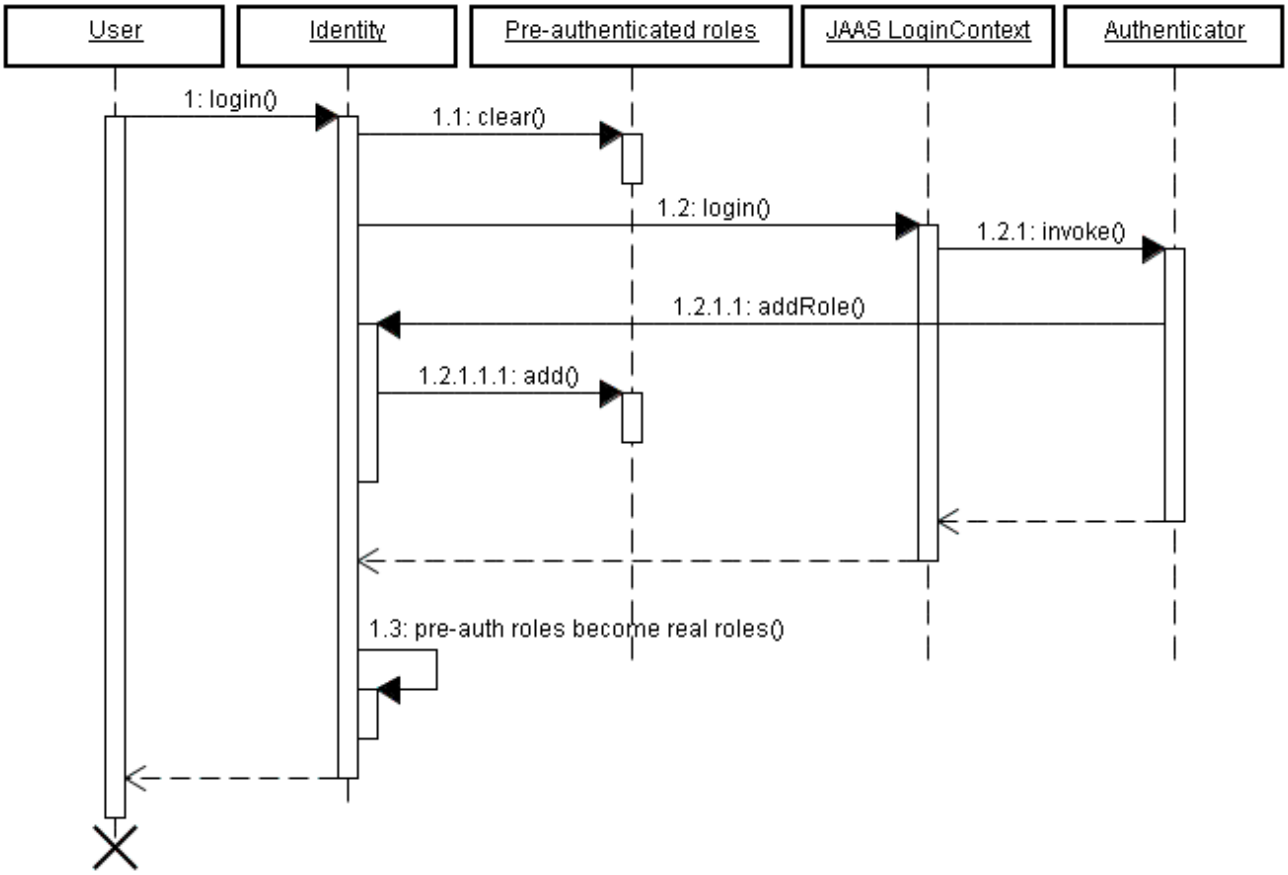
#######User#UserRole###################Bean######## ##### roles #
"admin", "user" ##########Set################
#######user#######NoResultException###################false####################

####

##
##########API###
##
Seam###

15.3.2.1. Identity.addRole()

Identity.addRole()###################################
####################addRole()###
##



########################Identity.addRole()################################

15.3.2.2.

##
####org.jboss.seam.security.loginSuccessful#################################

```
@In UserStats userStats;

@Observer("org.jboss.seam.security.loginSuccessful")
public void updateUserStats()
{
  userStats.setLastLoginDate(new Date());
  userStats.incrementLoginCount();
}
```

#############Authenticator#########################
################################

15.3.3.

`credentials` ########username###password#################################
################username#password###################
###################`identity.login()`##################username#password#####################

```
<div>
   <h:outputLabel for="name" value="Username"/>
   <h:inputText id="name" value="#{credentials.username}"/>
</div>

<div>
   <h:outputLabel for="password" value="Password"/>
   <h:inputSecret id="password" value="#{credentials.password}"/>
</div>

<div>
   <h:commandButton value="Login" action="#{identity.login}"/>
</div
>
```

#login#############`#{identity.logout}`###

15.3.4.

##

- #######`components.xml`######
- ##########
- ############################

15.3.5. Remember Me

Seam##################WEB####################"Remember
me"(######################
##############################username#############################password###################

#2###password#####################################

##

############################2###
################XSS###

```
            ##################################################################
            ####################Remember me################################### ###

            ########################################Remember
            Password############
            ##############################username#password####################
            ########################################Remember
            Me############################                                    OS-
            x#Safari#############OS################################
            ##################################################################

            ################Remember
            Me############################################
            ######username##############################
```

######Remember me(username############################## ##################remember me#################rememberMe.enabled############

```
 <div>
   <h:outputLabel for="name" value="User name"/>
   <h:inputText id="name" value="#{credentials.username}"/>
 </div>

 <div>
   <h:outputLabel for="password" value="Password"/>
   <h:inputSecret id="password" value="#{credentials.password}" redisplay="true"/>
 </div
>

 <div class="loginRow">
   <h:outputLabel for="rememberMe" value="Remember me"/>
   <h:selectBooleanCheckbox id="rememberMe" value="#{rememberMe.enabled}"/>
 </div
>
```

15.3.5.1. ########remember me##

################ remember me ################################## Seam## ####org.jboss.seam.security.TokenStore ################################## ################JpaTokenStore##

################################ ######################

```
@Entity
public class AuthenticationToken implements Serializable {
  private Integer tokenId;
  private String username;
  private String value;

  @Id @GeneratedValue
  public Integer getTokenId() {
    return tokenId;
  }

  public void setTokenId(Integer tokenId) {
    this.tokenId = tokenId;
  }

  @TokenUsername
  public String getUsername() {
    return username;
  }

  public void setUsername(String username) {
    this.username = username;
  }

  @TokenValue
  public String getValue() {
    return value;
  }

  public void setValue(String value) {
    this.value = value;
  }
}
```

######################username##############@TokenUsername
#@TokenValue###

############Bean###################### JpaTokenStore####### #### components.xml
#token-class#################

```
<security:jpa-token-store                                    token-
class="org.jboss.seam.example.seamspace.AuthenticationToken" />
```

```
```

##############components.xml   #RememberMe#################
mode#autoLogin#################

```
<security:remember-me mode="autoLogin"/>
```

####remember me##

####################################components.xml#################

```
<event type="org.jboss.seam.security.notLoggedIn">
   <action execute="#{redirect.captureCurrentView}"/>
   <action execute="#{identity.tryLogin()}"/>
  </event>
  <event type="org.jboss.seam.security.loginSuccessful">
   <action execute="#{redirect.returnToCapturedView}"/>
  </event
>
```

15.3.6.

##pages.xml###
######API#########################

- NotLoggedInException - ##
- AuthorizationException   -
  ##

NotLoggedInException###
###AuthorizationException#################################
#####################################pages.xml########

```
<pages>

   ...

   <exception class="org.jboss.seam.security.NotLoggedInException">
```

```
        <redirect view-id="/login.xhtml">
            <message
>You must be logged in to perform this action</message>
        </redirect>
    </exception>

    <exception class="org.jboss.seam.security.AuthorizationException">
        <end-conversation/>
        <redirect view-id="/security_error.xhtml">
            <message
>You do not have the necessary security privileges to perform this action.</message>
        </redirect>
    </exception>

</pages
>
```

#####web#####################################Seam##############################

15.3.7.

###Seam##############################
(`pages.xml`#) ##########

```
<pages login-view-id="/login.xhtml">

    <page view-id="/members/*" login-required="true"/>

    ...

</pages
>
```

####

###

###
##############`components.xml`###

```
<event type="org.jboss.seam.security.notLoggedIn">
```

```
    <action execute="#{redirect.captureCurrentView}"/>
</event>

<event type="org.jboss.seam.security.postAuthenticate">
    <action execute="#{redirect.returnToCapturedView}"/>
</event
>
```

##############################authenticate()##################

15.3.8. HTTP##

######################Seam##RFC2617##HTTPBasic####HTTPDigest##################
####################components.xml# authentication-filter####################

```
 <web:authentication-filter url-pattern="*.seam" auth-type="basic"/>
```

####################auth-type#basic########################digest#######
################## key # realm##############

```
    <web:authentication-filter  url-pattern="*.seam"  auth-type="digest"  key="AA3JK34aSDlkj"
realm="My App"/>
```

key########## realm##################################

15.3.8.1.

#################org.jboss.seam.security.digest.DigestAuthenticator################validatePass
##########

```
  public boolean authenticate()
  {
    try
    {
      User user = (User) entityManager.createQuery(
        "from User where username = :username")
```

```
            .setParameter("username", identity.getUsername())
            .getSingleResult();

        return validatePassword(user.getPassword());
      }
      catch (NoResultException ex)
      {
        return false;
      }
    }
```

15.3.9.

################################API############################

15.3.9.1. ############JAAS###

##Seam#######API############JAAS##############components.xml#jaas-config-name
#########################JAAS#############
#####JBossAS####################### other#####JBossAS#####
UsersRolesLoginModule#########################components.xml##############

```
<security:identity jaas-config-name="other"/>
```

#####Seam################JAAS###################################Seam#####################

15.4. ID###

ID############ID###############LDAP#########Seam##################API#######
ID##API####identityManager###

######identityManager######IdentityStores#############
############################LDAP############################

15.4.1. ID########

identityManager
####################ID##################LDAP####################RDB#####################

Seam#IdentityStore######IdentityStore############
####RDB####################JpaIdentityStore########ID##############identityManager
############################
#####LdapIdentityStore##LDAP##########################

identityManager   #########identityStore   #
roleIdentityStore############################IdentityStore############Seam############EL#####
############################
JpaIdentityStore#########identityStore############roleIdentityStore
######################
####components.xml#LdapIdentityStore#identityManager###

```
<security:identity-manager identity-store="#{ldapIdentityStore}"/>
```

###############LdapIdentityStore##############JpaIdentityStore######identityManager
#########

```
<security:identity-manager
  identity-store="#{ldapIdentityStore}"
  role-identity-store="#{jpaIdentityStore}"/>
```

##########ID##########################

15.4.2. JpaIdentityStore

##ID##
###
##########################Bean############################

15.4.2.1. JpaIdentityStore###

JpaIdentityStore   #   user-class   ###role-class##############
###
##############SeamSpace#components.xml#############

```
<security:jpa-identity-store
  user-class="org.jboss.seam.example.seamspace.MemberAccount"
  role-class="org.jboss.seam.example.seamspace.MemberRole"/>
```

15.4.2.2.

##Bean#######
##################################

15.1.

#######	##	##
@UserPrincipal	####	###############username############################
@UserPassword	####	##password############ password############hash####################md5, sha # none###### ```@UserPassword (hash="md5") public String getPasswordHash() { return passwordHash; }``` Seam################################PasswordHash##########
@UserFirstName	#####	##
@UserLastName	#####	##
@UserEnabled	#####	## ############################boolean########### ##
@UserRoles	####	## ########################

15.2.

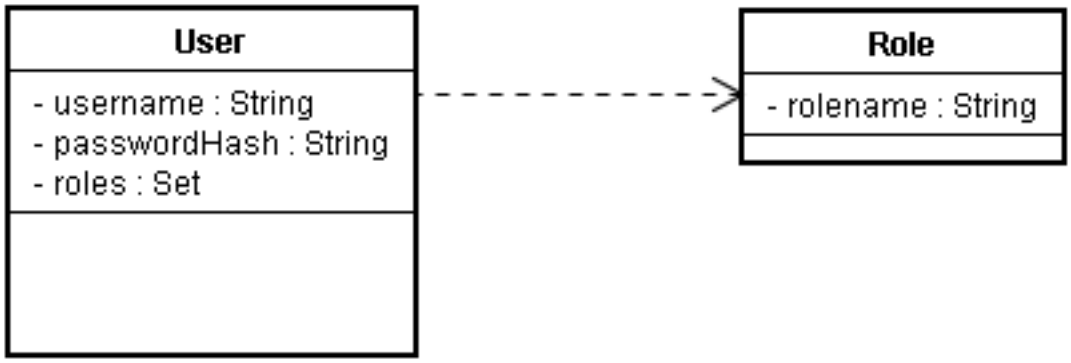
#######	##	##
@RoleName	####	######################################
@RoleGroups	#####	###
@RoleConditional	#####	###################################

15.4.2.3. ###### Bean##

########JpaIdentityStore###

15.4.2.3.1.

###################UserRoles####many-to-many###########user#role###############



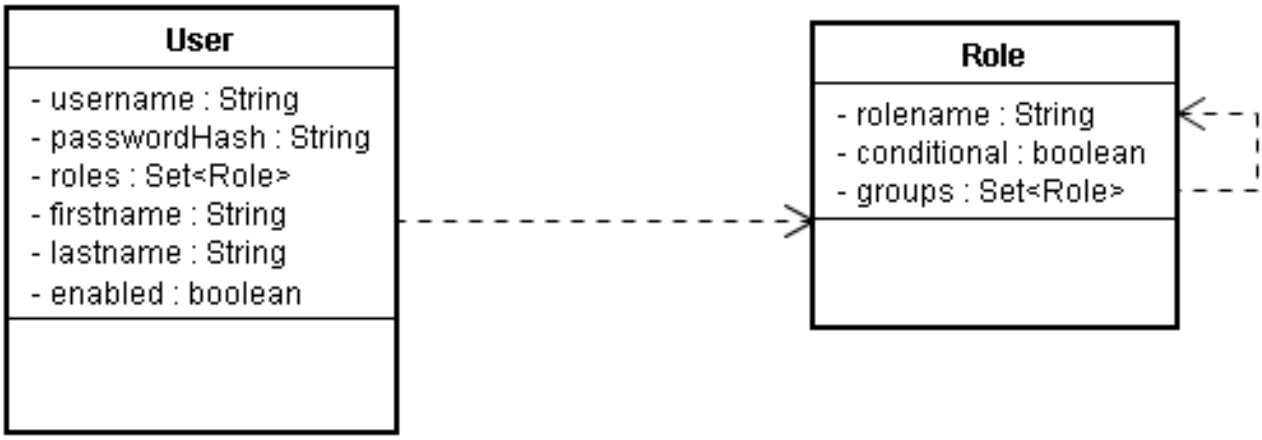

```
@Entity
public class User {
  private Integer userId;
  private String username;
  private String passwordHash;
  private Set<Role
> roles;

  @Id @GeneratedValue
  public Integer getUserId() { return userId; }
  public void setUserId(Integer userId) { this.userId = userId; }

  @UserPrincipal
  public String getUsername() { return username; }
  public void setUsername(String username) { this.username = username; }

  @UserPassword(hash = "md5")
  public String getPasswordHash() { return passwordHash; }
  public void setPasswordHash(String passwordHash) { this.passwordHash = passwordHash; }

  @UserRoles
  @ManyToMany(targetEntity = Role.class)
  @JoinTable(name = "UserRoles",
    joinColumns = @JoinColumn(name = "UserId"),
    inverseJoinColumns = @JoinColumn(name = "RoleId"))
```

```
  public Set<Role
> getRoles() { return roles; }
  public void setRoles(Set<Role
> roles) { this.roles = roles; }
}
```

```
@Entity
public class Role {
  private Integer roleId;
  private String rolename;

  @Id @Generated
  public Integer getRoleId() { return roleId; }
  public void setRoleId(Integer roleId) { this.roleId = roleId; }

  @RoleName
  public String getRolename() { return rolename; }
  public void setRolename(String rolename) { this.rolename = rolename; }
}
```

15.4.2.3.2.

##




```
@Entity
public class User {
  private Integer userId;
  private String username;
  private String passwordHash;
```

```
  private Set<Role
> roles;
  private String firstname;
  private String lastname;
  private boolean enabled;

  @Id @GeneratedValue
  public Integer getUserId() { return userId; }
  public void setUserId(Integer userId) { this.userId = userId; }

  @UserPrincipal
  public String getUsername() { return username; }
  public void setUsername(String username) { this.username = username; }

  @UserPassword(hash = "md5")
  public String getPasswordHash() { return passwordHash; }
  public void setPasswordHash(String passwordHash) { this.passwordHash = passwordHash; }

  @UserFirstName
  public String getFirstname() { return firstname; }
  public void setFirstname(String firstname) { this.firstname = firstname; }

  @UserLastName
  public String getLastname() { return lastname; }
  public void setLastname(String lastname) { this.lastname = lastname; }

  @UserEnabled
  public boolean isEnabled() { return enabled; }
  public void setEnabled(boolean enabled) { this.enabled = enabled; }

  @UserRoles
  @ManyToMany(targetEntity = Role.class)
  @JoinTable(name = "UserRoles",
    joinColumns = @JoinColumn(name = "UserId"),
    inverseJoinColumns = @JoinColumn(name = "RoleId"))
  public Set<Role
> getRoles() { return roles; }
  public void setRoles(Set<Role
> roles) { this.roles = roles; }
}
```

```
@Entity
public class Role {
```

```
  private Integer roleId;
  private String rolename;
  private boolean conditional;

  @Id @Generated
  public Integer getRoleId() { return roleId; }
  public void setRoleId(Integer roleId) { this.roleId = roleId; }

  @RoleName
  public String getRolename() { return rolename; }
  public void setRolename(String rolename) { this.rolename = rolename; }

  @RoleConditional
  public boolean isConditional() { return conditional; }
  public void setConditional(boolean conditional) { this.conditional = conditional; }

  @RoleGroups
  @ManyToMany(targetEntity = Role.class)
  @JoinTable(name = "RoleGroups",
    joinColumns = @JoinColumn(name = "RoleId"),
    inverseJoinColumns = @JoinColumn(name = "GroupId"))
  public Set<Role
> getGroups() { return groups; }
  public void setGroups(Set<Role
> groups) { this.groups = groups; }

}
```

15.4.2.4. JpaIdentityStore#######

```
IdentityManager#ID#########JpaIdentityStore###########
IdentityManager#####################
```

15.4.2.4.1. JpaIdentityStore.EVENT_PRE_PERSIST_USER

```
#######IdentityManager.createUser()##########
#################################################################
#########JpaIdentityStore##### user-class##########

##########createUser()#########################################################
```

15.4.2.4.2. JpaIdentityStore.EVENT_USER_CREATED

```
#######IdentityManager.createUser()################
##############################################
```

#######EVENT_PRE_PERSIST_USER###################################
###

15.4.2.4.3. JpaIdentityStore.EVENT_USER_AUTHENTICATED

#######IdentityManager.authenticate()############
###

15.4.3. LdapIdentityStore

##ID#######LDAP###################################
##
##############IID#######################################

15.4.3.1. LdapIdentiyStore###

#####components.xml ######LdapIdentityStore############

 # 15.3. LdapIdentityStore######

#####	######	##
server-address	localhost	LDAP########
server-port	389	LDAP################
user-context-DN	ou=Person,dc=acme,dc=com	######################DN)
user-DN-prefix	uid=	######################username####
user-DN-suffix	,ou=Person,dc=acme,dc=com	######################username####
role-context-DN	ou=Role,dc=acme,dc=com	######################DN)
role-DN-prefix	cn=	##################################
role-DN-suffix	,ou=Roles,dc=acme,dc=com	##################################
bind-DN	cn=Manager,dc=acme,dc=com	LDAP#######################
bind-credentials	secret	LDAP##############################
user-role-attribute	roles	##################################
role-attribute-is-DN	true	##################################
user-name-attribute	uid	##############username###########
user-password-attribute	userPassword	##############password###########
first-name-attribute	null	##############first name###############

#####	######	##
last-name-attribute	sn	##############last name###############
full-name-attribute	cn	##################################
enabled-attribute	null	##################################
role-name-attribute	cn	################################
object-class-attribute	objectClass	##################################
role-object-classes	organizationalRole	##############################
user-object-classes	person,uidObject	##############################
security-authentication-type	simple	The security level to use. Possible values are "none", "simple" and "strong".

15.4.3.2. LdapIdentityStore###

############# directory.mycompany.com########LDAP###########LdapIdentityStore ###########
##################ou=Person,dc=mycompany,dc=com###############username#####uid############
###############ou=Roles,dc=mycompany,dc=com#################roles#############
############################cn ############
###########enabled###false#####################

```
<security:ldap-identity-store
  server-address="directory.mycompany.com"
  bind-DN="cn=Manager,dc=mycompany,dc=com"
  bind-credentials="secret"
  user-DN-prefix="uid="
  user-DN-suffix=",ou=Person,dc=mycompany,dc=com"
  role-DN-prefix="cn="
  role-DN-suffix=",ou=Roles,dc=mycompany,dc=com"
  user-context-DN="ou=Person,dc=mycompany,dc=com"
  role-context-DN="ou=Roles,dc=mycompany,dc=com"
  user-role-attribute="roles"
  role-name-attribute="cn"
  user-object-classes="person,uidObject"
```

```
    enabled-attribute="enabled"
    />
```

15.4.4. ###ID########

Seam##################################ID##########org.jboss.seam.security.management.Identit

IdentityStore####################JavaDoc##########

15.4.5. ID#######

If you are using the Identity Management features in your Seam application, then it is not required to provide an authenticator component (see previous Authentication section) to enable authentication. Simply omit the `authenticate-method` from the `identity` configuration in `components.xml`, and the `SeamLoginModule` will by default use `IdentityManager` to authenticate your application's users, without any special configuration required.

15.4.6. ID########

IdentityManager####################Seam##################

```
@In IdentityManager identityManager;
```

########instance()###############

```
IdentityManager identityManager = IdentityManager.instance();
```

#######IdentityManager#API###########

15.4. ID###API

####	###	##
createUser(String name, String password)	####	#####username#password########## ##########true#######false######
deleteUser(String name)	####	################## #######true#######false######
createRole(String role)	####	################### ##########true#######false######
deleteRole(String name)	####	################# ##########true#######false######
enableUser(String name)	####	######################## ##################################

####	###	##
disableUser(String name)	####	######################## #######true#######false######
changePassword(String name, String password)	####	##################password######## ######true#######false######
isUserEnabled(String name)	####	############################ true#######false######
grantRole(String name, String role)	####	######################## #################### ###############true###############
revokeRole(String name, String role)	####	############################## ###################################
userExists(String name)	####	################true######false###
listUsers()	###	ABC########################
listUsers(String filter)	###	##########################ABC###
listRoles()	###	################
getGrantedRoles(String name)	###	################################
getImpliedRoles(String name)	###	################################### ################################### ####admin####user################ user############
authenticate(String name, String password)	####	#####ID#######username#password### ########true######false###### ###################################Id login##########Identity.login()####
addRoleToGroup(String role, String group)	####	############################ ##########true######
removeRoleFromGroup(String role, String group)	####	######################## #######true######
listRoles()	###	############

ID##API##
#####IdentityManager############################ ##################

15.5. ID##

####	##########	##############
createUser()	seam.user	##
deleteUser()	seam.user	##

####	##########	############
`createRole()`	`seam.role`	##
`deleteRole()`	`seam.role`	##
`enableUser()`	`seam.user`	##
`disableUser()`	`seam.user`	##
`changePassword()`	`seam.user`	##
`isUserEnabled()`	`seam.user`	####
`grantRole()`	`seam.user`	##
`revokeRole()`	`seam.user`	##
`userExists()`	`seam.user`	####
`listUsers()`	`seam.user`	####
`listRoles()`	`seam.role`	####
`addRoleToGroup()`	`seam.role`	##
`removeRoleFromGroup()`	`seam.role`	##

######admin##################ID##

```
rule ManageUsers
  no-loop
  activation-group "permissions"
when
  check: PermissionCheck(name == "seam.user", granted == false)
  Role(name == "admin")
then
  check.grant();
end

rule ManageRoles
  no-loop
  activation-group "permissions"
when
  check: PermissionCheck(name == "seam.role", granted == false)
  Role(name == "admin")
then
  check.grant();
end
```

15.5.

######API##
###############message.properties#################################
###

15.6.

#######	##
org.jboss.seam.loginSuccessful	######API#######################
org.jboss.seam.loginFailed	##
org.jboss.seam.NotLoggedIn	##
org.jboss.seam.AlreadyLoggedIn	######################################

15.6.

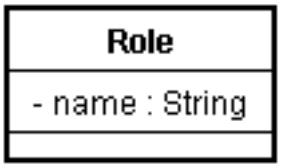
Seam#######API##
######################
##components.xml#################################

15.6.1.

Seam###
Seam######API###

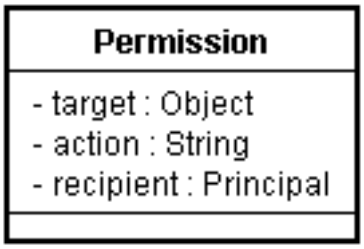
15.6.1.1.

##"admin"# "user"#
"customer"################
###



15.6.1.2.

###################################1###########
###
#################### ”##(target)”#”##(action)” #”###(receipient)”###############
###
#########Bob#####################################
##Bob##########



##################################target:action ###############

15.6.2.

####################################@Restrict#################

@Restrict####### ####################

@Restrict###########EL###

15.6.2.1. @Restrict#######

@Restrict###########Seam###
####################@Restrict###
################################Identity.checkRestriction()############
############@Restrict#############################@Restrict###################

##@Restrict#component:methodName####### ######################

```
@Name("account")
public class AccountAction {
   @Restrict public void delete() {
     ...
   }
}
```

######delete()#########account:delete####################
######@Restrict("#{s:hasPermission('account','delete')}")############################

```
@Restrict @Name("account")
public class AccountAction {
   public void insert() {
     ...
   }
```

```
    @Restrict("#{s:hasRole('admin')}")
    public void delete() {
      ...
    }
}
```

##################@Restrict#####################`Restrict###

############# #{s:hasRole()} ############# s:hasRole # s:hasPermission#EL#####
Identity################### #########API########EL#############

EL########@Restrict#########Seam##
###

```
@Name("account")
public class AccountAction {
    @In Account selectedAccount;
    @Restrict("#{s:hasPermission(selectedAccount,'modify')}")
    public void modify() {
        selectedAccount.modify();
    }
}
```

##########hasPermission()############selectedAccout###########
#######Seam##############Identity#hasPermission()################Account#####################

15.6.2.2.

#####@Restrict###Identity.checkRestri

```
public void deleteCustomer() {
    Identity.instance().checkRestriction("#{s:hasPermission(selectedCustomer,'delete')}");
}
```

####true##############

- ######################NotLoggedInException######
- ###################AuthorizationException########

########Java#######hasRole()#hasPermission()###############

```
if (!Identity.instance().hasRole("admin"))
```

```
    throw new AuthorizationException("Must be admin to perform this action");

if (!Identity.instance().hasPermission("customer", "create"))
    throw new AuthorizationException("You may not create new customers");
```

15.6.3.

```
#########################################################################
Seam##############################################EL##########1######
2########### ###############

##############################
################################################
identity.isLoggedIn()#################
```

```
<h:form class="loginForm" rendered="#{not identity.loggedIn}"
>
```

```
##########################################
###manager#######################################
##############################
```

```
<h:outputLink action="#{reports.listManagerReports}" rendered="#{s:hasRole('manager')}">
   Manager Reports
</h:outputLink
>
```

```
####################manager#############outputLink#########rendered###################<s:div>
# <s:span>###############

###########                h:dataTable##########################################
EL#s:hasPermission#####################################################################
#########################h:dataTable########
```

```
<h:dataTable value="#{clients}" var="cl">
   <h:column>
      <f:facet name="header"
>Name</f:facet>
      #{cl.name}
   </h:column>
   <h:column>
```

```
      <f:facet name="header"
>City</f:facet>
      #{cl.city}
   </h:column>
   <h:column>
      <f:facet name="header"
>Action</f:facet>
      <s:link value="Modify Client" action="#{clientAction.modify}"
            rendered="#{s:hasPermission(cl,'modify')}"/>
      <s:link value="Delete Client" action="#{clientAction.delete}"
            rendered="#{s:hasPermission(cl,'delete')}"/>
   </h:column>
</h:dataTable
>
```

15.6.4.

##########################pages.xml##
page######<restrict/>########## ####restrict##########
#########GET############## /viewId.xhtml:render
##########JSF######################### /viewId.xhtml:restore############
################

```
<page view-id="/settings.xhtml">
   <restrict/>
</page
>
```

############GET######/settings.xhtml:render##################/
settings.xhtml:restore###########

```
<page view-id="/reports.xhtml">
   <restrict
>#{s:hasRole('admin')}</restrict>
</page
>
```

#########faces####non-faces#########admin##################

15.6.5.

Seam###################read,insert,update###delete################################

##@Restrict##############

```
@Entity
@Name("customer")
@Restrict
public class Customer {
  ...
}
```

###@Restrict#########################entity:action#################
################################action # read, insert, update #### delete########

########################@Restrict ##

- @PostLoad - ##read ####################
- @PrePersist - ################## (####################### #######insert ####################
- @PreUpdate - ###################### #######update###################
- @PreRemove - ###################### #######delete###################

####insert##
##

```
@PrePersist @Restrict
public void prePersist() {}
```

/META-INF/orm.xml###

/META-INF/orm.xml#######################:

```
<?xml version="1.0" encoding="UTF-8"?>
<entity-mappings xmlns="http://java.sun.com/xml/ns/persistence/orm"
                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                     xsi:schemaLocation="http://java.sun.com/xml/ns/persistence/orm
 http://java.sun.com/xml/ns/persistence/orm_1_0.xsd"
                 version="1.0">
```

```
    <entity class="Customer">
      <pre-persist method-name="prePersist" />
    </entity>

  </entity-mappings
  >
```

#####Customer#prePersist() #####@Restrict#############

####################MemberBlog###seamspace#######
##MemberBlog#########

```
rule InsertMemberBlog
  no-loop
  activation-group "permissions"
when
  principal: Principal()
                    memberBlog:          MemberBlog(member          :          member          ->
 (member.getUsername().equals(principal.getName())))
  check: PermissionCheck(target == memberBlog, action == "insert", granted == false)
then
  check.grant();
end;
```

######Principal###memberBlog:insert##############
############### "principal: Principal()" ############
##########Principal################principal##########
###################################Principal######################### ####JBoss
Rules ################

####JPA######Seam##################################

15.6.5.1. JPA##############

EJB3######Bean############EntityListener#############META-INF/
orm.xml#########################

```
<?xml version="1.0" encoding="UTF-8"?>
<entity-mappings xmlns="http://java.sun.com/xml/ns/persistence/orm"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="http://java.sun.com/xml/ns/persistence/orm http://java.sun.com/
xml/ns/persistence/orm_1_0.xsd"
```

```
            version="1.0">

    <persistence-unit-metadata>
        <persistence-unit-defaults>
            <entity-listeners>
                <entity-listener class="org.jboss.seam.security.EntitySecurityListener"/>
            </entity-listeners>
        </persistence-unit-defaults>
    </persistence-unit-metadata>

</entity-mappings
>
```

15.6.5.2. #####Hibernate######################

Seam######Hibernate#SessionFactory######################orm.xml##############################

15.6.6.

Seam#@Restrict##################################@Restrict#####EL##############################

Seam#####CRUD##
###org.jboss.seam.annotations.security######################

- @Insert
- @Read
- @Update
- @Delete

##
#############

```
@Insert(Customer.class)
public void createCustomer() {
  ...
}
```

#############Customer#######################################
###############Customer.class#java.lang.Class###inse

##
##

```
 public void updateCustomer(@Update Customer customer) {
   ...
 }
```

################################@PermissionCheck#############

```
@Target({METHOD, PARAMETER})
@Documented
@Retention(RUNTIME)
@Inherited
@PermissionCheck
public @interface Promote {
  Class value() default void.class;
}
```

###@PermissionCheck#########################

```
@PermissionCheck("upgrade")
```

15.6.7.

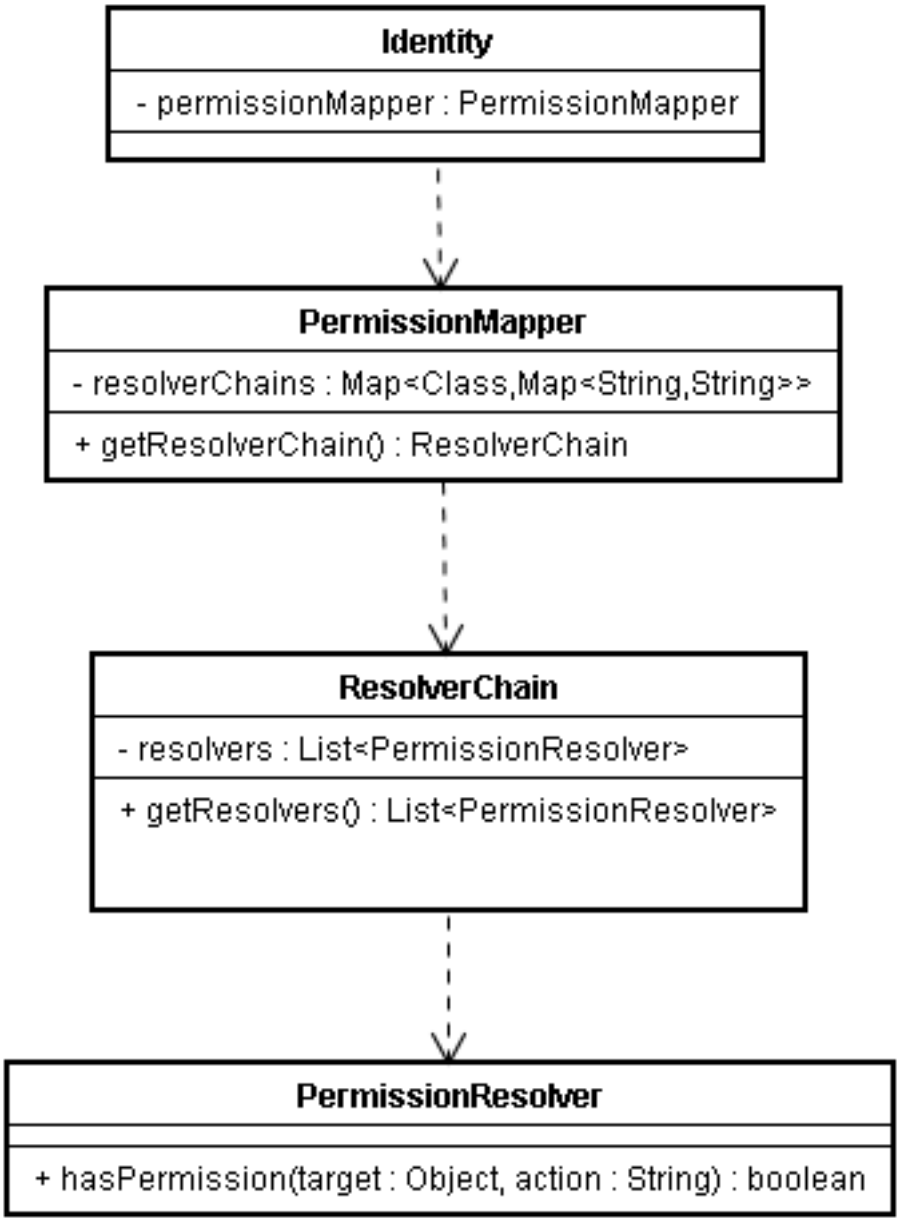
###################################Seam###
Seam#################admin###org.jboss.s
################################org.jboss.seam.annotations.security.RoleCheck############

```
@Target({METHOD})
@Documented
@Retention(RUNTIME)
@Inherited
@RoleCheck
public @interface User {
}
```

#############@User###user#

15.6.8.

Seam###
###############Seam##



############################

15.6.8.1.

##
Seam####`PermissionResolver`######### ####################

- `RuleBasedPermissionResolver` - ###############Drools###########################
- `PersistentPermissionResolver` - ##

15.6.8.1.1.

##############################PermissionResolver############################
###PermissionResolver###Seam###ResolverChai

15.7.

#####	####	##
####	`hasPermission(Object target, String action)`	#######Identity.getPrincipal()##################
`void`	`filterSetByAction(Set<Object> targets, String action)`	This method should remove any objects from the specified set, that would return `true` if passed to the `hasPermission()` method with the same `action` parameter value.

##

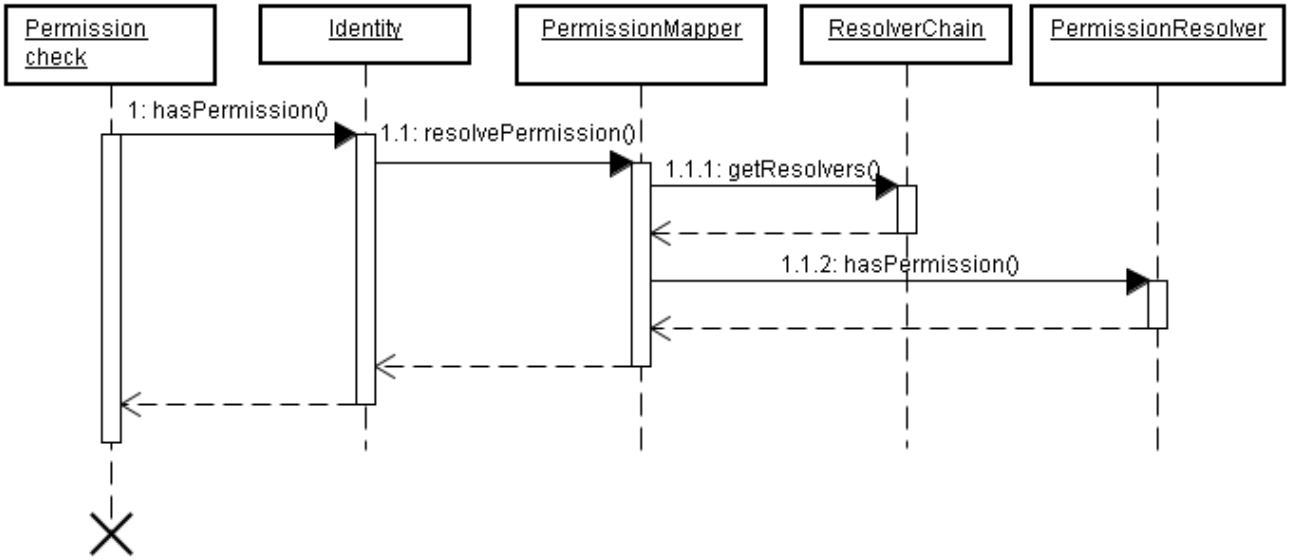
As they are cached in the user's session, any custom `PermissionResolver` implementations must adhere to a couple of restrictions. Firstly, they may not contain any state that is finer-grained than session scope (and the scope of the component itself should either be application or session). Secondly, they must not use dependency injection as they may be accessed from multiple threads simultaneously. In fact, for performance reasons it is recommended that they are annotated with `@BypassInterceptors` to bypass Seam's interceptor stack altogether.

15.6.8.2.

`ResolverChain#PermissionResolvers`###

The default `ResolverChain` consists of all permission resolvers discovered during application deployment. The `org.jboss.seam.security.defaultResolverChainCreated` event is raised (and the `ResolverChain` instance passed as an event parameter) when the default `ResolverChain` is created. This allows additional resolvers that for some reason were not discovered during deployment to be added, or for resolvers that are in the chain to be re-ordered or removed.

##
##############################EL#`s:hasPermission`#####API`Identity.checkPermission`#############



- 1 ##################EL######Identity.hasPermission()###################
- 1.1. Identity##############PermissionMapper.resolvePermission()###########
- 1.1.1. PermissionMapper##############ResolverChain#######Map################################Resolv ###ResolverChain#######ResolverChain.getResolvers()############PermissionResolverS########
- 1.1.2. ResolverChain#####PermissionResolver####PermissionMapper############################hasPermis #####PermissionResolverS #true#########################PermissionMapper#Identity####true###### ####PermissionResolverS#true######## ########################

15.6.9.

Seam########################RuleBasedPermissionResolver##Drools(JBoss Rules)################################### ################ 1## 2######Drool###

15.6.9.1.

Seam###################################Drool#######jar############################

- drools-api.jar
- drools-compiler.jar

- drools-core.jar
- drools-decisiontables.jar
- drools-templates.jar
- janino.jar
- antlr-runtime.jar
- mvel2.jar

15.6.9.2.

RuleBasedPermissionResolver##########components.xml#Drool#####################
########################securityRules###################

```
<components xmlns="http://jboss.com/products/seam/components"
            xmlns:core="http://jboss.com/products/seam/core"
            xmlns:security="http://jboss.com/products/seam/security"
            xmlns:drools="http://jboss.com/products/seam/drools"
            xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:schemaLocation=
              "http://jboss.com/products/seam/core http://jboss.com/products/seam/core-2.2.xsd
                    http://jboss.com/products/seam/components  http://jboss.com/products/seam/
components-2.2.xsd
               http://jboss.com/products/seam/drools http://jboss.com/products/seam/drools-2.2.xsd
                   http://jboss.com/products/seam/security http://jboss.com/products/seam/security-
2.2.xsd">

    <drools:rule-base name="securityRules">
       <drools:rule-files>
          <value>/META-INF/security.drl</value>
       </drools:rule-files>
    </drools:rule-base>

 </components>
```

################RuleBasedPermissionResolver#security-rules###############

```
 <security:rule-based-permission-resolver security-rules="#{prodSecurityRules}"/>
```

RuleBase###################################

15.6.9.3.

##############################jar#####/META-INF#########################
#########security.drl############components.xml##############################

##############
######Drools###

```
package MyApplicationPermissions;

 import org.jboss.seam.security.permission.PermissionCheck;
 import org.jboss.seam.security.Role;

 rule CanUserDeleteCustomers
 when
   c: PermissionCheck(target == "customer", action == "delete")
   Role(name == "admin")
 then
   c.grant();
 end
```

################ #############
Drool###

####PermissionCheck####Role#########################
###

(######################)
#######CanUserDeleteCustomers###

#################################### (LHS) #### (RHS)
########################LHS######### (###################################)
#########
LHS#when#################RHS#LHS#####################################
RHS#then############ ##########end#######

#######LHS###################### #################

```
c: PermissionCheck(target == "customer", action == "delete")
```

#########################target#####"customer"####target#####"delete"###PermissionCheck##########

############
Drool###
hasPermission()###############PermissionCheck################*(Fact)*#################

##PermissionCheck##############################hasPermission("account",
"create")#######target###
"account"#action###"create"###PermissionCheck##

PermissionCheck##########################org.jboss.seam.security.Role##########
Role##
###################################Role###
##########PermissionCheck#
Role##################java.security.Principal################

######################RuleBasedPermissionResolver.instance().getSecurityContext().insert()####
################Role###

#######LHS#c:####################
##PermissionCheck##
LHS#2###############

```
Role(name == "admin")
```

################"admin"###name#Role############################
###admin##############customer:de

######################### ######RHS##########

```
c.grant()
```

RHS#Java###################c######### (#########PermissionCheck############) #grant()##########

15.6.9.4.

##
##
###
##########################MemberBlog#####################user############################

```
rule CanCreateBlogComment
  no-loop
  activation-group "permissions"
when
  blog: MemberBlog()
  check: PermissionCheck(target == blog, action == "create", granted == false)
  Role(name == "user")
then
  check.grant();
```

```
end
```

15.6.9.5.

##PermissionCheck#acti

```
rule CanDoAnythingToCustomersIfYouAreAnAdmin
when
  c: PermissionCheck(target == "customer")
  Role(name == "admin")
then
  c.grant();
end;
```

#########admin###############customer#####################################

15.6.10.

Seam############################PersistentPermissionResolver#################################
#####################################

15.6.10.1.

########components.xml#####PersistentPermissionResolver#####PermissionStore#############
############################JpaIdentityStore#########
############################permission-store##############

```
<security:persistent-permission-resolver permission-store="#{myCustomPermissionStore}"/>
```

15.6.10.2.

PersistentPermissionResolver###
Seam#######PermissionStore##JpaPermissionStore#####################################
##############PermissionStore##

15.8.

#####	####	##
List<Permission>	listPermissions(Object target)	##################################P
List<Permission>	listPermissions(Object target, String action)	##################################

#####	####	##
List<Permission>	listPermissions(Set<Object> targets, String action)	###################################
####	grantPermission(Permission)	##########Permission##############
####	grantPermissions(List<Permission> permissions)	############ List########Permission############
####	revokePermission(Permission permission)	############Permission############
####	revokePermissions(List<Permission> permissions)	#######################Permission##
List<String>	listAvailableActions(Object target)	################################### ###################################

15.6.10.3. JpaPermissionStore####

#############Seam#########PermissionStore## ### ###

#######################################DB#############user-permission-class######### ##user-permission-class######role-permission-class###############

###

```
<security:jpa-permission-store user-permission-class="com.acme.model.AccountPermission" />
```

###

```
<security:jpa-permission-store user-permission-class="com.acme.model.UserPermission"
    role-permission-class="com.acme.model.RolePermission" />
```

15.6.10.3.1. #######

#######################################org.jboss.seam.annotations.security.permission######### #################################

15.9. ###### #######

#######	#####	##
@PermissionTarget	##########	

#######	#####	##
		## #####java.lang.String##########
@PermissionAction	##########	## #####java.lang.String##########
@PermissionUser	##########	## #####java.lang.String#########username###
@PermissionRole	##########	## #####java.lang.String####################
@PermissionDiscriminator	##########	## ################################## #######user############################## #########userValue#roleValue############# ####user#####u##role#####r##############

@PermissionDiscriminator
(userValue="u", roleValue="r")

15.6.10.3.2.

##############SeamSpace########

```
@Entity
@Name("message")
@Scope(EVENT)
public class Message implements Serializable
{
  private Long id;
  private String title;
  private String text;
  private boolean read;
  private Date datetime;

  @Id @GeneratedValue
  public Long getId() {
    return id;
  }
  public void setId(Long id) {
    this.id = id;
  }
```

```
  @NotNull @Length(max=100)
  public String getTitle() {
    return title;
  }
  public void setTitle(String title) {
    this.title = title;
  }

  @NotNull @Lob
  public String getText() {
    return text;
  }
  public void setText(String text) {
    this.text = text;
  }

  @NotNull
  public boolean isRead() {
    return read;
  }
  public void setRead(boolean read) {
    this.read = read;
  }

  @NotNull
  @Basic @Temporal(TemporalType.TIMESTAMP)
  public Date getDatetime() {
    return datetime;
  }
  public void setDatetime(Date datetime) {
    this.datetime = datetime;
  }

}
```

```
##########getDiscriminator()#####
@PermissionDiscriminator#########################################################JpaPermissio
####getRecipient()#####@PermissionUser#@PermissionRole#############
###########
discriminator###############recipient#########################################
```

15.6.10.3.3.

##
############org.jboss.seam.annotation.security.permission###########

15.10. ### #######

#######	#####	##
@Permissions	###	############ ##########@Permission##########
@Permission	###	## action################## ################################## mask################

####################### ###########SeamSpace#######

```
@Permissions({
  @Permission(action = "view"),
  @Permission(action = "comment")
})
@Entity
public class MemberImage implements Serializable {
```

#####view#comment################MemberImage##########################

15.6.10.3.4. #######

##action###DB#########################
###

####### "Bob"####MemberImage#######Bean##########
view#comment##################################action##################################"view,comm
##

```
@Permissions({
  @Permission(action = "view", mask = 1),
  @Permission(action = "comment", mask = 2)
})
@Entity
public class MemberImage implements Serializable {
```

###############

action##########"3"#bit 1 # 2 #on##########
###

mask###2####################

15.6.10.3.5.

JpaPermissionStore###
################ID#####################*identifier* *strategy*##########
#####ID#########################ID##########

IdentifierStrategy####################################

```
public interface IdentifierStrategy {
  boolean canIdentify(Class targetClass);
  String getIdentifier(Object target);
}
```

#######canIdentify()###true######
#2#####getIdentifier()#####################################

Seam####IdentifierStrategy###ClassIdentifierStrategy#EntityIdentifierStrategy###############

###############ID####################org.jboss.seam.annotations.security.permission.Identifier#
############################## ########name############################
IdentifierStrategy#############

15.6.10.3.6.

###name######ID##########Seam###D#####################@Identifier##
##########ID#"customer"######

```
@Identifier(name = "customer")
public class Customer {
```

###########"customerAction"#####:

```
@Name("customerAction")
public class CustomerAction {
```

################ "Customer"#####:

```
public class Customer {
```

15.6.10.3.7.

##ID#########Bean#######ID###ID#########
ID################ClassIdentifierStrategy###### ######## (##########*id*)
#
PersistenceProvider###
@Entity#######################, ######################ID###############

```
@Identifier(value = EntityIdentifierStrategy.class)
public class Customer {
```

##

```
@Entity
public class Customer {
  private Integer id;
  private String firstName;
  private String lastName;

  @Id
  public Integer getId() { return id; }
  public void setId(Integer id) { this.id = id; }

  public String getFirstName() { return firstName; }
  public void setFirstName(String firstName) { this.firstName = firstName; }

  public String getLastName() { return lastName; }
  public void setLastName(String lastName) { this.lastName = lastName; }
}
```

id#1#Customer################"Customer:1"######
###################################

```
@Entity
@Identifier(name = "cust")
public class Customer {
```

######id#123#Customer#"cust:123"################

15.7.

Seam############################ID##API#######################################API#PermissionManager##########

15.7.1.

PermissionManager##Seam##JpaPermissionStore####################################components.xml#permission-store#######

```
<security:permission-manager permission-store="#{ldapPermissionStore}"/>
```

#####PermissionManager#################

15.11. ############API#####

#####	####	##
List<Permission>	listPermissions(Object target, String action)	############################
List<Permission>	listPermissions(Object target)	############################
####	grantPermission(Permission permission)	######################Permis#########true####
####	grantPermissions(List<Permission> permissions)	##########################Per#########true####
####	revokePermission(Permission permission)	######################Permi
####	revokePermissions(List<Permission> permissions)	##########################Pe#########true####
List<String>	listAvailableActions(Object target)	###

15.7.2.

PermissionManager###

15.12. #########

####	##########	##############
listPermissions()	###target	seam.read-permissions
grantPermission()	#########Permission##############Permissions	seam.grant-permission
grantPermission()	#########Permission	seam.grant-permission
grantPermissions()	##############Permissions	seam.grant-permission
revokePermission()	#########Permission	seam.revoke-permission
revokePermissions()	##############Permissions	seam.revoke-permission

15.8. SSL#########

Seam#HTTPS########page#############################
######pages.xml###########scheme######################### #####/login.xhtml#HTTPS#############

```
<page view-id="/login.xhtml" scheme="https"/>
```

############JSF#s:link#s:button####### (view#######)
##/
login.xhtml#HTTPS#################s:link##login.xhmtl##HTTPS################

```
<s:link view="/login.xhtml" value="Login"/>
```

############# (##########) ##
schema="https"############http###################https###############

##################*scheme*#####################HTTPS################
######################################
#########HTTPS#########################HTTPS###############HTTPS##############################
HTTP#######scheme################pages.xml##########

```
<page view-id="*" scheme="http" />
```

#####HTTPS################schema##############

#######components.xml##############################HTTP#####################

```
<web:session invalidate-on-scheme-change="true"/>
```

########HTTPS#######HTTP######################ID##################

15.8.1.

#####HTTP#HTTPS###################pages.xml #pages ######http-port #### https-port###############

```
<pages xmlns="http://jboss.com/products/seam/pages"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://jboss.com/products/seam/pages  http://jboss.com/products/seam/pages-2.2.xsd"
      no-conversation-view-id="/home.xhtml"
      login-view-id="/login.xhtml"
      http-port="8080"
      https-port="8443">
```

15.9.

##########API############Seam#CAPCHA(Completely *A*utomated *P*ublic *T*uring test to tell *C*omputers and *H*umans *A*part)###

15.9.1. #####

###################Seam###################web.xml###

```
<servlet>
   <servlet-name
>Seam Resource Servlet</servlet-name>
   <servlet-class
>org.jboss.seam.servlet.SeamResourceServlet</servlet-class>
</servlet>

<servlet-mapping>
```

```
   <servlet-name
>Seam Resource Servlet</servlet-name>
   <url-pattern
>/seam/resource/*</url-pattern>
</servlet-mapping
>
```

15.9.2.

####################################:

```
<h:graphicImage value="/seam/resource/captcha"/>
<h:inputText id="verifyCaptcha" value="#{captcha.response}" required="true">
  <s:validate />
</h:inputText>
<h:message for="verifyCaptcha"/>
```

```
#########     graphicImage     ####################inputText##################
#####################################
```

15.9.3.

###

```
@Name("org.jboss.seam.captcha.captcha")
@Scope(SESSION)
public class HitchhikersCaptcha extends Captcha
{
  @Override @Create
  public void init()
  {
    setChallenge("What is the answer to life, the universe and everything?");
    setCorrectResponse("42");
  }

  @Override
  public BufferedImage renderChallenge()
  {
     BufferedImage img = super.renderChallenge();
     img.getGraphics().drawOval(5, 3, 60, 14); //add an obscuring decoration
     return img;
  }
```

```
}
```

15.10. ######

######################Seam#################### 6. ##########################

15.13. ######

######	##
org.jboss.seam.security.loginSuccessful	#############
org.jboss.seam.security.loginFailed	#############
org.jboss.seam.security.alreadyLoggedIn	############################
org.jboss.seam.security.notLoggedIn	######################################
org.jboss.seam.security.notAuthorized	###
org.jboss.seam.security.preAuthenticate	###############
org.jboss.seam.security.postAuthenticate	###############
org.jboss.seam.security.loggedOut	################
org.jboss.seam.security.credentialsUpdated	############################
org.jboss.seam.security.rememberMe	Identity#rememberMe###############

15.11.

```
#########################################################################
Seam#########RunAsOperation############                    ####################
Principal#Subject#################################

#########RunAsOperation##########addRole()
################################### execute() ################################
```

```
new RunAsOperation() {
  public void execute() {
    executePrivilegedOperation();
  }
}.addRole("admin")
 .run();
```

```
###getPrincipal()          #          getSubject()#####Principal#########
Subject######################## ####RunAsOperation########run()###########
```

15.12. ID########Identity component####

##############################Identity###Credent
#############Identity################APPLICATION######Identity###Identity#############################

```
@Name("org.jboss.seam.security.identity")
@Scope(SESSION)
@Install(precedence = APPLICATION)
@BypassInterceptors
@Startup
public class CustomIdentity extends Identity
{
  private static final LogProvider log = Logging.getLogProvider(CustomIdentity.class);

  private String companyCode;

  public String getCompanyCode()
  {
    return companyCode;
  }

  public void setCompanyCode(String companyCode)
  {
    this.companyCode = companyCode;
  }

  @Override
  public String login()
  {
    log.info("###### CUSTOM LOGIN CALLED ######");
    return super.login();
  }
}
```

##

SESSION##############################Identity########@Startup##########
#############Seam#######################

15.13. OpenID

OpenID is a community standard for external web-based authentication. The basic idea is that any web application can supplement (or replace) its local handling of authentication by delegating responsibility to an external OpenID server of the user's chosing. This benefits the user, who no longer has to remember a name and password for every web application he uses, and the developer, who is relieved of some of the burden of maintaining a complex authentication system.

When using OpenID, the user selects an OpenID provider, and the provider assigns the user an OpenID. The id will take the form of a URL, for example `http://maximoburrito.myopenid.com` however, it's acceptable to leave off the `http://` part of the identifier when logging into a site. The web application (known as a relying party in OpenID-speak) determines which OpenID server to contact and redirects the user to the remote site for authentication. Upon successful authentication the user is given the (cryptographically secure) token proving his identity and is redirected back to the original web application.The local web application can then be sure the user accessing the application controls the OpenID he presented.

It's important to realize at this point that authentication does not imply authorization. The web application still needs to make a determination of how to use that information. The web application could treat the user as instantly logged in and give full access to the system or it could try and map the presented OpenID to a local user account, prompting the user to register if he hasn't already. The choice of how to handle the OpenID is left as a design decision for the local application.

15.13.1. Configuring OpenID

Seam uses the openid4java package and requires four additional JARs to make use of the Seam integration. These are: `htmlparser.jar`, `openid4java.jar`, `openxri-client.jar` and `openxri-syntax.jar`.

OpenID processing requires the use of the `OpenIdPhaseListener`, which should be added to your `faces-config.xml` file. The phase listener processes the callback from the OpenID provider, allowing re-entry into the local application.

```
<lifecycle>
   <phase-listener>org.jboss.seam.security.openid.OpenIdPhaseListener</phase-listener>
</lifecycle>
```

With this configuration, OpenID support is available to your application. The OpenID support component, `org.jboss.seam.security.openid.openid`, is installed automatically if the openid4java classes are on the classpath.

15.13.2. Presenting an OpenIdDLogin form

To initiate an OpenID login, you can present a simply form to the user asking for the user's OpenID. The `#{openid.id}` value accepts the user's OpenID and the `#{openid.login}` action initiates an authentication request.

```
<h:form>
   <h:inputText value="#{openid.id}" />
   <h:commandButton action="#{openid.login}" value="OpenID Login"/>
</h:form>
```

When the user submits the login form, he will be redirected to his OpenID provider. The user will eventually return to your application through the Seam pseudo-view `/openid.xhtml`, which is provided by the `OpenIdPhaseListener`. Your application can handle the OpenID response by means of a `pages.xml` navigation from that view, just as if the user had never left your application.

15.13.3. Logging in immediately

The simplest strategy is to simply login the user immediately. The following navigation rule shows how to handle this using the `#{openid.loginImmediately()}` action.

```
<page view-id="/openid.xhtml">
   <navigation evaluate="#{openid.loginImmediately()}">
      <rule if-outcome="true">
         <redirect view-id="/main.xhtml">
            <message>OpenID login successful...</message>
         </redirect>
      </rule>
      <rule if-outcome="false">
         <redirect view-id="/main.xhtml">
            <message>OpenID login rejected...</message>
         </redirect>
      </rule>
   </navigation>
</page>
```

Thie `loginImmediately()` action checks to see if the OpenID is valid. If it is valid, it adds an OpenIDPrincipal to the identity component, marks the user as logged in (i.e. `#{identity.loggedIn}` will be true) and returns true. If the OpenID was not validated, the method returns false, and the user re-enters the application un-authenticated. If the user's OpenID is valid, it will be accessible using the expression `#{openid.validatedId}` and `#{openid.valid}` will be true.

15.13.4. Deferring login

You may not want the user to be immediately logged in to your application. In that case, your navigation should check the `#{openid.valid}` property and redirect the user to a local registration or processing page. Actions you might take would be asking for more information and creating a local user account or presenting a captcha to avoid programmatic registrations. When you are done processing, if you want to log the user in, you can call the `loginImmediately` method, either through EL as shown previously or by directly interaction with the `org.jboss.seam.security.openid.OpenId` component. Of course, nothing prevents you from writing custom code to interact with the Seam identity component on your own for even more customized behaviour.

15.13.5. Logging out

Logging out (forgetting an OpenID association) is done by calling `#{openid.logout}`. If you are not using Seam security, you can call this method directly. If you are using Seam security, you should continue to use `#{identity.logout}` and install an event handler to capture the logout event, calling the OpenID logout method.

```
<event type="org.jboss.seam.security.loggedOut">
   <action execute="#{openid.logout}" />
</event>
```

It's important that you do not leave this out or the user will not be able to login again in the same session.

##########

Seam ########################## ###
Seam

16.1.

JEE ########################
##

##

Note that all i18n features in Seam work only in JSF context.

########## 1
UTF-8
################## ##########################

16.1.1.

tomcat
################ Tomcat ### JBoss AS ######## `URIEncoding="UTF-8"`
################ JBoss AS 4.2 ## `${JBOSS_HOME}/server/(default)/deploy/jboss-web.deployer/server.xml` #######

```
<Connector port="8080" URIEncoding="UTF-8"/>
```

######## ############ JBoss AS ##

```
<Connector port="8080" useBodyEncodingForURI="true"/>
```

16.1.2.

############ ##### ################## (## ##############)# ###
ASCII ######## ASCII
################ ########################

######### ASCII ############### #### Unicode ###### Unicode ################
JVM ############ ##### ASCII
ASCII ########################## \uXXXX ########## Java ###### Unicode
############ XXXX ######## 16 #####

You can write your translation of labels (*#16.3. #####*) to your messages resource bundle in the native encoding and then convert the content of the file into the escaped format through the tool `native2ascii` provided in the JDK. This tool will convert a file written in your native encoding to one that represents non-ASCII characters as Unicode escape sequences.

########## *Java 5 ######* [http://java.sun.com/j2se/1.5.0/docs/tooldocs/index.html#intl] ### *Java 6 ######* [http://java.sun.com/javase/6/docs/technotes/tools/#intl] ########## ##### ####### UTF-8 ##################

```
$ native2ascii -encoding UTF-8 messages_cs.properties >
 messages_cs_escaped.properties
```

16.1.3.

##
##

#################### `<f:view locale="cs_CZ"/>` ############## (#### JSF ##################)# xml ##################### xml ################################ ####### xml ### `<?xml version="1.0" encoding="UTF-8"?>` ########################

JSF # Facelet ################################ ## `components.xml`

```
<web:character-encoding-filter encoding="UTF-8"
  override-client="true"
  url-pattern="*.seam" />
```

16.2.

################ `java.util.Locale` ############## (########## `locale` ####################)# ######## ################## Seam ################## JSF #########

- HTTP ########## (#########) #### #### `faces-config.xml` ################################ ##########################
- ################ `faces-config.xml` ########## ##########################
- ##########################

Seam ######## `org.jboss.seam.international.localeSelector.language`# `org.jboss.seam.international.localeSelector.country` ###

`org.jboss.seam.international.localeSelector.variant` ################ ## ####
##################

############################ ################################ Seam
#######JSP ####Facelet
########################

```
<h:selectOneMenu value="#{localeSelector.language}">
   <f:selectItem itemLabel="English" itemValue="en"/>
   <f:selectItem itemLabel="Deutsch" itemValue="de"/>
   <f:selectItem itemLabel="Francais" itemValue="fr"/>
</h:selectOneMenu>
<h:commandButton action="#{localeSelector.select}"
   value="#{messages['ChangeLanguage']}"/>
```

#####`faces-config.xml` ########################## ###########

```
<h:selectOneMenu value="#{localeSelector.localeString}">
   <f:selectItems value="#{localeSelector.supportedLocales}"/>
</h:selectOneMenu>
<h:commandButton action="#{localeSelector.select}"
   value="#{messages['ChangeLanguage']}"/>
```

Seam # JSF
#################

###################################### ##### JSF ###### (/META-INF/faces-config.xml)
`<locale-config>` ############################## #### Seam
Java EE ############################## ########
`org.jboss.seam.international.localeConfig`
Seam
########## Seam ########## XML ########## ##############################

```
<international:locale-config default-locale="fr_CA" supported-locales="en fr_CA fr_FR"/>
```

######################### ######################### #########################

16.3.

JSF ############################## `<f:loadBundle />` ################## Seam
######################### ########## EL ###################### Seam `messages`
####################

16.3.1.

Seam # java.util.ResourceBundle ######## (########## org.jboss.seam.core.resourceBundle #######)# ## ######## Seam ############### messages ###### messages.properties# messages_en.properties# messages_en_AU.properties ############################ ########### WEB-INF/classes ############

####messages_en.properties ###

```
Hello=Hello
```

####messages_en_AU.properties ###

```
Hello=G'day
```

org.jboss.seam.core.resourceLoader.bundleNames ##### Seam ############ ############################## ############################## (####) ########

```
<core:resource-loader>
   <core:bundle-names>
      <value>mycompany_messages</value>
      <value>standard_messages</value>
   </core:bundle-names>
</core:resource-loader>
```

########################## ##JSF###ID##################### ####ID#### / ############## ### /welcome/hello.jsp ################### ########### welcome/hello_en.properties #######

pages.xml ######################

```
<page view-id="/welcome/hello.jsp" bundle="HelloMessages"/>
```

HelloMessages.properties ############ /welcome/hello.jsp

16.3.2.

###Seam######################## ######### <f:loadBundle ... /> ###################### ##################

```
<h:outputText value="#{messages['Hello']}"/>
```

####

```
<h:outputText value="#{messages.Hello}"/>
```

##################### EL ############

```
Hello=Hello, #{user.firstName} #{user.lastName}
```

```
Hello=G'day, #{user.firstName}
```

#########################

```
@In private Map<String, String> messages;
```

```
@In("#{messages['Hello']}") private String helloMessage;
```

16.3.3. Faces

`facesMessages` ################################### ########faces messages #######

```
@Name("hello")
@Stateless
public class HelloBean implements Hello {
   @In FacesMessages facesMessages;

   public String sayIt() {
      facesMessages.addFromResourceBundle("Hello");
   }
}
```

##################`Hello, Gavin King` ##### `G'day, Gavin` ########

16.4.

java.util.Timezone ################## ###org.jboss.seam.international.timezone ############ Seam ######## ### org.jboss.seam.international.timezoneSelector ###### ################################## ####### <f:convertDateTime> ################## ##### JSF ############## UTC #########UTC ########## ###################

Seam################################Seam###########
###Seam##Seam############### <s:convertDateTime> #########

Seam also provides a default date converter to convert a string value to a date. This saves you from having to specify a converter on input fields that are simply capturing a date. The pattern is selected according the the user's locale and the time zone is selected as described above.

16.5.

Seam############################## ###API###########API########## ###################### ##

########################:

```
<theme:theme-selector cookie-enabled="true">
   <theme:available-themes>
      <value>default</value>
      <value>accessible</value>
      <value>printable</value>
   </theme:available-themes>
</theme:theme-selector>
```

####################################

################################## #### default #### default.properties ################## ####default.properties #############

```
css ../screen.css
template /template.xhtml
```

CSS ############ facelet ################ (##################################)#

###JSP#facelet#################### ####facelet##################:

```
<link href="#{theme.css}" rel="stylesheet" type="text/css" />
```

##

```
<link href="#{facesContext.externalContext.requestContextPath}#{theme.css}"
   rel="stylesheet" type="text/css" />
```

############ facelet## `<ui:composition>` ##################

```
<ui:composition xmlns="http://www.w3.org/1999/xhtml"
   xmlns:ui="http://java.sun.com/jsf/facelets"
   xmlns:h="http://java.sun.com/jsf/html"
   xmlns:f="http://java.sun.com/jsf/core"
   template="#{theme.template}">
```

################## ##

```
<h:selectOneMenu value="#{themeSelector.theme}">
   <f:selectItems value="#{themeSelector.themes}"/>
</h:selectOneMenu>
<h:commandButton action="#{themeSelector.select}" value="Select Theme"/>
```

16.6.

################################## ################################### ### `components.xml` # `cookie-enabled` ############

```
<theme:theme-selector cookie-enabled="true">
   <theme:available-themes>
      <value>default</value>
      <value>accessible</value>
      <value>printable</value>
   </theme:available-themes>
</theme:theme-selector>

<international:locale-selector cookie-enabled="true"/>
```

Seam Text

##########
##########wiki####blog##
Seam## *Seam Text* ########################## Web ############### `<s:formattedText/>` ############# Seam Text#ANTLR###################### Seam Text##########ANTLR######################

17.1.

######:

```
It's easy to make *emphasis*, |monospace|,
~deleted text~, super^scripts^ or _underlines_.
```

`<s:formattedText/>` ######## ###HTML#######:

```
<p>
It's easy to make <i
>emphasis</i
>, <tt
>monospace</tt>
<del
>deleted text</del
>, super<sup
>scripts</sup
> or <u
>underlines</u
>.
</p
>
```

########################## ###+ ##########:

```
+This is a big heading
You /must/ have some text following a heading!

++This is a smaller heading
This is the first paragraph. We can split it across multiple
lines, but we must end it with a blank line.
```

```
This is the second paragraph.
```

(################ ##) ###### HTML ##:

```
<h1
>This is a big heading</h1>
<p>
You <i
>must</i
> have some text following a heading!
</p>

<h2
>This is a smaller heading</h2>
<p>
This is the first paragraph. We can split it across multiple
lines, but we must end it with a blank line.
</p>

<p>
This is the second paragraph.
</p
>
```

######## # ################### = #######:

```
An ordered list:

#first item
#second item
#and even the /third/ item

An unordered list:

=an item
=another item
```

```
<p>
An ordered list:
</p>
```

```
<ol
>
<li
>first item</li>
<li
>second item</li>
<li
>and even the <i
>third</i
> item</li>
</ol>

<p>
An unordered list:
</p>

<ul>
<li
>an item</li>
<li
>another item</li>
</ul
>
```

##########:

```
The other guy said:

"Nyeah nyeah-nee
/nyeah/ nyeah!"

But what do you think he means by "nyeah-nee"?
```

```
<p>
The other guy said:
</p>

<q
>Nyeah nyeah-nee
<i
```

```
>nyeah</i
> nyeah!</q>

<p>
But what do you think he means by <q
>nyeah-nee</q
>?
</p
>
```

17.2.

*, |, # #########<, >, & ###HTML######### \ #########:

```
You can write down equations like 2\*3\=6 and HTML tags
like \<body\
> using the escape character: \\.
```

```
<p>
You can write down equations like 2*3=6 and HTML tags
like &lt;body&gt; using the escape character: \.
</p
>
```

(`)

```
My code doesn't work:

`for (int i=0; i<100; i--)
{
    doSomething();
}`

Any ideas?
```

```
<p>
My code doesn't work:
</p>
```

```
<pre
>for (int i=0; i&lt;100; i--)
{
   doSomething();
}</pre>

<p>
Any ideas?
</p
>
```

##################################
(##)#
###

```
This is a |<tag attribute="value"/>| example.
```

(##### ####)#

17.3.

####################:

```
Go to the Seam website at [=
>http://jboss.com/products/seam].
```

####################:

```
Go to [the Seam website=
>http://jboss.com/products/seam].
```

###################wiki################Seam Text####################

17.4. HTML###

############HTML################## (############
##)# ####################

```
You might want to link to <a href="http://jboss.com/products/seam"
>something
```

```
cool</a
>, or even include an image: <img src="/logo.jpg"/>
```

#########:

```
<table>
   <tr
><td
>First name:</td
><td
>Gavin</td
></tr>
   <tr
><td
>Last name:</td
><td
>King</td
></tr>
</table
>
```

######################!

17.5. Using the SeamTextParser

The `<s:formattedText/>` JSF component internally uses the `org.jboss.seam.text.SeamTextParser`. You can use that class directly and implement your own text parsing, rendering, or HTML sanitation procedure. This is especially useful if you have a custom frontend for entering rich text, such as a Javascript-based HTML editor, and you want to validate user input to protect your website against Cross-Site Scripting (XSS) attacks. Another usecase are custom wiki text parsing and rendering engines.

The following example defines a custom text parser that overrides the default HTML sanitizer:

```
public class MyTextParser extends SeamTextParser {

   public MyTextParser(String myText) {
      super(new SeamTextLexer(new StringReader(myText)));

      setSanitizer(
         new DefaultSanitizer() {
            @Override
```

```
            public void validateHtmlElement(Token element) throws SemanticException {
               // TODO: I want to validate HTML elements myself!
            }
         }
      );
   }

   // Customizes rendering of Seam text links such as [Some Text=>http://example.com]
   @Override
   protected String linkTag(String descriptionText, String linkText) {
      return "<a href=\"" + linkText + "\">My Custom Link: " + descriptionText + "</a>";
   }

   // Renders a <p> or equivalent tag
   @Override
   protected String paragraphOpenTag() {
      return "<p class=\"myCustomStyle\">";
   }

   public void parse() throws ANTLRException {
      startRule();
   }

}
```

The `linkTag()` and `paragraphOpenTag()` methods are just some of many you can override to customize rendered output. These methods generally return `String`. See the Javadoc for more details.

Also consult the Javadoc of `org.jboss.seam.text.SeamTextParser.DefaultSanitizer` for more information on what HTML elements, attributes, and attribute values or filtered by default.

iText PDF

Seam now includes a component set for generating documents using iText. The primary focus of Seam's iText document support is for the generation of PDF documents, but Seam also offers basic support for RTF document generation.

18.1. PDF

iText support is provided by `jboss-seam-pdf.jar`. This JAR contains the iText JSF controls, which are used to construct views that can render to PDF, and the DocumentStore component, which serves the rendered documents to the user. To include PDF support in your application, put `jboss-seam-pdf.jar` in your `WEB-INF/lib` directory along with the iText JAR file. There is no further configuration needed to use Seam's iText support.

Seam iText ################### Facelets ############## ############### JSP ################## ### seam-ui #################

`examples/itext` ################ PDF ################## ################ ########### PDF ################################

18.1.1.

`<p:document>`	## ####### `http://jboss.com/products/seam/pdf` ########### facelet XHTML ######## ################ `document` ########## `document` ### Seam ######## DocumentStore ############## HTML ########################## ## • `type` — ################## ##### `PDF`# `RTF`# `HTML` ########## Seam # PDF ################### PDF #################### • `pageSize` — ############### ############ `LETTER` # A4 ##### ############ `com.lowagie.text.PageSize` ######### #### pageSize ###################### ##### #612 792## LETTER ############### • `orientation` — ######### ##### `portrait` # `landscape` ### landscape ########################## • `margins` — ## ## ## ######### • `marginMirroring` — ########################

	• disposition — ######## PDF ############### HTTP Content-Disposition ######## ## inline ########################## attachment ###### ######## inline ### • fileName — ###### ######################## ######## • title • subject • keywords • author • creator ###

```
<p:document xmlns:p="http://jboss.com/products/seam/pdf"
>
  The document goes here.

</p:document
>
```

18.1.2.

Useful documents will need to contain more than just text; however, the standard UI components are geared towards HTML generation and are not useful for generating PDF content. Instead, Seam provides a special UI components for generating suitable PDF content. Tags like `<p:image>` and `<p:paragraph>` are the basic foundations of simple documents. Tags like `<p:font>` provide style information to all the content surrounding them.

`<p:paragraph>`	## ############ ## ## • firstLineIndent

	• extraParagraphSpace • leading • multipliedLeading • spacingBefore — ###################### • spacingAfter — ###################### • indentationLeft • indentationRight • keepTogether ### <p:paragraph alignment="justify"> This is a simple document. It isn't very fancy. </p:paragraph >
<p:text>	## text ######## JSF ## HTML ####################### outputText ######### ## • value — ######### #################### ### <p:paragraph> The item costs <p:text value="#{product.price}"> <f:convertNumber type="currency" currencySymbol="$"/> </p:text> </p:paragraph >
<p:html>	## html ### HTML ###### PDF ##########

	## • value — ############ ###

```
<p:html value="This is HTML with <b
>some markup</b
>." />
<p:html>
   <h1
>This is more complex HTML</h1>
   <ul>
      <li
>one</li>
      <li
>two</li>
      <li
>three</li>
   </ul>
</p:html>

<p:html>
      <s:formattedText value="*This* is |Seam Text| as HTML.  It's
 very^cool^." />
</p:html
>
```

<p:font>	## ## ## • name — ######## ##### COURIER# HELVETICA# TIMES-ROMAN# SYMBOL# ZAPFDINGBATS ##### • size — ############### • style — ############ ################## NORMAL# BOLD# ITALIC# OBLIQUE# UNDERLINE# LINE-THROUGH

	• `color` — The font color. (see *#18.1.7.1. #Color Values#* for color values)
	• `encoding` — ################
	###
	
	`<p:font name="courier" style="bold" size="24">`
	`  <p:paragraph`
	`>My Title</p:paragraph>`
	`</p:font`
	`>`

`<p:textcolumn>`	##
	
	`p:textcolumn` inserts a text column that can be used to control the flow of text. The most common case is to support right to left direction fonts.
	
	##
	
	• `left` — The left bounds of the text column
	• `right` — The right bounds of the text column
	• `direction` — The run direction of the text in the column: `RTL`, `LTR`, `NO-BIDI`, `DEFAULT`
	
	###
	
	`<p:textcolumn left="400" right="600" direction="rtl"`
	`>`
	`  <p:font name="/Library/Fonts/Arial Unicode.ttf"`
	`  encoding="Identity-H"`
	`  embedded="true"`
	`>#{phrases.arabic}</p:font`
	`>`
	`</p:textcolumn`
	`>`

`<p:newPage>`	## `p:newPage` ############# ###

	<p:newPage />
<p:image>	## p:image inserts an image into the document. Images can be loaded from the classpath or from the web application context using the value attribute. ################################ imageData ######################### #### java.awt.Image ############ ## • value — ## • rotation — ############## • height — ########## • width — ######### • alignment— ########## (#########18.1.7.2. ######## ###) • alt — ################## • indentationLeft • indentationRight • spacingBefore — ###################### • spacingAfter — ###################### • widthPercentage • initialRotation • dpi • scalePercent — ############ (#######)# x # y ################### 2 ################### • scaleToFit — Specifies the X any Y size to scale the image to. The image will be scale to fit those dimensions as closely as possible while preserving the XY ratio of the image. • wrap

	• `underlying` ###

```
<p:image value="/jboss.jpg" />
```

```
<p:image value="#{images.chart}" />
```

`<p:anchor>`	## `p:anchor` ########################## ############### ## • `name` — ################### • `reference` — The destination the link refers to. Links to other points in the document should begin with a "#". For example, "#link1" to refer to an anchor position with a `name` of `link1`. Links may also be a full URL to point to a resource outside of the document. ###

```
<p:listItem
><p:anchor reference="#reason1"
>Reason 1</p:anchor
></p:listItem
>
...
<p:paragraph>
   <p:anchor name="reason1"
>It's the quickest way to get "rich"</p:anchor
>
   ...
</p:paragraph
>
```

18.1.3.

`<p:header>`
 	
`<p:footer>`
	##

	p:header # p:footer ########## ## ## • alignment — ########## (######### #18.1.7.2. ######## ###)# • backgroundColor — ########## (######## #18.1.7.1. #Color Values# ###)# • borderColor — ########## borderColorLeft# borderColorRight# borderColorTop# borderColorBottom ########## (######## #18.1.7.1. #Color Values# ###)# • borderWidth — The width of the border. Individual border sides can be specified using borderWidthLeft, borderWidthRight, borderWidthTop and borderWidthBottom. ### <f:facet name="header"> <p:font size="12"> <p:footer borderWidthTop="1" borderColorTop="blue" borderWidthBottom="0" alignment="center"> Why Seam? [<p:pageNumber />] </p:footer> </p:font> </f:facet >
<p:pageNumber>	## ######### p:pageNumber ########## ########## 1 ########## ### <p:footer borderWidthTop="1" borderColorTop="blue" borderWidthBottom="0" alignment="center"> Why Seam? [<p:pageNumber />] </p:footer>

18.1.4.

`<p:chapter>`
 	
`<p:section>`
	##
	
	############################ `p:chapter` # `p:section`
	############################## ###################### ###
	############## ##### PDF
	##

> **Note**
>
> ##
>
> You cannot include a chapter into another chapter, this can be done only with section(s).

##

- `alignment` — ######################## (######### *#18.1.7.2. ######## ###*)#
- `number` — The chapter/section number. Every chapter/section should be assigned a number.
- `numberDepth` — The depth of numbering for chapter/section. All sections are numbered relative to their surrounding chapter/sections. The fourth section of the first section of chapter three would be section 3.1.4, if displayed at the default number depth of three. To omit the chapter number, a number depth of 2 should be used. In that case, the section number would be displayed as 1.4.

  > **Note**
  >
  > ##
  >
  > Chapter(s) can have a number or without it by setting numberDepth to 0.

###

```
<p:document xmlns:p="http://jboss.com/products/seam/pdf"
        title="Hello">

  <p:chapter number="1">
    <p:title
><p:paragraph
```

```
>Hello</p:paragraph
></p:title>
      <p:paragraph
>Hello #{user.name}!</p:paragraph>
   </p:chapter>

   <p:chapter number="2">
      <p:title
><p:paragraph
>Goodbye</p:paragraph
></p:title>
      <p:paragraph
>Goodbye #{user.name}.</p:paragraph>
   </p:chapter>

</p:document
>
```

`<p:header>`	## ############ p:title ############ ######/ ##################### ########## ############### p:paragraph ############

18.1.5.

List structures can be displayed using the `p:list` and `p:listItem` tags. Lists may contain arbitrarily-nested sublists. List items may not be used outside of a list. The following document uses the `ui:repeat` tag to to display a list of values retrieved from a Seam component.

```
<p:document xmlns:p="http://jboss.com/products/seam/pdf"
          xmlns:ui="http://java.sun.com/jsf/facelets"
          title="Hello">
   <p:list style="numbered">
      <ui:repeat value="#{documents}" var="doc">
         <p:listItem
>#{doc.name}</p:listItem>
      </ui:repeat>
   </p:list>
</p:document
>
```

`<p:list>`	##

- `style` — ######################## `NUMBERED# LETTERED# GREEK# ROMAN# ZAPFDINGBATS# ZAPFDINGBATS_NUMBER` ############ ######################
- `listSymbol` — ############ ############
- `indent` — ###############
- `lowerCase` — letters ################ ######################
- `charNumber` — ZAPFDINGBATS #### ########################
- `numberType` — ZAPFDINGBATS_NUMBER #### ################

###

```
<p:list style="numbered">
  <ui:repeat value="#{documents}" var="doc">
    <p:listItem
>#{doc.name}</p:listItem>
  </ui:repeat>
</p:list
>
```

`<p:listItem>`

##

`p:listItem` #################

##

- `alignment` — ############ (######### *#18.1.7.2. ######## ###*)#
- `indentationLeft` — ##########
- `indentationRight` — ##########
- `listSymbol` — ##############################

###

```
...
```

18.1.6.

p:table # p:cell ################## ######################### ###### 3 ####### 3 ################## ################

<p:table>	## p:table ################ ## • columns — ########### (##) ##### • widths — ########### ########## 1 ####### ##### widths="2 1 1" ####### 3 ### 1 ####### 2 ### 3 ####### 2 ################## • headerRows — ############################ ######################## • footerRows — ################## #### headerRows ########### ########### 2 ############ 1 ########### headerRows # 3 # footerRows # 1 ######### • widthPercentage — ################ • horizontalAlignment — ######## (######### #18.1.7.2. ######## ###)# • skipFirstHeader • runDirection • lockedWidth • splitRows • spacingBefore — ###################### • spacingAfter — ###################### • extendLastRow • headersInEvent • splitLate • keepTogether ###

```
<p:table columns="3" headerRows="1">
  <p:cell
>name</p:cell>
  <p:cell
>owner</p:cell>
  <p:cell
>size</p:cell>
  <ui:repeat value="#{documents}" var="doc">
    <p:cell
>#{doc.name}</p:cell>
    <p:cell
>#{doc.user.name}</p:cell>
    <p:cell
>#{doc.size}</p:cell>
  </ui:repeat>
</p:table
>
```

`<p:cell>`	## `p:cell` ################ ## • `colspan` — `colspan` # 1 ##################################### ########################## • `horizontalAlignment` — ######### (##### *#18.1.7.2. ######## ###*)# • `verticalAlignment` — ######### (##### *#18.1.7.2. ######## ###*)# • `padding` — `paddingLeft`# `paddingRight`# `paddingTop`# `paddingBottom` ########################## • `useBorderPadding` • `leading` • `multipliedLeading` • `indent` • `verticalAlignment`

- `extraParagraphSpace`
- `fixedHeight`
- `noWrap`
- `minimumHeight`
- `followingIndent`
- `rightIndent`
- `spaceCharRatio`
- `runDirection`
- `arabicOptions`
- `useAscender`
- `grayFill`
- `rotation`

###

```
<p:cell
>...</p:cell
>
```

18.1.7.

##

18.1.7.1. Color Values

Several ways of specifying colors are provided. A limited number of colors are supported by name. They are: `white`, `gray`, `lightgray`, `darkgray`, `black`, `red`, `pink`, `yellow`, `green`, `magenta`, `cyan` and `blue`. Colors can be specified as an integer value, as definied by `java.awt.Color`. Finally a color value may be specified as `rgb(r,g,b)` or `rgb(r,g,b,a)` with the red, green, blue alpha values specified as an integer between 0 and 255 or as a floating point percentages followed by a '%' sign.

18.1.7.2.

################ Seam PDF ############# `left#` `right#` `center#` `justify#` `justifyall` ########### ###### `top#` `middle#` `bottom#` `baseline` ###

18.2.

Charting support is also provided with `jboss-seam-pdf.jar`. Charts can be used in PDF documents or can be used as images in an HTML page. Charting requires the JFreeChart library (`jfreechart.jar` and `jcommon.jar`) to be added to the `WEB-INF/lib` directory. Four types of charts are currently supported: pie charts, bar charts and line charts. Where greater variety or control is needed, it is possible to construct charts using Java code.

`<p:chart>`	## Displays a chart created in Java by a Seam component. ## • `chart` — The chart object to display. • `height` — ######### • `width` — ######## ### `<p:chart chart="#{mycomponent.chart}" width="500" height="500" />`
`<p:barchart>`	## ########### ## • `chart` — The chart object to display, if programmatic chart creation is being used. • `dataset` — The dataset to be displayed, if programmatic dataset is being used. • `borderVisible` — ############################ • `borderPaint` — ################### • `borderBackgroundPaint` — ############### • `borderStroke` — • `domainAxisLabel` — ##############

- `domainLabelPosition` — The angle of the domain axis category labels. Valid values are `STANDARD`, `UP_45`, `UP_90`, `DOWN_45` and `DOWN_90`. Alternatively, the value can the positive or negative angle in radians.
- `domainAxisPaint` — ###########
- `domainGridlinesVisible`— ##############################
- `domainGridlinePaint`— ####################
- `domainGridlineStroke` — The stroke style of the domain gridlines, if visible.
- `height` — #########
- `width` — ########
- `is3D` — #### 2D #### 3D ########## boolean ####
- `legend` — ################## boolean ####
- `legendItemPaint`— ######################
- `legendItemBackgoundPaint`— ############################
- `legendOutlinePaint`— ##############
- `orientation` — ####### `vertical` (#####) ### `horizontal` ######
- `plotBackgroundPaint`— #########
- `plotBackgroundAlpha`— ######### (###) ###### 0 (#####) ## 1 (######) ###########
- `plotForegroundAlpha`— ####### (###) ###### 0 (#####) ## 1 (######) ###########
- `plotOutlinePaint`— ######################
- `plotOutlineStroke` — The stroke style of the range gridlines, if visible.
- `rangeAxisLabel` — ###############
- `rangeAxisPaint` — ###########
- `rangeGridlinesVisible`— ##############################

	• `rangeGridlinePaint`— ################### • `rangeGridlineStroke` — The stroke style of the range gridlines, if visible. • `title` — ############### • `titlePaint`— ################ • `titleBackgroundPaint`— ################ • `width` — ######## ### ``` <p:barchart title="Bar Chart" legend="true" width="500" height="500"> <p:series key="Last Year"> <p:data columnKey="Joe" value="100" /> <p:data columnKey="Bob" value="120" /> </p:series > <p:series key="This Year"> <p:data columnKey="Joe" value="125" /> <p:data columnKey="Bob" value="115" /> </p:series> </p:barchart > ```
`<p:linechart>`	## ############ ## • `chart` — The chart object to display, if programmatic chart creation is being used. • `dataset` — The dataset to be displayed, if programmatic dataset is being used. • `borderVisible` — ########################## • `borderPaint` — ################## • `borderBackgroundPaint` — ############### • `borderStroke` —

- `domainAxisLabel` — ##############
- `domainLabelPosition` — The angle of the domain axis category labels. Valid values are `STANDARD`, `UP_45`, `UP_90`, `DOWN_45` and `DOWN_90`. Alternatively, the value can the positive or negative angle in radians.
- `domainAxisPaint` — ###########
- `domainGridlinesVisible`— ###############################
- `domainGridlinePaint`— ####################
- `domainGridlineStroke` — The stroke style of the domain gridlines, if visible.
- `height` — #########
- `width` — ########
- `is3D` — #### 2D #### 3D ########## boolean ####
- `legend` — ################## boolean ####
- `legendItemPaint` — The default color of the text labels in the legend.
- `legendItemBackgoundPaint` — The background color for the legend, if different from the chart background color.
- `legendOutlinePaint` — The color of the border around the legend.
- `orientation` — ####### `vertical` (#####) ### `horizontal` ######
- `plotBackgroundPaint` — The color of the plot background.
- `plotBackgroundAlpha` — The alpha (transparency) level of the plot background. It should be a number between 0 (completely transparent) and 1 (completely opaque).
- `plotForegroundAlpha` — The alpha (transparency) level of the plot. It should be a number between 0 (completely transparent) and 1 (completely opaque).
- `plotOutlinePaint` — The color of the range gridlines, if visible.
- `plotOutlineStroke` — The stroke style of the range gridlines, if visible.
- `rangeAxisLabel` — ###############

	• `rangeAxisPaint` — ########## • `rangeGridlinesVisible` — Controls whether or not gridlines for the range axis are shown on the chart. • `rangeGridlinePaint` — The color of the range gridlines, if visible. • `rangeGridlineStroke` — The stroke style of the range gridlines, if visible. • `title` — ############### • `titlePaint` — The color of the chart title text. • `titleBackgroundPaint` — The background color around the chart title. • `width` — ######## ### `<p:linechart title="Line Chart" width="500" height="500"> <p:series key="Prices"> <p:data columnKey="2003" value="7.36" /> <p:data columnKey="2004" value="11.50" /> <p:data columnKey="2005" value="34.625" /> <p:data columnKey="2006" value="76.30" /> <p:data columnKey="2007" value="85.05" /> </p:series> </p:linechart >`
`<p:piechart>`	## ########### ## • `title` — ############### • `chart` — The chart object to display, if programmatic chart creation is being used. • `dataset` — The dataset to be displayed, if programmatic dataset is being used.

- `label` — The default label text for pie sections.
- `legend` — A boolean value indicating whether or not the chart should include a legend. Default value is true
- `is3D` —A boolean value indicating that the chart should be rendered in 3D instead of 2D.
- `labelLinkMargin` — The link margin for labels.
- `labelLinkPaint` — The paint used for the label linking lines.
- `labelLinkStroke` — he stroke used for the label linking lines.
- `labelLinksVisible` — A flag that controls whether or not the label links are drawn.
- `labelOutlinePaint` — The paint used to draw the outline of the section labels.
- `labelOutlineStroke` — The stroke used to draw the outline of the section labels.
- `labelShadowPaint` — The paint used to draw the shadow for the section labels.
- `labelPaint` — The color used to draw the section labels
- `labelGap` — The gap between the labels and the plot as a percentage of the plot width.
- `labelBackgroundPaint` — The color used to draw the background of the section labels. If this is null, the background is not filled.
- `startAngle` — The starting angle of the first section.
- `circular` — A boolean value indicating that the chart should be drawn as a circle. If false, the chart is drawn as an ellipse. The default is true.
- `direction` — The direction the pie section are drawn. One of: `clockwise` or `anticlockwise`. The default is `clockwise`.
- `sectionOutlinePaint` — The outline paint for all sections.
- `sectionOutlineStroke` — The outline stroke for all sections
- `sectionOutlinesVisible` — Indicates whether an outline is drawn for each section in the plot.

	• `baseSectionOutlinePaint` — The base section outline paint. • `baseSectionPaint` — The base section paint. • `baseSectionOutlineStroke` — The base section outline stroke. ### `<p:piechart title="Pie Chart" circular="false" direction="anticlockwise" startAngle="30" labelGap="0.1" labelLinkPaint="red" ><p:series key="Prices" ><p:data key="2003" columnKey="2003" value="7.36" /><p:data key="2004" columnKey="2004" value="11.50" /><p:data key="2005" columnKey="2005" value="34.625" /><p:data key="2006" columnKey="2006" value="76.30" /><p:data key="2007" columnKey="2007" value="85.05" /></p:series ></p:piechart >`
`<p:series>`	## #################### ## ## • `key` — ######## • `seriesPaint` — ################ • `seriesOutlinePaint` — ################## • `seriesOutlineStroke` — ######################## • `seriesVisible` — ################ boolean ### • `seriesVisibleInLegend` — A boolean indicating if the series should be listed in the legend. ###

```
<p:series key="data1">
   <ui:repeat value="#{data.pieData1}" var="item">
      <p:data columnKey="#{item.name}" value="#{item.value}" />
   </ui:repeat>
</p:series
>
```

`<p:data>`	## ########## ## • `key` — ########## • `series` — `<p:series>` ########## • `value` — ######## • `explodedPercent` — ######## explodedPercent ########## • `sectionOutlinePaint` — ########## • `sectionOutlineStroke` — ########## • `sectionPaint` — ########## ###

```
<p:data key="foo" value="20" sectionPaint="#111111"
      explodedPercent=".2" />
<p:data key="bar" value="30" sectionPaint="#333333" />
<p:data key="baz" value="40" sectionPaint="#555555"
      sectionOutlineStroke="my-dot-style" />
```

`<p:color>`	## ########## ## • `color` — ###### ########## *#18.1.7.1. #Color Values#*

	• color2 — ###################### • point — ############ • point2 — ############ ### <p:color id="foo" color="#0ff00f"/> <p:color id="bar" color="#ff00ff" color2="#00ff00" point="50 50" point2="300 300"/>
<p:stroke>	## ######################## ## • width — ###### • cap — ######### ##### butt# round# square ### • join — ########### ##### miter# round# bevel ### • miterLimit — ############################ • dash — ############################ ###################################### • dashPhase — The dash phase indicates the offset into the dash pattern that the line should be drawn with. ### <p:stroke id="dot2" width="2" cap="round" join="bevel" dash="2 3" />

18.3.

Seam # iText ################################ ########## PDF ############## Web ############################ HTML ############## ######################################

<p:barCode>	## ################

##

- `type` — iText ########################## `EAN13`# `EAN8`# `UPCA`# `UPCE`# `SUPP2`# `SUPP5`# `POSTNET`# `PLANET`# `CODE128`# `CODE128_UCC`# `CODE128_RAW`# `CODABAR` #########
- `code` — The value to be encoded by the barcode.
- `xpos` — For PDFs, the absolute y position of the barcode on the page.
- `ypos` — For PDFs, the absolute y position of the barcode on the page.
- `rotDegrees` — PDF ########################
- `barHeight` — barCode ##########
- `minBarWidth` — #########
- `barMultiplier` — ############## ### `POSTNET` # `PLANET` ################
- `barColor` — #########
- `textColor` — ###############
- `textSize` — ######################
- `altText` — HTML ######## `alt` #######

###

```
<p:barCode type="code128"
        barHeight="80"
        textSize="20"
        code="(10)45566(17)040301"
        codeType="code128_ucc"
        altText="My BarCode" />
```

18.4.

If you have a complex, pre-generated PDF with named fields, you can easily fill in the values from your application and present it to the user.

`<p:form>`	## ######################

	## • `URL` — ############ PDF ########### URL ### ######### (://) ########################## • `filename` — ##### PDF ################## • `exportKey` — ################### DocumentData ############ PDF ########### #########################
`<p:field>`	## ################## ## • `name` — ######### • `value` — ########## • `readOnly` — ############### ###### true ###

```
<p:form
      xmlns:p="http://jboss.com/products/seam/pdf"
      URL="http://localhost/Concept/form.pdf">
   <p:field name="person.name" value="Me, myself and I"/>
</p:form>
```

18.5. Swing/AWT

Seam now provides experimental support for rendering Swing components into a PDF image. Some Swing look and feels supports, notably ones that use native widgets, will not render correctly.

`<p:swing>`	## Swing ######## PDF ################## ## • `width` — ######################

- `height` — The height of the component to be rendered.
- `component` — Swing ### AWT ################

###

```
<p:swing width="310" height="120" component="#{aButton}" />
```

18.6. iText

########################## ################ ####
###################################

########## URL `/seam-doc.seam` ## PDF ############ ######## `/myDocument.pdf` ######### PDF ####### URL ########## ############### ###################### PDF ############### `*.pdf` ##### DocumentStoreServlet ##################

```
<servlet>
   <servlet-name
>Document Store Servlet</servlet-name>
   <servlet-class
>org.jboss.seam.document.DocumentStoreServlet</servlet-class>
</servlet>

<servlet-mapping>
   <servlet-name
>Document Store Servlet</servlet-name>
   <url-pattern
>*.pdf</url-pattern>
</servlet-mapping
>
```

#################### `use-extensions`
URL

```
<components xmlns="http://jboss.com/products/seam/components"
   xmlns:document="http://jboss.com/products/seam/document"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="
     http://jboss.com/products/seam/document http://jboss.com/products/seam/document-2.2.xsd
       http://jboss.com/products/seam/components http://jboss.com/products/seam/components-
2.2.xsd">
```

```
    <document:document-store use-extensions="true"/>
</components>
```

############################ ###################### ########################
documentStore # error-page
##

```
<document:document-store use-extensions="true" error-page="/documentMissing.seam" />
```

18.7.

iText ######## ###########

- *iText ######* [http://www.lowagie.com/iText/]
- *iText in Action (########)* [http://www.manning.com/lowagie/]

Microsoft® Excel®

Seam#######*JExcelAPI* [http://jexcelapi.sourceforge.net/]#########Microsoft® Excel® ##the Microsoft® Excel® spreadsheet application#95#97#2

19.1. Microsoft® Excel®

The Microsoft® Excel® spreadsheet application `jboss-seam-excel.jar`. This JAR contains the the Microsoft® Excel® spreadsheet application JSF controls, which are used to construct views that can render the document, and the DocumentStore component, which serves the rendered document to the user. To include the Microsoft® Excel® spreadsheet application support in your application, include `jboss-seam-excel.jar` in your `WEB-INF/lib` directory along with the `jxl.jar` JAR file. Furthermore, you need to configure the DocumentStore servlet in your web.xml

Microsoft® Excel® #################Seam################## Facelets ################ seam-ui #################

`examples/excel` ############Microsoft® Excel® ########################## ######################################

Microsoft® Excel® ######################API##`ExcelWorkbook`#############

```
<excel:excelFactory>
  <property name="implementations">
    <key
>myExcelExporter</key>
    <value
>my.excel.exporter.ExcelExport</value>
  </property>
</excel:excelFactory
>
```

####excel#####components##########

```
xmlns:excel="http://jboss.com/products/seam/excel"
```

####`myExcelExporter`#######UIWorkbook#################################"jxl"####"csv"#########

########.xls#################################*#18.6. #iText* ##############

###IE####https##*http://www.nwnetworks.com/iezones.htm/*#####web.xml###################################

19.2.

##############################<h:dataTable>######List#Set#Map#Array#DataModel##############

```
<e:workbook xmlns:e="http://jboss.com/products/seam/excel">
  <e:worksheet>
    <e:cell column="0" row="0" value="Hello world!"/>
  </e:worksheet>
</e:workbook>
```

####################################

```
<e:workbook xmlns:e="http://jboss.com/products/seam/excel">
  <e:worksheet value="#{data}" var="item">
    <e:column>
      <e:cell value="#{item.value}"/>
    </e:column>
  </e:worksheet>
</e:workbook>
```

############workbook##worksheet########worksheet#######

##############################

19.3. workbook##

workbook################worksheet################link###########

`<e:workbook>`	## • `type` — Defines which export module to be used. The value is a string and can be either "jxl" or "csv". The default is "jxl". • `templateURI` — A template that should be used as a basis for the workbook. The value is a string (URI).

- `arrayGrowSize` — The amount of memory by which to increase the amount of memory allocated to storing the workbook data. For processes reading many small workbooks inside a WAS it might be necessary to reduce the default size. Default value is 1 megabyte. The value is a number (bytes).
- `autoFilterDisabled` — Should autofiltering be disabled?. The value is a boolean.
- `cellValidationDisabled` — Should cell validation be ignored? The value is a boolean.
- `characterSet` — The character set. This is only used when the spreadsheet is read, and has no effect when the spreadsheet is written. The value is a string (character set encoding).
- `drawingsDisabled` — Should drawings be disabled? The value is a boolean.
- `excelDisplayLanguage` — The language in which the generated file will display. The value is a string (two character ISO 3166 country code).
- `excelRegionalSettings` — The regional settings for the generated excel file. The value is a string (two character ISO 3166 country code).
- `formulaAdjust` — Should formulas be adjusted? The value is a boolean.
- `gcDisabled` — Should garbage collection be disabled? The value is a boolean.
- `ignoreBlanks` — Should blanks be ignored? The value is a boolean.
- `initialFileSize` — The initial amount of memory allocated to store the workbook data when reading a worksheet. For processes reading many small workbooks inside a WAS it might be necessary to reduce the default size. Default value is 5 megabytes. The value is a number (bytes).
- `locale` — The locale used by JExcelApi to generate the spreadsheet. Setting this value has no effect on the language or region of the generated excel file. The value is a string.
- `mergedCellCheckingDisabled` — Should merged cell checking be disabled? The value is a boolean.

- `namesDisabled` — Should handling of names be disabled? The value is a boolean.
- `propertySets` — Should any property sets be enabled (such as macros) to be copied along with the workbook? Leaving this feature enabled will result in the JXL process using more memory. The value is a boolean.
- `rationalization` — Should the cell formats be rationalized before writing out the sheet? The value is a boolean. Default is true.
- `supressWarnings` — Should warnings be suppressed?. Due to the change in logging in version 2.4, this will now set the warning behaviour across the JVM (depending on the type of logger used). The value is a boolean.
- `temporaryFileDuringWriteDirectory` — Used in conjunction with the `useTemporaryFileDuringWrite` setting to set the target directory for the temporary files. This value can be NULL, in which case the normal system default temporary directory is used instead. The value is a string (the directory to which temporary files should be written).
- `useTemporaryFileDuringWrite` — Should a temporary file is used during the generation of the workbook. If not set, the workbook will take place entirely in memory. Setting this flag involves an assessment of the trade-offs between memory usage and performance. The value is a boolean.
- `workbookProtected` — Should the workbook be protected? The value is a boolean.
- `filename` — The filename to use for the download. The value is a string. Please note that if you map the DocumentServlet to some pattern, this file extension must also match.
- `exportKey` — A key under which to store the resulting data in a DocumentData object under the event scope. If used, there is no redirection.

###

- `<e:link/>` — Zero or more stylesheet links (see *#19.14.1. ##########*).
- `<e:worksheet/>` — Zero or more worksheets (see *#19.4. #worksheet###*).

	##### • ##

```
<e:workbook>
  <e:worksheet>
    <e:cell value="Hello World" row="0" column="0"/>
  </e:worksheet>
<e:workbook>
```

################A1####################

19.4. worksheet##

Worksheets are the children of workbooks and the parent of columns and worksheet commands. They can also contain explicitly placed cells, formulas, images and hyperlinks. They are the pages that make up the workbook.

`<e:worksheet>`	• `value` — An EL-expression to the backing data. The value is a string. The target of this expression is examined for an Iterable. Note that if the target is a Map, the iteration is done over the Map.Entry entrySet(), so you should use a .key or .value to target in your references. • `var` — The current row iterator variable name that can later be referenced in cell value attributes. The value is a string. • `name` — The name of the worksheet. The value is a string. Defaults to Sheet# where # is the worksheet index. If the given worksheet name exists, that sheet is selected. This can be used for merging several data sets into a single worksheet, just define the same name for them (using `startRow` and `startCol` to make sure that they don't occupy the same space). • `startRow` — Defines the starting row for the data. The value is a number. Used for placing the data in other places than the upper-left corner (especially useful if having multiple data sets for a single worksheet). The defaults is 0. • `startColumn` — Defines the starting column for the data. The value is a number. Used for placing the data in other places than the upper-

left corner (especially useful if having multiple data sets for a single worksheet). The default is 0.

- `automaticFormulaCalculation` — Should formulas be automatically calculated? The value is a boolean.
- `bottomMargin` — The bottom margin. The value is a number (inches).
- `copies` — The number of copies. The value is a number.
- `defaultColumnWidth` — The default column width. The value is a number (characters * 256).
- `defaultRowHeight` — The default row height. The value is a number (1/20ths of a point).
- `displayZeroValues` — Should zero-values be displayed? The value is a boolean.
- `fitHeight` — The number of pages vertically that this sheet will be printed into. The value is a number.
- `fitToPages` — Should printing be fit to pages? The value is a boolean.
- `fitWidth` — The number of pages widthwise which this sheet should be printed into. The value is a number.
- `footerMargin` — The margin for any page footer. The value is a number (inches).
- `headerMargin` — The margin for any page headers. The value is a number (inches).
- `hidden` — Should the worksheet be hidden? The value is a boolean.
- `horizontalCentre` — Should the worksheet be centered horizontally? The value is a boolean.
- `horizontalFreeze` — The row at which the pane is frozen vertically. The value is a number.
- `horizontalPrintResolution` — The horizontal print resolution. The value is a number.
- `leftMargin` — The left margin. The value is a number (inches).
- `normalMagnification` — The normal magnification factor (not zoom or scale factor). The value is a number (percentage).

- `orientation` — The paper orientation for printing this sheet. The value is a string that can be either "landscape" or "portrait".
- `pageBreakPreviewMagnification` — The page break preview magnification factor (not zoom or scale factors). The value is a number (percentage).
- `pageBreakPreviewMode` — Show page in preview mode? The value is a boolean.
- `pageStart` — The page number at which to commence printing. The value is a number.
- `paperSize` — The paper size to be used when printing this sheet. The value is a string that can be one of "a4", "a3", "letter", "legal" etc (see *jxl.format.PaperSize* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/PaperSize.html]).
- `password` — The password for this sheet. The value is a string.
- `passwordHash` — The password hash - used only when copying sheets. The value is a string.
- `printGridLines` — Should grid lines be printed? The value is a boolean.
- `printHeaders` — Should headers be printed? The value is a boolean.
- `sheetProtected` — Should the sheet be protected (read-only)? The value is a boolean.
- `recalculateFormulasBeforeSave` — Should the formulas be re-calculated when the sheet is saved? The value is a boolean. Default value is false.
- `rightMargin` — The right margin. The value is a number (inches).
- `scaleFactor` — The scale factor for this sheet to be used when printing. The value is a number (percent).
- `selected` — Should the sheet be selected when the workbook opens? The value is a boolean.
- `showGridLines` — Should gridlines be shown? The value is a boolean.
- `topMargin` — The top margin. The value is a number (inches).
- `verticalCentre` — Center verically? The value is a boolean.

- `verticalFreeze` — The row at which the pane is frozen vertically. The value is a number.
- `verticalPrintResolution` — The vertical print resolution. The value is a number.
- `zoomFactor` — The zoom factor. Do not confuse zoom factor (which relates to the on screen view) with scale factor (which refers to the scale factor when printing). The value is a number (percentage).

###

- `<e:printArea/>` — Zero or more print area definitions (see *#19.11. #printArea###printTitle###*).
- `<e:printTitle/>` — Zero or more print title definitions (see *#19.11. #printArea###printTitle###*).
- `<e:headerFooter/>` — Zero or more header/footer definitions (see *#19.10. #header###footer###*).
- 0##################*#19.12. ##########################*

#####

- `header`— Contents that will be placed at the top of the data block, above the column headers (if any).
- `footer`— Contents that will be placed at the bottom of the data block, below the column footers (if any).

```
<e:workbook>
  <e:worksheet name="foo" startColumn="1" startRow="1">
      <e:column value="#{personList}" var="person">
        <f:facet name="header">
          <e:cell value="Last name"/>
        </f:facet>
       <e:cell value="#{person.lastName}"/>
    </e:column>
  </e:worksheet>
<e:workbook>
```

###B2##############"foo"###################

19.5. column##

column####worksheet########cell###image###formula###hyperlink#####################################
#########

`<e:column>`	## • ## ### • `<e:cell/>` — Zero or more cells (see *#19.6. #cell###*). • `<e:formula/>` — Zero or more formulas (see *#19.7. #formula###*). • `<e:image/>` — Zero or more images (see *#19.8. #image###*). • `<e:hyperLink/>` — Zero or more hyperlinks (see *#19.9. #hyperlink###*). ##### • `header` — This facet can/will contain one `<e:cell>` , `<e:formula>` , `<e:image>` or `<e:hyperLink>` that will be used as header for the column. • `footer` — This facet can/will contain one `<e:cell>` , `<e:formula>` , `<e:image>` or `<e:hyperLink>` that will be used as footer for the column.

```
<e:workbook>
    <e:worksheet value="#{personList}" var="person">
      <e:column>
        <f:facet name="header">
          <e:cell value="Last name"/>
        </f:facet>
        <e:cell value="#{person.lastName}"/>
      </e:column>
    </e:worksheet>
<e:workbook>
```

##########

19.6. cell##

Cells are nested within columns (for iteration) or inside worksheets (for direct placement using the `column` and `row` attributes) and are responsible for outputting the value (usually through an EL-expression involving the `var`-attribute of the datatable. See *???*

`<e:cell>`	## • `column` — The column where to place the cell. The default is the internal counter. The value is a number. Note that the value is 0-based. • `row` — The row where to place the cell. The default is the internal counter. The value is number. Note that the value is 0-based. • `value` — The value to display. Usually an EL-expression referencing the var-attribute of the containing datatable. The value is a string. • `comment` — A comment to add to the cell. The value is a string. • `commentHeight` — The height of the comment. The value is a number (in pixels). • `commentWidth` — A width of the comment. The value is a number (in pixels). ### • 0#################*#19.6.1. #validation*########### ##### • ##

```
        <e:workbook>
          <e:worksheet
>
            <e:column value="#{personList}" var="person">
              <f:facet name="header">
                  <e:cell value="Last name"/>
```

```
                </f:facet>
                <e:cell value="#{person.lastName}"/>
              </e:column>
            </e:worksheet>
          </e:workbook
>
```

##########

19.6.1. validation##

Validations are nested inside cells or formulas. They add constrains for the cell data.

`<e:numericValidation`	##

- `value` — The limit (or lower limit where applicable) of the validation. The value is a number.
- `value2` — The upper limit (where applicable) of the validation. The value is a number.
- `condition` — The validation condition. The value is a string.
  - "equal" - #####value#####################
  - "greater_equal" - #####value###############################
  - "less_equal" - #####value###############################
  - "less_than" - #####value#####################
  - "not_equal" - #####value#########################
  - "between" - #####value##########value2#####################
  - "not_between" - #####value##########value2#####################

###

- ##

#####

	• ##

```
        <e:workbook>
          <e:worksheet>
            <e:column value="#{personList}" var="person"
>
              <e:cell value="#{person.age">
                <e:numericValidation condition="between" value="4"
                  value2="18"/>
              </e:cell>
            </e:column>
          </e:worksheet>
        </e:workbook
>
```

######4#18####################################

`<e:rangeValidation>`	## • `startColumn` — The starting column of the range of values to validate against. The value is a number. • `startRow` — The starting row of the range of values to validate against. The value is a number. • `endColumn` — The ending column of the range of values to validate against. The value is a number. • `endRow` — The ending row of the range of values to validate against. The value is a number. ### • ## ##### • ##

```
        <e:workbook>
          <e:worksheet>
            <e:column value="#{personList}" var="person"
>
                <e:cell value="#{person.position">
                <e:rangeValidation startColumn="0" startRow="0"
                  endColumn="0" endRow="10"/>
              </e:cell>
            </e:column>
          </e:worksheet>
        </e:workbook
>
```

######A1:A10##

`<e:listValidation>`	## • ## ### • 0####listValidationItem## ##### • ##

e:listValidation#######e:listValidationItem####################

`<e:listValidationIte##>`	 • `value` — A values to validate against. ### • ## ##### • ##

```
        <e:workbook>
          <e:worksheet>
            <e:column value="#{personList}" var="person"
>
              <e:cell value="#{person.position">
                <e:listValidation>
                  <e:listValidationItem value="manager"/>
                  <e:listValidationItem value="employee"/>
                </e:listValidation>
              </e:cell>
            </e:column>
          </e:worksheet>
        </e:workbook
>
```

#########"manager"####"employee"################################

19.6.2.

Format masks are defined in the mask attribute in cells or formulas. There are two types of format masks, one for numbers and one for dates

19.6.2.1.

##"format1"#"accounting_float"#########
jxl.write.NumberFormats [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/write/NumberFormats.html]##########

################################## *java.text.DecimalFormat* [http://java.sun.com/javase/6/docs/api/java/text/DecimalFormat.html]###########"0.00"################################

19.6.2.2.

When encountering a format mask, first it is checked if it is in internal form, e.g "format1", "format2" and so on (see *jxl.write.DateFormats* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/write/DateFormats.html]).

################################## *java.text.DateFormat* [http://java.sun.com/javase/6/docs/api/java/text/DateFormat.html]#############"dd.MM.yyyy"################################

19.7. formula##

formula####column##################worksheet#####`column`###`row##################################`
#cell###########formula###

The formula of the cell is placed in the `value` -attribute as a normal the Microsoft® Excel® spreadsheet application notation. Note that when doing cross-sheet formulas, the worksheets must exist before referencing a formula against them. The value is a string.

```
<e:workbook>
  <e:worksheet name="fooSheet">
    <e:cell column="0" row="0" value="1"/>
  </e:worksheet>
  <e:worksheet name="barSheet">
    <e:cell column="0" row="0" value="2"/>
    <e:formula column="0" row="1"
      value="fooSheet!A1+barSheet1!A1">
      <e:font fontSize="12"/>
    </e:formula>
  </e:worksheet>
</e:workbook>
```

####BarSheet#B1####FooSheet#BarSheet#A1#################

19.8. image##

image####column##################worksheet#####`startColumn/startRow`###`rowSpan/columnSpan`#########################span###

`<e:image>`	## • `startColumn` — The starting column of the image. The default is the internal counter. The value is a number. Note that the value is 0-based. • `startRow` — The starting row of the image. The default is the internal counter. The value is a number. Note that the value is 0-based. • `columnSpan` — The column span of the image. The default is one resulting in the default width of the image. The value is a float.

	• `rowSpan` — The row span of the image. The default is the one resulting in the default height of the image. The value is a float. • `URI` — The URI to the image. The value is a string. ### • ## ##### • ##

```
<e:workbook>
  <e:worksheet>
    <e:image startRow="0" startColumn="0" rowSpan="4"
      columnSpan="4" URI="http://foo.org/logo.jpg"/>
  </e:worksheet>
</e:workbook
>
```

#########URI####A1:E5##########

19.9. hyperlink##

hyperlink####column##################worksheet#####`startColumn/startRow`###`endColumn/endRow`##############################URI###############

`<e:hyperlink>`	## • `startColumn` — The starting column of the hyperlink. The default is the internal counter. The value is a number. Note that the value is 0-based. • `startRow` — The starting row of the hyperlink. The default is the internal counter. The value is a number. Note that the value is 0-based. • `endColumn` — The ending column of the hyperlink. The default is the internal counter. The value is a number. Note that the value is 0-based.

	• `endRow` — The ending row of the hyperlink. The default is the internal counter. The value is a number. Note that the value is 0-based. • `URL` — The URL to link. The value is a string. • `description` — The description of the link. The value is a string. ### • ## ##### • ##

```
        <e:workbook>
          <e:worksheet>
            <e:hyperLink startRow="0" startColumn="0" endRow="4"
              endColumn="4" URL="http://seamframework.org"
              description="The Seam Framework"/>
          </e:worksheet>
        </e:workbook
>
```

####seamframework.org####################A1:E5##########

19.10. header###footer##

header###footer###worksheet##########facet######facet##

`<e:header>`	## • ## ### • ## #####

	• `left` — The contents of the left header/footer part. • `center` — The contents of the center header/footer part. • `right` — The contents of the right header/footer part.

`<e:footer>`	## • ## ### • ## ##### • `left` — The contents of the left header/footer part. • `center` — The contents of the center header/footer part. • `right` — The contents of the right header/footer part.

facet##

#date#	############
#page_number#	##############
#time#	############
#total_pages#	############
#worksheet_name#	###########
#workbook_name#	###########
#bold#	################bold###########
#italics#	################italic###########
#underline#	###########underline###########
#double_underline#	##############double_underline###########
#outline#	######################outline###########
#shadow#	################shadow###########
#strikethrough#	##############strikethrough###########
#subscript#	################subscript###########
#superscript#	################superscript###########
#superscript#	##############font_name=Verdana###########

#font_size#	################font_size=12############

```
        <e:workbook>
          <e:worksheet
>
            <e:header>
              <f:facet name="left">
                This document was made on #date# and has #total_pages# pages
              </f:facet>
              <f:facet name="right">
                #time#
              </f:facet>
            </e:header>
          <e:worksheet>
        </e:workbook>
```

19.11. printArea###printTitle##

printArea###printTitle##

`<e:printArea>`	## • `firstColumn` — The column of the top-left corner of the area. The parameter is a number. Note that the value is 0-based. • `firstRow` — The row of the top-left corner of the area. The parameter is a number. Note that the value is 0-based. • `lastColumn` — The column of the bottom-right corner of the area. The parameter is a number. Note that the value is 0-based. • `lastRow` — The row of the bottom-right corner of the area. The parameter is a number. Note that the value is 0-based. ### • ## ##### • ##

```
        <e:workbook>
          <e:worksheet
>
            <e:printTitles firstRow="0" firstColumn="0"
              lastRow="0" lastColumn="9"/>
            <e:printArea firstRow="1" firstColumn="0"
              lastRow="9" lastColumn="9"/>
          </e:worksheet>
        </e:workbook>
```

####A1:A10##############B2:J10##############

19.12.

##############workbook#########################

19.12.1.

################

`<e:groupRows>`	## • `startRow` — The row to start the grouping at. The value is a number. Note that the value is 0-based. • `endRow` — The row to end the grouping at. The value is a number. Note that the value is 0-based. • `collapse` — Should the grouping be collapsed initially? The value is a boolean. ### • ## ##### • ##

`<e:groupColumns>`	##

	• `startColumn` — The column to start the grouping at. The value is a number. Note that the value is 0-based. • `endColumn` — The column to end the grouping at. The value is a number. Note that the value is 0-based. • `collapse` — Should the grouping be collapsed initially? The value is a boolean. ### • ## ##### • ##

```
        <e:workbook>
          <e:worksheet
>
            <e:groupRows startRow="4" endRow="9" collapse="true"/>
            <e:groupColumns startColumn="0" endColumn="9" collapse="false"/>
          </e:worksheet>
        </e:workbook>
```

groups rows 5 through 10 and columns 5 through 10 so that the rows are initially collapsed (but not the columns).

19.12.2.

##########

`<e:rowPageBreak>`	## • `row` — The row to break at. The value is a number. Note that the value is 0-based. ### • ##

	##### • ##

```
<e:workbook>
  <e:worksheet
>
    <e:rowPageBreak row="4"/>
  </e:worksheet>
</e:workbook
>
```

####5#############

19.12.3.

#########

`<e:mergeCells>`	## • `startRow` — The row to start the merging from. The value is a number. Note that the value is 0-based. • `startColumn` — The column to start the merging from. The value is a number. Note that the value is 0-based. • `endRow` — The row to end the merging at. The value is a number. Note that the value is 0-based. • `endColumn` — The column to end the merging at. The value is a number. Note that the value is 0-based. ### • ## ##### • ##

```
        <e:workbook>
          <e:worksheet>
            <e:mergeCells startRow="0" startColumn="0" endRow="9" endColumn="9"/>
          </e:worksheet>
        </e:workbook
>
```

####A1:J10##########

19.13.

######XHML##################JSF#################org.jboss.seam.excel.excelExporter.export##

```
      <h:form id="theForm">
        <h:dataTable id="theDataTable" value="#{personList.personList}"
          var="person">
          ...
        </h:dataTable>
      </h:form>
```

###Microsoft® Excel®#######################

```
      <h:commandLink
        value="Export"
        action="#{excelExporter.export('theForm:theDataTable')}"
      />
```

############################s:link###

##############*19.14. ##########*#########

19.14.

CSS##font#border#background####CSS##########

The CSS attributes cascade down from parent to children and within one tag cascades over the CSS classes referenced in the `styleClass` attributes and finally over the CSS attributes defined in the `style` attribute. You can place them pretty much anywhere but e.g. placing a column width setting in a cell nested within that column makes little sense.

If you have format masks or fonts that use special characters, such as spaces and semicolons, you can escape the css string with " characters like xls-format-mask:'$;$'

19.14.1.

##########e:link##############e:link####workbook############

`<e:link>`	## • `URL` — The URL to the stylesheet ### • ## ##### • ##

```
<e:workbook>
  <e:link URL="/css/excel.css"/>
</e:workbook
>
```

####"/css/excel.css"###################

19.14.2.

###XLS-CSS#########################

xls-font-family	##############OS#########################
xls-font-size	#################

xls-font-color	######### *jxl.format.Colour* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/Colour.html]##########
xls-font-bold	##########"true"####"false"#########
xls-font-italic	##########"true"####"false"#########
xls-font-script-style	########## *jxl.format.ScriptStyle* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/ScriptStyle.html]##########
xls-font-underline-style	########## *jxl.format.UnderlineStyle* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/UnderlineStyle.html]##########
xls-font-struck-out	##########"true"####"false"#########
xls-font	##########'Times New Roman'##########"italic"#"bold"#"struckout"########## Example style="xls-font: red bold italic 22 Verdana"

19.14.3.

This group of XLS-CSS attributes defines the borders of the cell

xls-border-left-color	########## *jxl.format.Colour* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/Colour.html]##########
xls-border-left-line-style	The border line style of the left edge of the cell (see *jxl.format.BorderLineStyle* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/BorderLineStyle.html]).
xls-border-left	##########style="xls-border-left: thick red"#########
xls-border-top-color	########## *jxl.format.Colour* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/Colour.html]##########
xls-border-top-line-style	The border line style of the top edge of the cell (see *jxl.format.BorderLineStyle* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/BorderLineStyle.html]).
xls-border-top	A shorthand for setting line style and color of the top edge of the cell, e.g style="xls-border-top: red thick"
xls-border-right-color	########## *jxl.format.Colour* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/Colour.html]##########
xls-border-right-line-style	The border line style of the right edge of the cell (see *jxl.format.BorderLineStyle* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/BorderLineStyle.html]).
xls-border-right	##########style="xls-border-right: thick red"#########

xls-border-bottom-color	############ *jxl.format.Colour* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/Colour.html]##########
xls-border-bottom-line-style	The border line style of the bottom edge of the cell (see *jxl.format.BorderLineStyle* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/BorderLineStyle.html]).
xls-border-bottom	################################style="xls-border-bottom: thick red"#########
xls-border	####################################style="xls-border: thick red"#########

19.14.4.

This group of XLS-CSS attributes defines the background of the cell

xls-background-color	The color of the background (see *jxl.format.BorderLineStyle* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/BorderLineStyle.html]).
xls-background-pattern	########## *jxl.format.Pattern* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/Pattern.html]##########
xls-background	##

19.14.5.

This group of XLS-CSS attributes defines the column widths etc.

xls-column-width	###############5000###########XHTML#########e:column##########
xls-column-widths	###############5000## Example style="xls-column-widths: 5000, 5000, *, 10000"
xls-column-autosize	####################"true"####"false"#########
xls-column-hidden	####################"true"####"false"#########
xls-column-export	Should the column be shown in export? Valid values are "true" and "false". Default is "true".

19.14.6.

This group of XLS-CSS attributes defines the cell properties

xls-alignment	################## *jxl.format.Alignment* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/Alignment.html]##########
xls-force-type	####################"general"#"number"#"text"#"date"#"formula"#"bool"##########
xls-format-mask	###############*#19.6.2. ################ #19.6.2. #######*

xls-indentation	##########################
xls-locked	##"true"####"false"#########
xls-orientation	############## *jxl.format.Orientation* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/Orientation.html]##########
xls-vertical-alignment	################### *jxl.format.VerticalAlignment* [http://jexcelapi.sourceforge.net/resources/javadocs/current/docs/jxl/format/VerticalAlignment.html]##########
xls-shrink-to-fit	############################"true"####"false"#########
xls-wrap	####################"true"####"false"#########

19.14.7.

################XHTML#########XLS-CSS################################`xls-column-widths`##########UIColumn#style#styleClass################

19.14.8.

TODO

19.14.9.

############CSS######################

- When using .xhtml documents, stylesheets must be referenced through the `<e:link>` tag
- ###############CSS#XLS-CSS##

19.15. Internationalization

There are only two resources bundle keys used, both for invalid data format and both take a parameter (the invalid value)

- `org.jboss.seam.excel.not_a_number` — When a value thought to be a number could not be treated as such
- `org.jboss.seam.excel.not_a_date` — When a value thought to be a date could not be treated as such

19.16.

The core of the the Microsoft® Excel® spreadsheet application functionality is based on the excellent JExcelAPI library which can be found on *http://jexcelapi.sourceforge.net/* [http://jexcelapi.sourceforge.net] and most features and possible limitations are inherited from here.

##################################Seam##JBoss Seam JIRA#"excel"#####################################

RSS####

YARFRAW [http://yarfraw.sourceforge.net/]#########Seam#RSS#######################RSS###################"##### #####"#########

20.1.

RSS#############################`WEB-INF/lib`#######`jboss-seam-rss.jar`#############RSS###RSS#########################
#Seam RSS ##############

Seam RSS######### ########Facelets###########

20.2.

`examples/rss` ################ RSS ##RSS###################

################################xhtml######

```
<r:feed
  xmlns="http://www.w3.org/1999/xhtml"
  xmlns:ui="http://java.sun.com/jsf/facelets"
  xmlns:r="http://jboss.com/products/seam/rss"
  title="#{rss.feed.title}"
  uid="#{rss.feed.uid}"
  subtitle="#{rss.feed.subtitle}"
  updated="#{rss.feed.updated}"
  link="#{rss.feed.link}">
  <ui:repeat value="#{rss.feed.entries}" var="entry">
    <r:entry
      uid="#{entry.uid}"
      title="#{entry.title}"
      link="#{entry.link}"
      author="#{entry.author}"
      summary="#{entry.summary}"
      published="#{entry.published}"
      updated="#{entry.updated}"
        />
   </ui:repeat>
</r:feed>
```

20.3.

####################################0####################

`<r:feed>`	## • `uid` — An optional unique feed id. The value is a string. • `title` — The title of the feed. The value is a string. • `subtitle` — The subtitle of the feed. The value is a string. • `updated` — When was the feed updated? The value is a date. • `link` — The link to the source of the information. The value is a string. • `feedFormat` — The feed format. The value is a string and defaults to ATOM1. Valid values are RSS10, RSS20, ATOM03 and ATOM10. ### • `0##############` ##### • `##`

20.4.

###########"headlines"#######

`<r:feed>`	## • `uid` — An optional unique entry id. The value is a string. • `title` — The title of the entry. The value is a string. • `link` — A link to the item. The value is a string. • `author` — The author of the story. The value is a string. • `summary` — The body of the story. The value is a string. • `textFormat` — The format of the body and title of the story. The value is a string and valid values are "text" and "html". Defaults to "html".

	• `published` — When was the story first published? The value is a date. • `updated` — When was the story updated? The value is a date. ### • ## ##### • ##

20.5.

The core of the RSs functionality is based on the YARFRAW library which can be found on *http://yarfraw.sourceforge.net/* and most features and possible limitations are inherited from here.

ATOM 1.0##########*ATOM1.0##* [http://atompub.org/2005/07/11/draft-ietf-atompub-format-10.html]#######

RSS 2.0##########*RSS 2.0##* [http://cyber.law.harvard.edu/rss/rss.html]#######

#####

Seam ## ##

########## `jboss-seam-mail.jar` ########## ## JAR ################## JSF ######### `mailSession` #################

examples/mail ### ################## ##

Seam################################ *#37.3.4. #Seam##########*##

21.1.

Seam#########Facelets##

```
<m:message xmlns="http://www.w3.org/1999/xhtml"
   xmlns:m="http://jboss.com/products/seam/mail"
   xmlns:h="http://java.sun.com/jsf/html">

   <m:from name="Peter" address="peter@example.com" />
   <m:to name="#{person.firstname} #{person.lastname}"
>#{person.address}</m:to>
   <m:subject
>Try out Seam!</m:subject>

   <m:body>
      <p
><h:outputText value="Dear #{person.firstname}" />,</p>
      <p
>You can try out Seam by visiting
      <a href="http://labs.jboss.com/jbossseam"
>http://labs.jboss.com/jbossseam</a
>.</p>
      <p
>Regards,</p>
      <p
>Pete</p>
   </m:body>

</m:message
>
```

<m:message> ############## Seam # email ################### <m:message> ###### ############ <m:from> ### ####### <m:to> ## (###Facelets ########### EL ##############)# ## <m:subject> #########

<m:body> # email ########## HTML ########## JSF ####################

######################## ########### m:message ########### mailSession # email ################ ########### Seam ############################

```
@In(create=true)
private Renderer renderer;

public void send() {
   try {
      renderer.render("/simple.xhtml");
      facesMessages.add("Email sent successfully");
  }
  catch (Exception e) {
      facesMessages.add("Email sending failed: " + e.getMessage());
  }
}
```

########################

21.1.1.

Seam ########################## ################## java #################

jboss-seam-mail.jar ###### #########

```
<m:attachment value="/WEB-INF/lib/jboss-seam-mail.jar"/>
```

Seam ################## ################## ######## jboss-seam-mail.jar ############## ########### fileName ############

```
<m:attachment value="/WEB-INF/lib/jboss-seam-mail.jar" fileName="this-is-so-cool.jar"/>
```

java.io.File, java.net.URL ##############

```
<m:attachment value="#{numbers}"/>
```

byte[] #### java.io.InputStream

```
<m:attachment value="#{person.photo}" contentType="image/png"/>
```

#######byte[]#java.io.InputStream###############MIME#########################

########## ############# <m:attachment> ###### Seam ## PDF ### JSF ### ##############

```
<m:attachment fileName="tiny.pdf">
   <p:document
>
      A very tiny PDF
   </p:document>
</m:attachment
>
```

#################### (## ##################)# <ui:repeat> ##############

```
<ui:repeat value="#{people}" var="person">
                        <m:attachment       value="#{person.photo}"       contentType="image/jpeg"
 fileName="#{person.firstname}_#{person.lastname}.jpg"/>
</ui:repeat
>
```

################################

```
<m:attachment
   value="#{person.photo}"
   contentType="image/jpeg"
   fileName="#{person.firstname}_#{person.lastname}.jpg"
   status="personPhoto"
   disposition="inline" />
<img src="cid:#{personPhoto.contentId}" />
```

cid:#{...}######################IETF###
ID####################

"status"##

21.1.2. HTML/Text

##################HTML###

```
<m:body>
   <f:facet name="alternative"
>Sorry, your email reader can't show our fancy email,
please go to http://labs.jboss.com/jbossseam to explore Seam.</f:facet>
</m:body
>
```

21.1.3.

##<ui:repeat>##############

```
<ui:repeat value="#{allUsers} var="user">
   <m:to name="#{user.firstname} #{user.lastname}" address="#{user.emailAddress}" />
</ui:repeat
>
```

21.1.4.

(############)# ##########
<ui:repeat> ############

```
<ui:repeat value="#{people}" var="p">
   <m:message>
      <m:from name="#{person.firstname} #{person.lastname}"
>#{person.address}</m:from>
      <m:to name="#{p.firstname}"
>#{p.address}</m:to>
         ...
   </m:message>
</ui:repeat
>
```

21.1.5.

The mail templating example shows that facelets templating just works with the Seam mail tags.

jboss.org # template.xhtml ###############

```
<m:message>
  <m:from name="Seam" address="do-not-reply@jboss.com" />
  <m:to name="#{person.firstname} #{person.lastname}"
>#{person.address}</m:to>
  <m:subject
>#{subject}</m:subject>
  <m:body>
    <html>
      <body>
        <ui:insert name="body"
>This is the default body, specified by the template.</ui:insert>
      </body>
    </html>
  </m:body>
</m:message
>
```

jboss.org # `templating.xhtml` ###############

```
<ui:param name="subject" value="Templating with Seam Mail"/>
<ui:define name="body">
   <p
>This example demonstrates that you can easily use <i
>facelets templating</i
> in email!</p>
</ui:define
>
```

`WEB-INF/lib`#jar###############Facelets#####################
Seam######`web.xml`##`.taglib.xml`####################################Seam####JSF################
`web.xml`#################

#######Facelets#JSF###

21.1.6.

Seam ###################### ######## JSF ######################### ####################

```
<m:message charset="UTF-8">
  ...
</m:message
```

```
>
```

########################
######################Facelets###

```
<?xml version="1.0" encoding="UTF-8"?>
```

21.1.7.

####################################Seam####################*21.5.* ########
##

```
<m:message xmlns:m="http://jboss.com/products/seam/mail"
   importance="low"
   requestReadReceipt="true"/>
```

Otherwise you can add any header to the message using the `<m:header>` tag:

```
<m:header name="X-Sent-From" value="JBoss Seam"/>
```

21.2.

If you are using EJB then you can use a MDB (Message Driven Bean) to receive email. JBoss provides a JCA adaptor — `mail-ra.rar` — but the version distributed with JBoss AS 4.x has a number of limitations (and isn't bundled in some versions) therefore we recommend using the `mail-ra.rar` distributed with Seam (it's in the `extras/` directory in the Seam bundle). `mail-ra.rar` should be placed in `$JBOSS_HOME/server/default/deploy`; if the version of JBoss AS you use already has this file, replace it.

##

JBoss AS 5.x and newer has mail-ra.rar applied the patches, so there is no need to copy the mail-ra.rar from Seam distribution.

#################

```
@MessageDriven(activationConfig={
   @ActivationConfigProperty(propertyName="mailServer", propertyValue="localhost"),
```

```
    @ActivationConfigProperty(propertyName="mailFolder", propertyValue="INBOX"),
    @ActivationConfigProperty(propertyName="storeProtocol", propertyValue="pop3"),
    @ActivationConfigProperty(propertyName="userName", propertyValue="seam"),
    @ActivationConfigProperty(propertyName="password", propertyValue="seam")
})
@ResourceAdapter("mail-ra.rar")
@Name("mailListener")
public class MailListenerMDB implements MailListener {

    @In(create=true)
    private OrderProcessor orderProcessor;

    public void onMessage(Message message) {
      // Process the message
      orderProcessor.process(message.getSubject());
    }

}
```

##########onMessage(Message message)#########
MDB#########Seam##################MDB###persistence context###############

You can find more information on `mail-ra.rar` at *http://www.jboss.org/community/wiki/InboundJavaMail*.

JBoss AS#############################mail-ra.rar##

21.3.

Seam#########################jboss-seam-mail.jar#WEB-INF/lib########## JBoss AS############################Seam############################## JBoss AS##############JavaMailAPI##############JBoss AS######API#Impl#lib/mail.jar###Seam##################Java Activation Framework#####lib/activation.jar###Seam#################################

##

The Seam Mail module requires the use of Facelets as the view technology. Future versions of the library may also support the use of JSP. Additionally, it requires the use of the seam-ui package.

mailSession########SMTP###########JavaMail#########

21.3.1. `mailSession`

JEE ########## ### Seam ###Session################ JNDI ########javaMail#Session##############

mailSession##################*#32.9. ##################*##########

21.3.1.1. JBoss AS # JNDI

The JBossAS `deploy/mail-service.xml` configures a JavaMail session binding into JNDI. The default service configuration will need altering for your network. *http://www.jboss.org/community/wiki/JavaMail* describes the service in more detail.

```
<components xmlns="http://jboss.com/products/seam/components"
   xmlns:core="http://jboss.com/products/seam/core"
   xmlns:mail="http://jboss.com/products/seam/mail">

   <mail:mail-session session-jndi-name="java:/Mail"/>

</components
>
```

Seam # JNDI ## `java:/Mail` #####mailSession##############

21.3.1.2. Seam ###Session

mailSession#`components.xml`############
#####`smtp.example.com`###Seam#smtp###################

```
<components xmlns="http://jboss.com/products/seam/components"
   xmlns:core="http://jboss.com/products/seam/core"
   xmlns:mail="http://jboss.com/products/seam/mail">

   <mail:mail-session host="smtp.example.com"/>

</components
>
```

21.4. Meldware

Meldware###############Seam##########*buni.org* [http://buni.org]#### Meldware#`SMTP`, `POP3`,

IMAP#webmail#####################################JEE####################Seam###########JBoss AS###############

##

Seam##########Meldware###
###Meldware###########buni.org [http://buni.org]################################

21.5.

http://jboss.com/products/seam/mail
##################### message ########## ######## Seam ###################

Facelets############################
#############JSF#########################Javascript#######################urlBase##########

<m:message>
###############

- importance —##########(low,normal,high)#######normal##
- precedence —######################
- requestReadReceipt —######false########## true#################From: #################
- urlBase — #######urlBase#requestContextPath############ <h:graphicImage>#########################
- messageId — #####ID###########

<m:from>
############From:)######################

- name —#########
- address — ###############

<m:replyTo>
############(Reply-to:)#### ##############################

- address — ###############

<m:to>
<m:to> ######### ##### <ui:repeat>.

- `name` — ######
- `address` — ###########

<m:cc>
email # CC ########## CC ########## <m:cc> ######### ##### <ui:repeat> ######################

- `name` — ######
- `address` — ###########

<m:bcc>
email # BCC ########## BCC ########## <m:bcc> ######### ##### <ui:repeat> ######################

- `name` — ######
- `address` — ###########

<m:header>
##################`X-Sent-From: JBoss Seam`)

- `name` — ########## (e.g. `X-Sent-From`).
- `value` —##############`JBoss Seam`#

<m:attachment>
##########

- `value` — ######:
  - `String` —`String`#########################
  - `java.io.File` —EL##`File` ###################
  - `java.net.URL` — EL##`URL`#################
  - `java.io.InputStream` — EL##`InputStream`##################`fileName`#`contentType`################
  - `byte[]`— EL## `byte[]`.##################`fileName`#`contentType`################

  ############

  - If this tag contains a `<p:document>` tag, the document described will be generated and attached to the email. A `fileName` should be specified.
  - If this tag contains other JSF tags a HTML document will be generated from them and attached to the email. A `fileName` should be specified.

- `fileName` — ###############
- `contentType` —#######MIME######

<m:subject>
###########

<m:body>
##########
`alternative`########HTML##########html##

- `type` — `plain`####################################HTML##########

##########

Seam makes it very easy to perform work asynchronously from a web request. When most people think of asynchronicity in Java EE, they think of using JMS. This is certainly one way to approach the problem in Seam, and is the right way when you have strict and well-defined quality of service requirements. Seam makes it easy to send and receive JMS messages using Seam components.

But for cases when you are simply want to use a worker thread, JMS is overkill. Seam layers a simple asynchronous method and event facility over your choice of *dispatchers*:

- `java.util.concurrent.ScheduledThreadPoolExecutor` (#####)
- EJB ######## (EJB 3.0 ####)
- Quartz

This chapter first covers how to leverage Seam to simplify JMS and then explains how to use the simpler asynchronous method and event facility.

22.1. Seam

Seam makes it easy to send and receive JMS messages to and from Seam components. Both the message publisher and the message receiver can be Seam components.

You'll first learn to setup a queue and topic message publisher and then look at an example that illustrates how to perform the message exchange.

22.1.1.

JMS ########### Seam ################### Seam # ################################## ## `QueueConnectionFactory` # `TopicConnectionFactory` ############################

Seam ########JBossMQ ################ `UIL2ConnectionFactory` ####### ####### JMS ############### `seam.properties#web.xml` #####`components.xml` ## `queueConnection.queueConnectionFactoryJndiName`# `topicConnection.topicConnectionFactoryJndiName` ######################

Seam ### `TopicPublisher` #### `QueueSender` ############# `components.xml` ## topic ### queue #############

```
<jms:managed-topic-publisher name="stockTickerPublisher"
                             auto-create="true"
                             topic-jndi-name="topic/stockTickerTopic"/>

<jms:managed-queue-sender name="paymentQueueSender"
```

```
                auto-create="true"
                queue-jndi-name="queue/paymentQueue"/>
```

22.1.2.

Now, you can inject a JMS `TopicPublisher` and `TopicSession` into any Seam component to publish an object to a topic:

```
@Name("stockPriceChangeNotifier")
public class StockPriceChangeNotifier
{
  @In private TopicPublisher stockTickerPublisher;

  @In private TopicSession topicSession;

  public void publish(StockPrice price)
  {
     try
     {
       stockTickerPublisher.publish(topicSession.createObjectMessage(price));
     }
     catch (Exception ex)
     {
       throw new RuntimeException(ex);
     }
  }
}
```

or to a queue:

```
@Name("paymentDispatcher")
public class PaymentDispatcher
{
  @In private QueueSender paymentQueueSender;

  @In private QueueSession queueSession;

  public void publish(Payment payment)
  {
    try
    {
      paymentQueueSender.send(queueSession.createObjectMessage(payment));
```

```
    }
    catch (Exception ex)
    {
      throw new RuntimeException(ex);
    }
  }
}
```

22.1.3. ######## Bean

You can process messages using any EJB 3 message-driven bean. The MDB can even be a Seam component, in which case it's possible to inject other event- and application- scoped Seam components. Here's an example of the payment receiver, which delegates to a payment processor.

##

You'll likely need to set the create attribute on the `@In` annotation to true (i.e. create = true) to have Seam create an instance of the component being injected. This isn't necessary if the component supports auto-creation (e.g., it's annotated with `@Autocreate`).

First, create an MDB to receive the message.

```
@MessageDriven(activationConfig = {
   @ActivationConfigProperty(
      propertyName = "destinationType",
      propertyValue = "javax.jms.Queue"
   ),
   @ActivationConfigProperty(
      propertyName = "destination",
      propertyValue = "queue/paymentQueue"
   )
})
@Name("paymentReceiver")
public class PaymentReceiver implements MessageListener
{
  @Logger private Log log;

  @In(create = true) private PaymentProcessor paymentProcessor;

  @Override
```

```
  public void onMessage(Message message)
  {
    try
    {
      paymentProcessor.processPayment((Payment) ((ObjectMessage) message).getObject());
    }
    catch (JMSException ex)
    {
      log.error("Message payload did not contain a Payment object", ex);
    }
  }
}
```

Then, implement the Seam component to which the receiver delegates processing of the payment.

```
@Name("paymentProcessor")
public class PaymentProcessor
{
  @In private EntityManager entityManager;

  public void processPayment(Payment payment)
  {
    // perhaps do something more fancy
    entityManager.persist(payment);
  }
}
```

If you are going to be performing transaction operations in your MDB, you should ensure that you are working with an XA datasource. Otherwise, it won't be possible to rollback database changes if the database transaction commits and a subsequent operation being performed by the message fails.

22.1.4.

Seam Remoting ########### JavaScript ## JMS ######################## ######## # 25. ######## ###########

22.2.

```
#####################################################
ScheduledThreadPoolExecutor                    ##################################
##########################################    EJB    3.0    ######################
components.xml ########
```

```
<async:timer-service-dispatcher/>
```

################## EJB ########################
################################### Seam
EJB 3.0
###################### ############################

######### ######## Quartz ################################ EAR # Quartz #####
JAR (lib #########) ####### application.xml # Java ###################
Quartz ############## Quartz ######################### seam.quartz.properties
############# ### #### components.xml ##### Quartz ########################

```
<async:quartz-dispatcher/>
```

ScheduledThreadPoolExecutor # Seam API# EJB3 Timer# Quartz Scheduler
############# components.xml # 1 ######################

22.2.1.

(########)
###
##AJAX#######################

EJB###

```
@Local
public interface PaymentHandler
{
   @Asynchronous
   public void processPayment(Payment payment);
}
```

(JavaBean ########## ################################)

########bean##########

```
@Stateless
@Name("paymentHandler")
public class PaymentHandlerBean implements PaymentHandler
{
   public void processPayment(Payment payment)
```

```
   {
      //do some work!
   }
}
```

####################

```
@Stateful
@Name("paymentAction")
public class CreatePaymentAction
{
   @In(create=true) PaymentHandler paymentHandler;
   @In Bill bill;

   public String pay()
   {
      paymentHandler.processPayment( new Payment(bill) );
      return "success";
   }
}
```

################################ ##
#########################

############@Duration#@Expiration# @IntervalDuration############
######################

```
@Local
public interface PaymentHandler
{
   @Asynchronous
   public void processScheduledPayment(Payment payment, @Expiration Date date);

   @Asynchronous
   public void processRecurringPayment(Payment payment,
                                       @Expiration Date date,
                                       @IntervalDuration Long interval)'
}
```

```
@Stateful
@Name("paymentAction")
```

```
public class CreatePaymentAction
{
   @In(create=true) PaymentHandler paymentHandler;
   @In Bill bill;

   public String schedulePayment()
   {
      paymentHandler.processScheduledPayment( new Payment(bill), bill.getDueDate() );
      return "success";
   }

   public String scheduleRecurringPayment()
   {
      paymentHandler.processRecurringPayment( new Payment(bill), bill.getDueDate(),
                                    ONE_MONTH );
      return "success";
   }
}
```

######## ################## Timer ###################### ##### Timer ######## EJB3 ################ EJB 3 ############ ####### ScheduledThreadPoolExecutor ##### ############ JDK ### Future ####### Quartz ############ QuartzTriggerHandle ####### ##########################

```
@Local
public interface PaymentHandler
{
   @Asynchronous
   public Timer processScheduledPayment(Payment payment, @Expiration Date date);
}
```

```
@Stateless
@Name("paymentHandler")
public class PaymentHandlerBean implements PaymentHandler
{
   @In Timer timer;

   public Timer processScheduledPayment(Payment payment, @Expiration Date date)
   {
      //do some work!
```

```
        return timer; //note that return value is completely ignored
    }

}
```

```
@Stateful
@Name("paymentAction")
public class CreatePaymentAction
{
    @In(create=true) PaymentHandler paymentHandler;
    @In Bill bill;

    public String schedulePayment()
    {
        Timer timer = paymentHandler.processScheduledPayment( new Payment(bill),
                                                bill.getDueDate() );
        return "success";
    }
}
```

##################################

22.2.2. Quartz

Quartz ####### (######################) ######## @Asynchronous# @Duration# @Expiration# @IntervalDuration ################ #### ############ Quartz ######## 3 ####################

@FinalExpiration ######################### QuartzTriggerHandle ######################

```
        @In QuartzTriggerHandle timer;

    // Defines the method in the "processor" component
    @Asynchronous
    public QuartzTriggerHandle schedulePayment(@Expiration Date when,
                            @IntervalDuration Long interval,
                            @FinalExpiration Date endDate,
                            Payment payment)
    {
        // do the repeating or long running task until endDate
```

```
    }

    ... ...

    // Schedule the task in the business logic processing code
    // Starts now, repeats every hour, and ends on May 10th, 2010
    Calendar cal = Calendar.getInstance ();
    cal.set (2010, Calendar.MAY, 10);
    processor.schedulePayment(new Date(), 60*60*1000, cal.getTime(), payment);
```

`#######  QuartzTriggerHandle  ##############  ###################  #####`
`###############  QuartzTriggerHandle  ####################`
`########################################`

```
QuartzTriggerHandle handle =
        processor.schedulePayment(payment.getPaymentDate(),
                                  payment.getPaymentCron(),
                                  payment);
       payment.setQuartzTriggerHandle( handle );
       // Save payment to DB

       // later ...

       // Retrieve payment from DB
       // Cancel the remaining scheduled tasks
       payment.getQuartzTriggerHandle().cancel();
```

`@IntervalCron` ######################## Unix cron ############## ##### ########## 3 ####### 2:10pm # 2:44pm ########

```
    // Define the method
    @Asynchronous
    public QuartzTriggerHandle schedulePayment(@Expiration Date when,
                                @IntervalCron String cron,
                                Payment payment)
    {
       // do the repeating or long running task
    }

    ... ...
```

```
// Schedule the task in the business logic processing code
QuartzTriggerHandle handle =
  processor.schedulePayment(new Date(), "0 10,44 14 ? 3 WED", payment);
```

`@IntervalBusinessDay` ##########X###################### #####
##############2#### 14:00 ######## ######## 2010 ##################################

```
// Define the method
@Asynchronous
public QuartzTriggerHandle schedulePayment(@Expiration Date when,
                    @IntervalBusinessDay NthBusinessDay nth,
                    Payment payment)
{
   // do the repeating or long running task
}

... ...

// Schedule the task in the business logic processing code
QuartzTriggerHandle handle =
  processor.schedulePayment(new Date(),
     new NthBusinessDay(2, "14:00", WEEKLY), payment);
```

`NthBusinessDay` ########################## `additionalHolidays`
########################## (## ######## ###############)#

```
public class NthBusinessDay implements Serializable
{
    int n;
    String fireAtTime;
    List <Date
> additionalHolidays;
    BusinessDayIntervalType interval;
    boolean excludeWeekends;
    boolean excludeUsFederalHolidays;

    public enum BusinessDayIntervalType { WEEKLY, MONTHLY, YEARLY }

    public NthBusinessDay ()
    {
```

```
      n = 1;
      fireAtTime = "12:00";
      additionalHolidays = new ArrayList <Date
> ();
      interval = BusinessDayIntervalType.WEEKLY;
      excludeWeekends = true;
      excludeUsFederalHolidays = true;
    }
    ... ...
}
```

`@IntervalDuration# @IntervalCron# @IntervalNthBusinessDay ##################
############## RuntimeException #########`

22.2.3.

`############################## ################## Events #
raiseAsynchronousEvent() ############## ######################
raiseTimedEvent() ######### schedule ##########
(################################### TimerSchedule #####)#
##################################
##`

22.2.4.

`### ##### java.util.concurrent
################################## EJB3 ##################
############################## Seam ######################`

`######## ##
org.jboss.seam.async.asynchronousExceptionHandler
##`

```
@Scope(ScopeType.STATELESS)
@Name("org.jboss.seam.async.asynchronousExceptionHandler")
public class MyAsynchronousExceptionHandler extends AsynchronousExceptionHandler {

  @Logger Log log;

  @In Future timer;

  @Override
  public void handleException(Exception exception) {
    log.debug(exception);
```

```
    timer.cancel(false);
  }

}
```

Here, for example, using `java.util.concurrent` dispatcher, we inject its control object and cancel all future invocations when an exception is encountered

`public void handleAsynchronousException(Exception exception);` #####################################

```
  public void handleAsynchronousException(Exception exception) {
    log.fatal(exception);
  }
```

#####

In almost all enterprise applications, the database is the primary bottleneck, and the least scalable tier of the runtime environment. People from a PHP/Ruby environment will try to tell you that so-called "shared nothing" architectures scale well. While that may be literally true, I don't know of many interesting multi-user applications which can be implemented with no sharing of resources between different nodes of the cluster. What these silly people are really thinking of is a "share nothing except for the database" architecture. Of course, sharing the database is the primary problem with scaling a multi-user application — so the claim that this architecture is highly scalable is absurd, and tells you a lot about the kind of applications that these folks spend most of their time working on.

##

##################1##########
Seam###

- ###
- ORM########Hibernate#####JPA##################2#############
  ##################################
  ##
  ##################################
  ######################2##
  ##Seam##########
- Seam#########################
  ##
- ###Seam#######################################Bean########EJB#################################
- ##################Seam################################
  ##
- #############JBossCache#JBoss POJO
  Cache#EHCache#######Seam#cacheProvider##################
  ##
- #############JSF######################
  ORM########2##
  ##

2################################ORM##########################cacheProvider########<s:cache>

23.1. Seam###########

#####cacheProvider#######################

JBoss Cache 1.x (suitable for use in JBoss 4.2.x or later and other containers)

`org.jboss.cache.TreeCache`

JBoss Cache 2.x #JBoss 5..x##################

`org.jboss.cache.Cache`

JBoss POJO Cache 1.x (suitable for use in JBoss 4.2.x or later and other containers)

`org.jboss.cache.aop.PojoCache`

EHCache ####################

`net.sf.ehcache.CacheManager`

###Java###

`cacheProvider`#####################jar###################

JBoss Cache 1.x

- `jboss-cache.jar` - JBoss Cache 1.4.1
- `jgroups.jar` - JGroups 2.4.1

JBoss Cache 2.x

- `jboss-cache.jar` - JBoss Cache 2.2.0
- `jgroups.jar` - JGroups 2.6.2

JBoss POJO Cache 1.x

- `jboss-cache.jar` - JBoss Cache 1.4.1
- `jgroups.jar` - JGroups 2.4.1
- `jboss-aop.jar` - JBoss AOP 1.5.0

EHCache

- `ehcache.jar` - EHCache 1.2.3

####

JBoss####################JBossCache#########################JBossCac wiki [http://wiki.jboss.org/wiki/JBossCache]##############

Seam#EAR##################jar#############EAR#################

JBossCache####################`treecache.xml`################################EJB JAR#####`WEB-INF/classes`####JBossCache##JBossCache############

`treecache.xml`######`examples/blog/resources/treecache.xml`######

EHCache############################

##############components.xml##################

```
<components xmlns="http://jboss.com/products/seam/components"
            xmlns:cache="http://jboss.com/products/seam/cache">
  <cache:jboss-cache-provider configuration="META-INF/cache/treecache.xml" />
</components
>
```

Seam####################################

```
@Name("chatroomUsers")
@Scope(ScopeType.STATELESS)
public class ChatroomUsers
{
   @In CacheProvider cacheProvider;

   @Unwrap
   public Set<String> getUsers() throws CacheException   {
      Set<String> userList = (Set<String>) cacheProvider.get("chatroom", "userList");
      if (userList==null) {
         userList = new HashSet<String>();
         cacheProvider.put("chatroom", "userList", userList);
      }
      return userList;
   }
}
```

################components.xml##########

```
<components xmlns="http://jboss.com/products/seam/components"
            xmlns:cache="http://jboss.com/products/seam/cache">
  <cache:jboss-cache-provider name="myCache" configuration="myown/cache.xml"/>
  <cache:jboss-cache-provider name="myOtherCache" configuration="myother/cache.xml"/>
</components
>
```

23.2.

Seam####################JSF#####################<s:cache>#####<s:cache>#####pojoCache#####

`<s:cache>`##################################
########blog#########blog############

```
<s:cache key="recentEntries-#{blog.id}" region="welcomePageFragments">
  <h:dataTable value="#{blog.recentEntries}" var="blogEntry">
    <h:column>
      <h3
>#{blogEntry.title}</h3>
      <div>
        <s:formattedText value="#{blogEntry.body}"/>
      </div>
    </h:column>
  </h:dataTable>
</s:cache
>
```

`key`###blog#########################`region`#####

`<s:cache>` ######## ################ (###### ### blog #########)
########################

```
public void post() {
   ...
   entityManager.persist(blogEntry);
   cacheProvider.remove("welcomePageFragments", "recentEntries-" + blog.getId() );
}
```

################################JBossCache#######################

Web####

Seam#JBossWS###########JEE
#Web##############Web##########Seam###################################Seam###Web#########

24.1.

Web#######Seam#################Seam###################################SOAP##############

############`standard-jaxws-endpoint-config.xml`##Web##########jar#####`META-INF`#################################SOAP################

```
<jaxws-config xmlns="urn:jboss:jaxws-config:2.0"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xmlns:javaee="http://java.sun.com/xml/ns/javaee"
              xsi:schemaLocation="urn:jboss:jaxws-config:2.0 jaxws-config_2_0.xsd">
  <endpoint-config>
    <config-name
>Seam WebService Endpoint</config-name>
    <pre-handler-chains>
      <javaee:handler-chain>
        <javaee:protocol-bindings
>##SOAP11_HTTP</javaee:protocol-bindings>
          <javaee:handler>
            <javaee:handler-name
>SOAP Request Handler</javaee:handler-name>
            <javaee:handler-class
>org.jboss.seam.webservice.SOAPRequestHandler</javaee:handler-class>
          </javaee:handler>
        </javaee:handler-chain>
    </pre-handler-chains>
  </endpoint-config>
</jaxws-config
>
```

24.2. ###Web####

###Web#############################
Seam###SOAP##############SOAP############conversation ID##conversation ID###Web##############

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
```

```
   xmlns:seam="http://seambay.example.seam.jboss.org/">
  <soapenv:Header>
    <seam:conversationId xmlns:seam='http://www.jboss.org/seam/webservice'
>2</seam:conversationId>
  </soapenv:Header>
  <soapenv:Body>
    <seam:confirmAuction/>
  </soapenv:Body>
</soapenv:Envelope
>
```

###SOAP###############SOAP###############conversation ID#####2####conversationId###############Web## ID###################Web###############

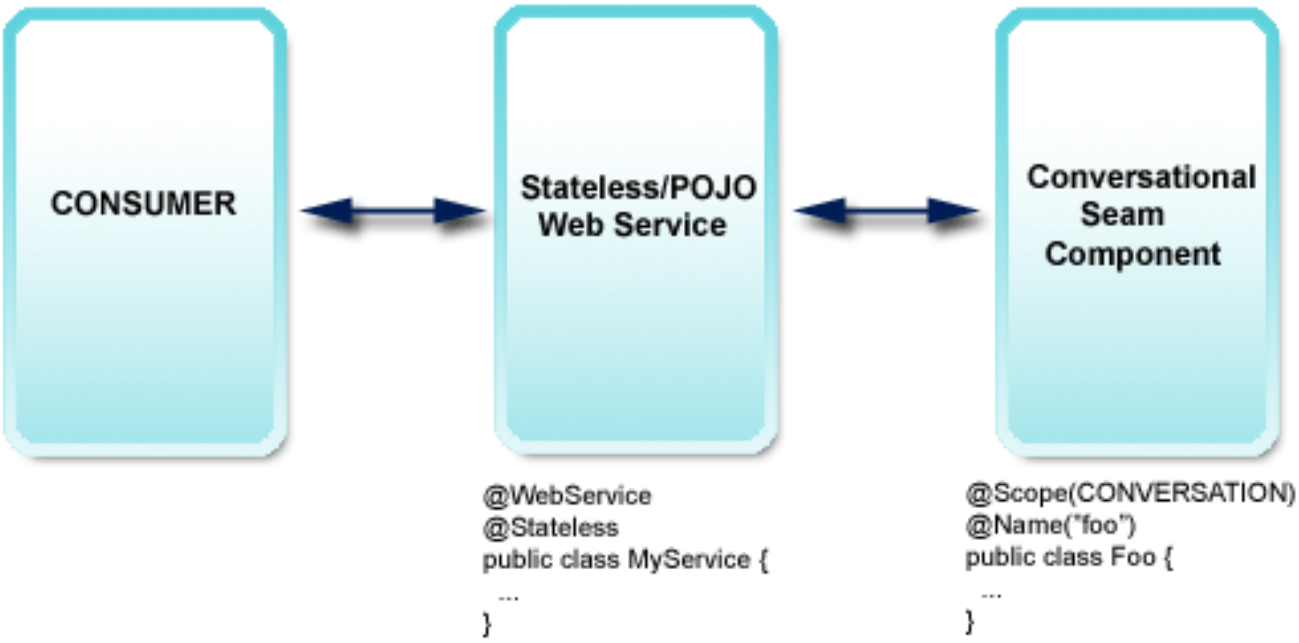
######### conversationId#######http://www.jboss.org/seam/webservice################################Seam#######conversation ID###

```
<env:Envelope xmlns:env='http://schemas.xmlsoap.org/soap/envelope/'>
  <env:Header>
    <seam:conversationId xmlns:seam='http://www.jboss.org/seam/webservice'
>2</seam:conversationId>
  </env:Header>
  <env:Body>
    <confirmAuctionResponse xmlns="http://seambay.example.seam.jboss.org/"/>
  </env:Body>
</env:Envelope
>
```

#######################conversationId###########

24.2.1.

Web##################Bean####POJO#########################Web##########Seam#################



###########Bean######Web########@Name#####Seam#########################Seam##############

24.3. Web######

Web#################################Seam#/examples#########seamBay#####################################Web#########################

```
@Stateless
@WebService(name = "AuctionService", serviceName = "AuctionService")
public class AuctionService implements AuctionServiceRemote
{
  @WebMethod
  public boolean login(String username, String password)
  {
    Identity.instance().setUsername(username);
    Identity.instance().setPassword(password);
    Identity.instance().login();
    return Identity.instance().isLoggedIn();
  }

  // snip
}
```

####Web#################Bean##JSR-181###########javax.jws######JWS#########################@WebService###############Web########login()#####@WebMethod########Web#####################@WebService########name#serviceName###

#########Web##########################Web#####################################Web########
#######AuctionServiceRemote########@WebMethod################login()#################

#############Web#########login()######Seam#####Identity#####################################

#################Web##########AuctionAction.createAuction()#########################

```
@WebMethod
public void createAuction(String title, String description, int categoryId)
{
   AuctionAction action = (AuctionAction) Component.getInstance(AuctionAction.class, true);
   action.createAuction();
   action.setDetails(title, description, categoryId);
}
```

####AuctionAction#########

```
@Begin
public void createAuction()
{
   auction = new Auction();
   auction.setAccount(authenticatedAccount);
   auction.setStatus(Auction.STATUS_UNLISTED);
   durationDays = DEFAULT_AUCTION_DURATION;
}
```

######Web#####################Seam#################################

24.4. RESTEasy ###RESTful HTTP Web####

Seam#JAX-RS ##(JSR 311)###RESTEasy################Seam#############”##”#############################

- RESTEasy ######################################
- SeamResourceServlet ###HTTP/REST ################web.xml ########
- Seam ######################Seam#####################################

24.4.1. RESTEasy

First, get the RESTEasy libraries and the `jaxrs-api.jar`, deploy them with the other libraries of your application. Also deploy the integration library, `jboss-seam-resteasy.jar`.

In seam-gen based projects, this can be done by appending `jaxrs-api.jar`, `resteasy-jaxrs.jar` and `jboss-seam-resteasy.jar` to the `deployed-jars.list` (war deployment) or `deployed-jars-ear.list` (ear deployment) file. For a JBoss Tools based project, copy the libraries mentioned above to the `EarContent/lib` (ear deployment) or `WebContent/WEB-INF/lib` (war deployment) folder and reload the project in the IDE.

######`@javax.ws.rs.Path`#########################HTTP##############Seam#######`SeamResourceServ`

- The URI starts with the host and context path of your application, e.g. `http://your.hostname/myapp`.
- Then the pattern mapped in `web.xml` for the `SeamResourceServlet`, e.g `/seam/resource` if you follow the common examples, is appended. Change this setting to expose your RESTful resources under a different base. Note that this is a global change and other Seam resources (e.g. `s:graphicImage` and `s:captcha`) are then also served under that base path.
- The RESTEasy integration for Seam then appends a configurable string to the base path, by default this is `/rest`. Hence, the full base path of your resources would e.g. be `/myapp/seam/resource/rest`. We recommend that you change this string in your application (details below). You could for example add a version number to prepare for a future REST API upgrade of your services (old clients would keep the old URI base): `/myapp/seam/resource/restv1`.
- Finally, the actual resource is available under the defined `@Path`, e.g. a resource mapped with `@Path("/customer")` would be available under `/myapp/seam/resource/rest/customer`.

As an example, the following resource definition would return a plaintext representation for any GET requests using the URI `http://your.hostname/myapp/seam/resource/rest/customer/123`:

```
@Path("/customer")
public class MyCustomerResource {

   @GET
   @Path("/{customerId}")
   @Produces("text/plain")
   public String getCustomer(@PathParam("customerId") int id) {
       return ...;
   }

}
```

No additional configuration is required; you do not have to edit `web.xml` or any other setting if these defaults are acceptable. However, you can configure RESTEasy in your Seam application. First import the `resteasy` namespace into your XML configuration (`components.xml`) file header:

```
<components
  xmlns="http://jboss.com/products/seam/components"
  xmlns:resteasy="http://jboss.com/products/seam/resteasy"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation=
    http://jboss.com/products/seam/resteasy
       http://jboss.com/products/seam/resteasy-2.2.xsd
    http://jboss.com/products/seam/components
       http://jboss.com/products/seam/components-2.2.xsd">
```

`############/rest################`

```
<resteasy:application resource-path-prefix="/restv1"/>
```

The full base path to your resources is now `/myapp/seam/resource/restv1/{resource}` - note that your `@Path` definitions and mappings do NOT change. This is an application-wide switch usually used for versioning of the HTTP interface.

`############@javax.ws.rs.Path#######@javax.ws.rs.ext.Provider#######Seam#####################`

```
<resteasy:application
    scan-providers="false"
    scan-resources="false"
    use-builtin-providers="true">

    <resteasy:resource-class-names>
       <value>org.foo.MyCustomerResource</value>
       <value>org.foo.MyOrderResource</value>
       <value>org.foo.MyStatelessEJBImplementation</value>
    </resteasy:resource-class-names>

    <resteasy:provider-class-names>
       <value>org.foo.MyFancyProvider</value>
    </resteasy:provider-class-names>

 </resteasy:application>
```

`use-built-in-providers`##RESTEasy####################################
#########JSON#JAXB#####################################

RESTEasy supports plain EJBs (EJBs that are not Seam components) as resources. Instead of configuring the JNDI names in a non-portable fashion in `web.xml` (see RESTEasy documentation), you can simply list the EJB implementation classes, not the business interfaces, in `components.xml` as shown above. Note that you have to annotate the `@Local` interface of the EJB with `@Path`, `@GET`, and so on - not the bean implementation class. This allows you to keep your application deployment-portable with the global Seam `jndi-pattern` switch on `<core:init/>`. Note that plain (non-Seam component) EJB resources will not be found even if scanning of resources is enabled, you always have to list them manually. Again, this whole paragraph is only relevant for EJB resources that are not also Seam components and that do not have an `@Name` annotation.

###############URI###########

```
<resteasy:application>

   <resteasy:media-type-mappings>
     <key>txt</key><value>text/plain</value>
   </resteasy:media-type-mappings>

   <resteasy:language-mappings>
     <key>deutsch</key><value>de-DE</value>
   </resteasy:language-mappings>

</resteasy:application>
```

`#########.txt.deutsch` ###URI############`Accept`###`Accept-Language`########`text/plain#de-DE`##############

24.4.2. Resources as Seam components

############################RESTEasy############################RESTEasy#########################
RS###

You can write resources as Seam components and benefit from the richer lifecycle management of Seam, and interception for bijection, security, and so on. Simply make your resource class a Seam component:

```
@Name("customerResource")
@Path("/customer")
public class MyCustomerResource {

   @In
   CustomerDAO customerDAO;
```

```
    @GET
    @Path("/{customerId}")
    @Produces("text/plain")
    public String getCustomer(@PathParam("customerId") int id) {
        return customerDAO.find(id).getName();
    }

}
```

An instance of `customerResource` is now handled by Seam when a request hits the server. This is a Seam JavaBean component that is `EVENT`-scoped, hence no different than the default JAX-RS lifecycle. You get full Seam injection and interception support, and all other Seam components and contexts are available to you. Currently also supported are `APPLICATION` and `STATELESS` resource Seam components. These three scopes allow you to create an effectively stateless Seam middle-tier HTTP request-processing application.

You can annotate an interface and keep the implementation free from JAX-RS annotations:

```
@Path("/customer")
public interface MyCustomerResource {

    @GET
    @Path("/{customerId}")
    @Produces("text/plain")
    public String getCustomer(@PathParam("customerId") int id);

}
```

```
@Name("customerResource")
@Scope(ScopeType.STATELESS)
public class MyCustomerResourceBean implements MyCustomerResource {

    @In
    CustomerDAO customerDAO;

    public String getCustomer(int id) {
        return customerDAO.find(id).getName();
    }

}
```

You can use `SESSION`-scoped Seam components. By default, the session will however be shortened to a single request. In other words, when an HTTP request is being processed by the RESTEasy integration code, an HTTP session will be created so that Seam components can utilize that context. When the request has been processed, Seam will look at the session and decide if the session was created only to serve that single request (no session identifier has been provided with the request, or no session existed for the request). If the session has been created only to serve this request, the session will be destroyed after the request!

Assuming that your Seam application only uses event, application, or stateless components, this procedure prevents exhaustion of available HTTP sessions on the server. The RESTEasy integration with Seam assumes by default that sessions are not used, hence anemic sessions would add up as every REST request would start a session that will only be removed when timed out.

If your RESTful Seam application has to preserve session state across REST HTTP requests, disable this behavior in your configuration file:

```
<resteasy:application destroy-session-after-request="false"/>
```

Every REST HTTP request will now create a new session that will only be removed by timeout or explicit invalidation in your code through `Session.instance().invalidate()`. It is your responsibility to pass a valid session identifier along with your HTTP requests, if you want to utilize the session context across requests.

`CONVERSATION`-scoped resource components and mapping of conversations to temporary HTTP resources and paths is planned but currently not supported.

EJB Seam components are supported as REST resources. Always annotate the local business interface, not the EJB implementation class, with JAX-RS annotations. The EJB has to be `STATELESS`.

Sub-resources as defined in the JAX RS specification, section 3.4.1, can also be Seam component instances:

```
@Path("/garage")
@Name("garage")
public class GarageService
{
  ...

  @Path("/vehicles")
  public VehicleService getVehicles() {
    return (VehicleService) Component.getInstance(VehicleService.class);
  }
```

```
}
```

##

RESTEasy components do not support hot redeployment. As a result, the components should never be placed in the `src/hot` folder. The `src/main` folder should be used instead.

##

Provider classes can currently not be Seam components. Although you can configure an `@Provider` annotated class as a Seam component, it will at runtime be managed by RESTEasy as a singleton with no Seam interception, bijection, etc. The instance will not be a Seam component instance. We plan to support Seam component lifecycle for JAX-RS providers in the future.

24.4.3. Securing resources

You can enable the Seam authentication filter for HTTP Basic and Digest authentication in `components.xml`:

```
<web:authentication-filter url-pattern="/seam/resource/rest/*" auth-type="basic"/>
```

See the Seam security chapter on how to write an authentication routine.

After successful authentication, authorization rules with the common `@Restrict` and `@PermissionCheck` annotations are in effect. You can also access the client `Identity`, work with permission mapping, and so on. All regular Seam security features for authorization are available.

24.4.4. Mapping exceptions to HTTP responses

Section 3.3.4 of the JAX-RS specification defines how checked or unchecked exceptions are handled by the JAX RS implementation. In addition to using an exception mapping provider as defined by JAX-RS, the integration of RESTEasy with Seam allows you to map exceptions to HTTP response codes within Seam's `pages.xml` facility. If you are already using `pages.xml` declarations, this is easier to maintain than potentially many JAX RS exception mapper classes.

Exception handling within Seam requires that the Seam filter is executed for your HTTP request. Ensure that you do filter *all* requests in your `web.xml`, not - as some Seam examples might show - a request URI pattern that doesn't cover your REST request paths. The following example intercepts *all* HTTP requests and enables Seam exception handling:

```
<filter>
   <filter-name>Seam Filter</filter-name>
   <filter-class>org.jboss.seam.servlet.SeamFilter</filter-class>
</filter>

<filter-mapping>
   <filter-name>Seam Filter</filter-name>
   <url-pattern>/*</url-pattern>
</filter-mapping>
```

To convert the unchecked `UnsupportedOperationException` thrown by your resource methods to a `501 Not Implemented` HTTP status response, add the following to your `pages.xml` descriptor:

```
<exception class="java.lang.UnsupportedOperationException">
   <http-error error-code="501">
      <message>The requested operation is not supported</message>
   </http-error>
</exception>
```

Custom or checked exceptions are handled the same:

```
<exception class="my.CustomException" log="false">
   <http-error error-code="503">
     <message>Service not available: #{org.jboss.seam.handledException.message}</message>
   </http-error>
</exception>
```

You do not have to send an HTTP error to the client if an exception occurs. Seam allows you to map the exception as a redirect to a view of your Seam application. As this feature is typically used for human clients (web browsers) and not for REST API remote clients, you should pay extra attention to conflicting exception mappings in `pages.xml`.

Note that the HTTP response still passes through the servlet container, so an additional mapping might apply if you have `<error-page>` mappings in your `web.xml` configuration. The HTTP status code would then be mapped to a rendered HTML error page with status `200 OK`!

24.4.5. Exposing entities via RESTful API

Seam makes it really easy to use a RESTful approach for accessing application data. One of the improvements that Seam introduces is the ability to expose parts of your SQL database for remote access via plain HTTP calls. For this purpose, the Seam/RESTEasy integration module provides

two components: `ResourceHome` and `ResourceQuery`, which benefit from the API provided by the Seam Application Framework (*# 13. Seam###############*). These components allow you to bind domain model entity classes to an HTTP API.

24.4.5.1. ResourceQuery

ResourceQuery exposes entity querying capabilities as a RESTful web service. By default, a simple underlying Query component, which returns a list of instances of a given entity class, is created automatically. Alternatively, the ResourceQuery component can be attached to an existing Query component in more sophisticated cases. The following example demonstrates how easily ResourceQuery can be configured:

```
<resteasy:resource-query
  path="/user"
  name="userResourceQuery"
  entity-class="com.example.User"/>
```

With this single XML element, a ResourceQuery component is set up. The configuration is straightforward:

- The component will return a list of `com.example.User` instances.
- The component will handle HTTP requests on the URI path `/user`.
- The component will by default transform the data into XML or JSON (based on client's preference). The set of supported mime types can be altered by using the `media-types` attribute, for example:

```
<resteasy:resource-query
  path="/user"
  name="userResourceQuery"
  entity-class="com.example.User"
  media-types="application/fastinfoset"/>
```

Alternatively, if you do not like configuring components using XML, you can set up the component by extension:

```
@Name("userResourceQuery")
@Path("user")
public class UserResourceQuery extends ResourceQuery<User>
{
}
```

Queries are read-only operations, the resource only responds to GET requests. Furthermore, ResourceQuery allows clients of a web service to manipulate the resultset of a query using the following path parameters:

Parameter name	Example	Description
start	/user?start=20	Returns a subset of a database query result starting with the 20th entry.
show	/user?show=10	Returns a subset of the database query result limited to 10 entries.

For example, you can send an HTTP GET request to `/user?start=30&show=10` to get a list of entries representing 10 rows starting with row 30.

##

RESTEasy uses JAXB to marshall entities. Thus, in order to be able to transfer them over the wire, you need to annotate entity classes with `@XMLRootElement`. Consult the JAXB and RESTEasy documentation for more information.

24.4.5.2. ResourceHome

Just as ResourceQuery makes Query's API available for remote access, so does ResourceHome for the Home component. The following table describes how the two APIs (HTTP and Home) are bound together.

 # 24.1.

HTTP method	Path	Function	ResourceHome method
GET	{path}/{id}	Read	getResource()
POST	{path}	Create	postResource()
PUT	{path}/{id}	Update	putResource()
DELETE	{path}/{id}	Delete	deleteResource()

- You can GET, PUT, and DELETE a particular user instance by sending HTTP requests to /user/{userId}
- Sending a POST request to `/user` creates a new user entity instance and persists it. Usually, you leave it up to the persistence layer to provide the entity instance with an identifier value and thus an URI. Therefore, the URI is sent back to the client in the `Location` header of the HTTP response.

The configuration of ResourceHome is very similar to ResourceQuery except that you need to explicitly specify the underlying Home component and the Java type of the entity identifier property.

```
<resteasy:resource-home
  path="/user"
  name="userResourceHome"
  entity-home="#{userHome}"
  entity-id-class="java.lang.Integer"/>
```

Again, you can write a subclass of ResourceHome instead of XML:

```
@Name("userResourceHome")
@Path("user")
public class UserResourceHome extends ResourceHome<User, Integer>
{

  @In
  private EntityHome<User
> userHome;

  @Override
  public Home<?, User
> getEntityHome()
  {
    return userHome;
  }
}
```

For more examples of ResourceHome and ResourceQuery components, take a look at the *Seam Tasks* example application, which demonstrates how Seam/RESTEasy integration can be used together with a jQuery web client. In addition, you can find more code example in the *Restbay* example, which is used mainly for testing purposes.

24.4.6. Testing resources and providers

Seam includes a unit testing utility class that helps you create unit tests for a RESTful architecture. Extend the `SeamTest` class as usual and use the `ResourceRequestEnvironment.ResourceRequest` to emulate HTTP requests/response cycles:

```
import org.jboss.seam.mock.ResourceRequestEnvironment;
import org.jboss.seam.mock.EnhancedMockHttpServletRequest;
import org.jboss.seam.mock.EnhancedMockHttpServletResponse;
```

```
import static org.jboss.seam.mock.ResourceRequestEnvironment.ResourceRequest;
import static org.jboss.seam.mock.ResourceRequestEnvironment.Method;

public class MyTest extends SeamTest {

  ResourceRequestEnvironment sharedEnvironment;

  @BeforeClass
  public void prepareSharedEnvironment() throws Exception {
     sharedEnvironment = new ResourceRequestEnvironment(this) {
        @Override
        public Map<String, Object> getDefaultHeaders() {
          return new HashMap<String, Object>() {{
             put("Accept", "text/plain");
          }};
        }
      };
  }

  @Test
  public void test() throws Exception
  {
    //Not shared: new ResourceRequest(new ResourceRequestEnvironment(this), Method.GET,
"/my/relative/uri)

    new ResourceRequest(sharedEnvironment, Method.GET, "/my/relative/uri)
    {
      @Override
      protected void prepareRequest(EnhancedMockHttpServletRequest request)
      {
        request.addQueryParameter("foo", "123");
        request.addHeader("Accept-Language", "en_US, de");
      }

      @Override
      protected void onResponse(EnhancedMockHttpServletResponse response)
      {
        assert response.getStatus() == 200;
        assert response.getContentAsString().equals("foobar");
      }

    }.run();
  }
```

```
}
```

This test only executes local calls, it does not communicate with the `SeamResourceServlet` through TCP. The mock request is passed through the Seam servlet and filters and the response is then available for test assertions. Overriding the `getDefaultHeaders()` method in a shared instance of `ResourceRequestEnvironment` allows you to set request headers for every test method in the test class.

Note that a `ResourceRequest` has to be executed in a `@Test` method or in a `@BeforeMethod` callback. You can not execute it in any other callback, such as `@BeforeClass`.

########

Seam ## Web ##### AJAX (Asynchronous Javascript and XML) ################################## ######### Seam ### ##################### - ############### AJAX ################################ ###### AJAX ### Web ######################## ###Seam Remoting ########################

25.1.

####################web.xml######Seam Resource###################

```
<servlet>
  <servlet-name
>Seam Resource Servlet</servlet-name>
  <servlet-class
>org.jboss.seam.servlet.SeamResourceServlet</servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name
>Seam Resource Servlet</servlet-name>
  <url-pattern
>/seam/resource/*</url-pattern>
</servlet-mapping
>
```

########Web#######JavaScript###

```
<script type="text/javascript" src="seam/resource/remoting/resource/remote.js"
></script
>
```

########### ################################
#####################################
###
############################ #### @Name("customerAction")
######################### Bean ###### ###################

```
<script type="text/javascript"
        src="seam/resource/remoting/interface.js?customerAction"
></script
```

```
>
```

##

```
<script type="text/javascript"
      src="seam/resource/remoting/interface.js?customerAction&accountAction"
></script
>
```

Javascript ####### s:remote
#####################################

```
 <s:remote include="customerAction,accountAction"/>
```

25.2. "Seam"######

########################## ### Seam Javascript ############### ######### remote.js
###################
Seam.Component ###################### Seam.Remoting ### ######################
#################

25.2.1. Hello World

Seam ######################## ######################## ######helloAction
######## Seam ################

```
@Stateless
@Name("helloAction")
public class HelloAction implements HelloLocal {
   public String sayHello(String name) {
      return "Hello, " + name;
   }
}
```

#################################### @WebRemote ###################
################################

```
@Local
public interface HelloLocal {
  @WebRemote
  public String sayHello(String name);
}
```

That's all the server-side code we need to write.

##

If you are performing a persistence operation in the method marked `@WebRemote` you will also need to add a `@Transactional` annotation to the method. Otherwise, your method would execute outside of a transaction without this extra hint.That's because unlike a JSF request, Seam does not wrap the remoting request in a transaction automatically.

Now for our web page - create a new page and import the `helloAction` component:

```
<s:remote include="helloAction"/>
```

##

```
<button onclick="javascript:sayHello()"
>Say Hello</button
>
```

##

```
<script type="text/javascript">
  //<![CDATA[

  function sayHello() {
    var name = prompt("What is your name?");
    Seam.Component.getInstance("helloAction").sayHello(name, sayHelloCallback);
  }

  function sayHelloCallback(result) {
    alert(result);
  }
```

```
  // ]]>
</script
>
```

######! ############################## ################################ ############### hello ######################## ################ Seam # `/examples/remoting/helloworld` ######### Hello World ##################

############################## ################################ Javascript ############## ## ##############

```
Seam.Component.getInstance("helloAction").sayHello(name, sayHelloCallback);
```

######### `Seam.Component.getInstance("helloAction")` ## `helloAction` ################"###"###### ############################## ################ `sayHello(name, sayHelloCallback);` ######

########################`sayHello`########## ########`name`######## #########`sayHelloCallback`## ######### `sayHello`################## ######Seam Remoting ###################### ### `sayHelloCallback` Javascript ################## ######################### ### `void` ############ ################## ################

`sayHelloCallback`############################## ################################

25.2.2. Seam.Component

`Seam.Component` Javascript ######## Seam ################################ #########`newInstance()`# `getInstance()`## ############### #########`newInstance()`
##################################`getInstance()`######################

25.2.2.1. Seam.Component.newInstance()

###############JavaBean ################################ ###################### ################ getter/setter ######### ## ######## Seam ###### ###############

```
@Name("customer")
@Entity
public class Customer implements Serializable
{
  private Integer customerId;
```

```
  private String firstName;
  private String lastName;

  @Column public Integer getCustomerId() {
    return customerId;
  }

  public void setCustomerId(Integer customerId} {
    this.customerId = customerId;
  }

  @Column public String getFirstName() {
    return firstName;
  }

  public void setFirstName(String firstName) {
    this.firstName = firstName;
  }

  @Column public String getLastName() {
    return lastName;
  }

  public void setLastName(String lastName) {
    this.lastName = lastName;
  }
}
```

########## Customer ########################

```
var customer = Seam.Component.newInstance("customer");
```

########## customer ######################

```
customer.setFirstName("John");
// Or you can set the fields directly
customer.lastName = "Smith";
```

25.2.2.2. Seam.Component.getInstance()

`getInstance()######` Seam `#####` Bean ##########################
################################## ##########################
##

##############
###`customer`###############`customerAction`########`saveCustomer()`#############

```
Seam.Component.getInstance("customerAction").saveCustomer(customer);
```

25.2.2.3. Seam.Component.getComponentName()

`null`

```
if (Seam.Component.getComponentName(instance) == "customer")
  alert("Customer");
else if (Seam.Component.getComponentName(instance) == "staff")
  alert("Staff member");
```

25.2.3. Seam.Remoting

Seam Remoting ####################`Seam.Remoting`############
######################## ####################

25.2.3.1. Seam.Remoting.createType()

######### Seam ########### JavaBean ##################
###################
`createType()` ############### ############### Java #############

```
var widget = Seam.Remoting.createType("com.acme.widgets.MyWidget");
```

25.2.3.2. Seam.Remoting.getTypeName()

######################### `Seam.Component.getComponentName()` ######
########################## ################ `null` ###### ########## Java
###########

25.3.

############## ######################## seam/resource/remoting/interface.js
################ s:remote ##############

```
<script type="text/javascript"
      src="seam/resource/remoting/interface.js?customerAction"
></script
>
```

```
<s:remote include="customerAction"/>
```

##########################
###

Bean
########################## ### ##############
###################################

################### Seam ################### ############# Bean ####
############### ##### ######## JavaBean ###################### ################
######## JavaBean (######### Bean # ###### Bean ######) #### ############
@WebRemote ############# ########################## ###### ##### Bean
########## EJB ### JavaBean ######################################

25.4.

Seam ############ ########## / ############################## ####### ID
############################

25.4.1. ## ID

########################## Seam Remoting############ID
################################ ID
##########Seam.Remoting.getContext().getConversationId######## #######ID
##########Seam.Remoting.getContext().setConversationId()########

##ID ##### Seam.Remoting.getContext().setConversationId()##########
##########################ID ############## ##########ID
########################ID#################### ################
#################

25.4.2.

ID ##################### ##### JavaScript ################### ID ########## ID

```
Seam.Remoting.getContext().setConversationId( #{conversation.id} );
```

25.5.

Seam ## ########################## ######################

`Seam.Remoting.startBatch()`#### ## ############ ######################## ######################
##################################`Seam.Remoting.executeBatch()`######
(##########)

`startBatch()`################################
`Seam.Remoting.cancelBatch()`##

####################`/examples/remoting/chatroom`########

25.6.

25.6.1. ####### /

############################
##

25.6.1.1. String

String ############## ### Javascript String ################

25.6.1.2. Number

Java ############################## ############## String ############## ######################## ########################## `Byte`# `Double`# `Float`# `Integer`# `Long`# `Short` ##################

25.6.1.3. Boolean

Boolean ############# Javascript# Boolean ################ Java boolean ########

25.6.2. JavaBeans

JavaBeans # Seam ####### JavaBean ######## ### non-component ############## ## (Seam ######### `Seam.Component.newInstance()`# ##### `Seam.Remoting.createType()`)#

##
##########
##

```
@Name("myAction")
public class MyAction implements MyActionLocal {
  public void doSomethingWithObject(Object obj) {
    // code
  }
}
```

`myWidget` ########################## `myAction` ################################## `myWidget` ####### ########## `MyWidget`

```
<s:remote include="myAction,myWidget"/>
```

`myWidget` ####### `Seam.Component.newInstance("myWidget")` ############ `myAction.doSomethingWithObject()`

25.6.3.

String ############## ######### Javascript Date ###################### ###### `java.util.Date` ###### (### `java.sql.Date` # `java.sql.Timestamp` ########)#

25.6.4. Enum

########## Enum # String ########## Enum ################## enum # String ######## ########################

```
@Name("paintAction")
public class paintAction implements paintLocal {
  public enum Color {red, green, blue, yellow, orange, purple};

  public void paint(Color color) {
    // code
```

```
 }
}
```

`paint()` ##### `red` ############## String ################

```
Seam.Component.getInstance("paintAction").paint("red");
```

############## #### ############ enum ########## (###################### enum ##########)# ######### String ##########

25.6.5.

25.6.5.1. Bag

Bag #### ####### #### ################### (### Map #### ################)# Javascript ################### ### ########## Javascript ######## ############################### #### Javascript ######## ################## ##### bag ##

25.6.5.2. Map

Javascript ### Map ################### ##### Map ### Seam Remoting ############### ######################### Map ######## ### `Seam.Remoting.Map` ##############

```
var map = new Seam.Remoting.Map();
```

Javascript #### Map ###################### `size()`# `isEmpty()`# `keySet()`# `values()`# `get(key)`# `put(key,  value)`# `remove(key)`# `contains(key)` ####### ########## Java ############ ##### `keySet()` ### `values()` ########### ################ Javascript Array

25.7.

##############
##
################ ############ Javascript ## `setDebug()` ##################

```
Seam.Remoting.setDebug(true);
```

components.xml ######################

```
<remoting:remoting debug="true"/>
```

############### setDebug(false) ######## ########################## Seam.Remoting.log(message) ########

25.8. Handling Exceptions

When invoking a remote component method, it is possible to specify an exception handler which will process the response in the event of an exception during component invocation. To specify an exception handler function, include a reference to it after the callback parameter in your JavaScript:

```
var callback = function(result) { alert(result); };
var exceptionHandler = function(ex) { alert("An exception occurred: " + ex.getMessage()); };
Seam.Component.getInstance("helloAction").sayHello(name, callback, exceptionHandler);
```

If you do not have a callback handler defined, you must specify `null` in its place:

```
var exceptionHandler = function(ex) { alert("An exception occurred: " + ex.getMessage()); };
Seam.Component.getInstance("helloAction").sayHello(name, null, exceptionHandler);
```

The exception object that is passed to the exception handler exposes one method, `getMessage()` that returns the exception message which is produced by the exception thrown by the `@WebRemote` method.

25.9.

################################ ### ############## ################

25.9.1.

"Please Wait..." ######################## Seam.Remoting.loadingMessage

```
Seam.Remoting.loadingMessage = "Loading...";
```

25.9.2.

############################## displayLoadingMessage() ### hideLoadingMessage() #################

```
// don't display the loading indicator
Seam.Remoting.displayLoadingMessage = function() {};
Seam.Remoting.hideLoadingMessage = function() {};
```

25.9.3.

############################## ################## displayLoadingMessage() # hideLoadingMessage() ##############################

```
Seam.Remoting.displayLoadingMessage = function() {
   // Write code here to display the indicator
 };

 Seam.Remoting.hideLoadingMessage = function() {
   // Write code here to hide the indicator
 };
```

25.10.

################ ################ XML ################ ################ Javascript ################## ######################## (Javabeans ##)# ## ############################## ## ################ ## ######### (############)# ##

Seam Remoting ## ######### @WebRemote ######## exclude ## ########### (#.#) ######################## String ########## ############### ##

######## Widget ##########

```
@Name("widget")
public class Widget
{
  private String value;
  private String secret;
  private Widget child;
  private Map<String,Widget> widgetMap;
  private List<Widget> widgetList;
```

```
  // getters and setters for all fields
}
```

25.10.1.

######### Widget ############## secret ############################### ###########

```
@WebRemote(exclude = {"secret"})
public Widget getWidget();
```

#secret############### secret ############ ####
Widget ### child
Widget ###### #### child # secret
##

```
@WebRemote(exclude = {"child.secret"})
public Widget getWidget();
```

25.10.2. Map

############################## Map# ##################### (List# Set# Array ##)#
######## ################### ##### Widget # widgetList ######### Widget ##########
Widget # secret ################

```
@WebRemote(exclude = {"widgetList.secret"})
public Widget getWidget();
```

Map ########################## Map ########### [key] ######## Map
################ [value] ################## ##### widgetMap ############## secret
####################

```
@WebRemote(exclude = {"widgetMap[value].secret"})
public Widget getWidget();
```

25.10.3.

###
############## (######## Seam ############) ############ (######## Seam
################) ########################

```
@WebRemote(exclude = {"[widget].secret"})
public Widget getWidget();
```

25.10.4.

##

```
@WebRemote(exclude = {"widgetList.secret", "widgetMap[value].secret"})
public Widget getWidget();
```

25.11. Transactional Requests

By default there is no active transaction during a remoting request, so if you wish to perform database updates during a remoting request, you need to annotate the `@WebRemote` method with `@Transactional`, like so:

```
@WebRemote @Transactional(TransactionPropagationType.REQUIRED)
public void updateOrder(Order order) {
  entityManager.merge(order);
}
```

25.12. JMS

Seam Remoting # JMS ############################ ################ JMS ################ ### ###################### ### ############################

25.12.1.

JMS ################ ## Seam Remoting ###################################### `seam.properties#` `web.xml` ### `components.xml` # `org.jboss.seam.remoting.messaging.subscriptionRegistry.allowedTopics` ###################

```
<remoting:remoting poll-timeout="5" poll-interval="1"/>
```

25.12.2. JMS Topic

JMS Topic

```
function subscriptionCallback(message)
{
  if (message instanceof Seam.Remoting.TextMessage)
    alert("Received message: " + message.getText());
}

Seam.Remoting.subscribe("topicName", subscriptionCallback);
```

Seam.Remoting.subscribe() ###################### ############ JMS Topic ##### ###################################

#################### ######################## ################################## instanceof ############ Seam.Remoting.TextMessage ### Seam.Remoting.ObjectMessage ################## TextMessage ### text ############## (####### getText() #####)# ObjectMessage ### value ################## (##### getValue() #########)#

25.12.3.

###################### Seam.Remoting.unsubscribe() #################

```
Seam.Remoting.unsubscribe("topicName");
```

25.12.4.

################################### #### Seam.Remoting.pollInterval ## ################################ ########## ######## 10 ######

########## Seam.Remoting.pollTimeout ## ################## ### ###### 0### ##

pollTimeout ################## ############################ ## ################## ####################

########## components.xml ################# ###### Javascript ############## ## ####################################

components.xml:

```
<remoting:remoting poll-timeout="5" poll-interval="1"/>
```

JavaScript:

```
// Only wait 1 second between receiving a poll response and sending the next poll request.
Seam.Remoting.pollInterval = 1;

// Wait up to 5 seconds on the server for new messages
Seam.Remoting.pollTimeout = 5;
```

Seam#Google Web Toolkit

########Ajax#########Google Web Toolkit (GWT)################Seam#GWT#########Seam############################

GWT#####GWT tools#################################### *http://code.google.com/webtoolkit/*##############GWT####################################

26.1.

Seam#########GWT##########################Seam#############################80#########*# Seam #### Seam ################*########

26.2.

GWT#######Seam###GW

```
public interface MyService extends RemoteService {
   public String askIt(String question);
 }
```

######################AsyncCallback ##########################

```
public interface MyServiceAsync extends RemoteService {
  public void askIt(String question, AsyncCallback callback);
}
```

##MyServiceAsync####################GWT#######################

######################Seam##############

```
@Name("org.jboss.seam.example.remoting.gwt.client.MyService")
public class ServiceImpl implements MyService {

  @WebRemote
  public String askIt(String question) {

    if (!validate(question)) {
      throw new IllegalStateException("Hey, this shouldn't happen, I checked on the client, " +
      "but its always good to double check.");
    }
    return "42. Its the real question that you seek now.";
```

```
  }

  public boolean validate(String q) {
    ValidationUtility util = new ValidationUtility();
    return util.isValid(q);
  }
}
```

The name of the seam component *must* match the fully qualified name of the GWT client interface (as shown), or the seam resource servlet will not be able to find it when a client makes a GWT call. The methods that are to be made accessible via GWT also need to be annotated with the `@WebRemote` annotation.

26.3. GWT####### Seam

##

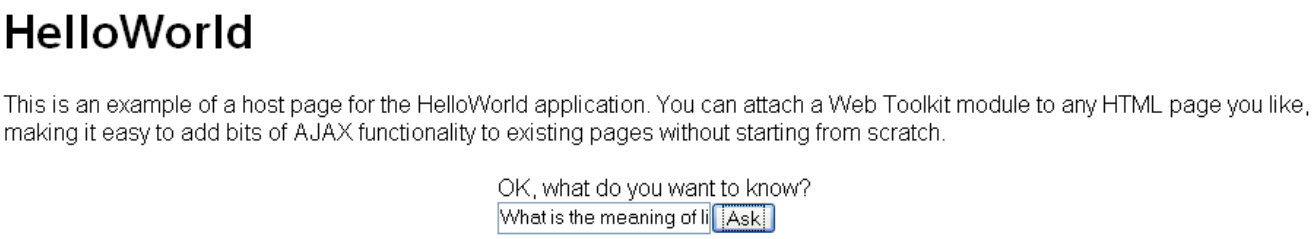
```
private MyServiceAsync getService() {
  String endpointURL = GWT.getModuleBaseURL() + "seam/resource/gwt";

  MyServiceAsync svc = (MyServiceAsync) GWT.create(MyService.class);
  ((ServiceDefTarget) svc).setServiceEntryPoint(endpointURL);
  return svc;
}
```

##

```
public class AskQuestionWidget extends Composite {
  private AbsolutePanel panel = new AbsolutePanel();

  public AskQuestionWidget() {
    Label lbl = new Label("OK, what do you want to know?");
    panel.add(lbl);
    final TextBox box = new TextBox();
    box.setText("What is the meaning of life?");
    panel.add(box);
    Button ok = new Button("Ask");
    ok.addClickListener(new ClickListener() {
      public void onClick(Widget w) {
        ValidationUtility valid = new ValidationUtility();
        if (!valid.isValid(box.getText())) {
```

```
            Window.alert("A question has to end with a '?'");
          } else {
            askServer(box.getText());
          }
        }
      });
      panel.add(ok);

      initWidget(panel);
    }

    private void askServer(String text) {
      getService().askIt(text, new AsyncCallback() {
        public void onFailure(Throwable t) {
          Window.alert(t.getMessage());
        }

        public void onSuccess(Object data) {
          Window.alert((String) data);
        }
      });
    }

    ...
```

##########`askServer()`##`askServer()`##(####################)####################



###############Seam############### `examples/remoting/gwt`############

26.4. GWT#Ant#####

GWT###############-Javascript####(##ant########GWT###GUI###############task#jar###########GWT(######################################

######ant###############

```
<taskdef uri="antlib:de.samaflost.gwttasks"
  resource="de/samaflost/gwttasks/antlib.xml"
  classpath="./lib/gwttasks.jar"/>

  <property file="build.properties"/>
```

########build.properties###########

```
gwt.home=/gwt_home_dir
```

########GWT#####################################

```
<!-- GWT#zg#)j#####gY.
  GWT##F4#oS6gYLGWT##%k######Y##LB#~Y -->
  <target name="gwt-compile">
    <!-- Sn4## GWT#### "re homing" WfD~Yng
    GWT####`Q#cfD~Y - URL#####kY#_#kLjcfD~Y -->
    <delete>
      <fileset dir="view"/>
    </delete>
    <gwt:compile outDir="build/gwt"
      gwtHome="${gwt.home}"
      classBase="${gwt.module.name}"
      sourceclasspath="src"/>
    <copy todir="view">
      <fileset dir="build/gwt/${gwt.module.name}"/>
    </copy>
  </target
>
```

################GWT###war#webapp###.
GWT#HTML#Javascript##########################gwt-compile
##########################GWT######################

GWT##############################GWT###

Spring Framework

The Spring integration (part of the Seam IoC module) allows easy migration of Spring-based projects to Seam and allows Spring applications to take advantage of key Seam features like conversations and Seam's more sophisticated persistence context management.

##! Spring ###### jboss-seam-ioc ############## ##################### seam-spring ##############

Spring # Seam ###################

- Seam ############## Spring Bean #########
- Spring Bean # Seam ################
- Spring Bean # Seam ############
- Spring Bean # Seam #################
- Seam ####### # Spring Web ##############
- Spring PlatformTransactionManagement #####
- Spring # `OpenEntityManagerInViewFilter` ### `OpenSessionInViewFilter` ###### Seam #########
- `@Asynchronous` ######## Spring `TaskExecutors` #####

27.1. Seam ######## Spring Bean

Seam ############## Spring Bean ############ `<seam:instance/>` ################## Seam ################## Seam ##### Spring Bean ####################

```
<beans xmlns="http://www.springframework.org/schema/beans"
   xmlns:seam="http://jboss.com/products/seam/spring-seam"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.springframework.org/schema/beans
                       http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
                       http://jboss.com/products/seam/spring-seam
                       http://jboss.com/products/seam/spring-seam-2.2.xsd">
```

Seam ############ Spring Bean

```
<bean id="someSpringBean" class="SomeSpringBeanClass" scope="prototype">
```

```
   <property name="someProperty">
      <seam:instance name="someComponent"/>
   </property>
</bean
>
```

############## EL #########

```
<bean id="someSpringBean" class="SomeSpringBeanClass" scope="prototype">
   <property name="someProperty">
      <seam:instance name="#{someExpression}"/>
   </property>
</bean
>
```

Seam ################ Spring Bean id # Spring Bean ########################

```
<seam:instance name="someComponent" id="someSeamComponentInstance"/>

<bean id="someSpringBean" class="SomeSpringBeanClass" scope="prototype">
   <property name="someProperty" ref="someSeamComponentInstance">
</bean>
```

##!

Seam## Spring ############ Seam ############## Spring ############################## ###### Spring Bean ################# ############### #### Bean ############################ Bean############ Bean ################### #### Seam ## ################### ### ### Spring Bean ############ ##### Spring Bean ## ##### ########## *(scope impedance)* ######## ################### Seam ### Spring ### Seam ##

`<seam:instance/>` ####### Seam ########################

```
<seam:instance id="seamManagedEM" name="someManagedEMComponent" proxy="true"/>

<bean id="someSpringBean" class="SomeSpringBeanClass">
   <property name="entityManager" ref="seamManagedEM">
</bean
```

```
>
```

Spring Bean ## Seam ############################## (Spring `OpenEntityManagerInView` ########## Seam ############################## *Spring # Seam ################ ################*)

27.2. Spring Bean # Seam

Spring Bean # Seam ################################ ### ###### 2 ######

- EL ###### Spring Bean #########
- Spring Bean # Seam ##########

######### 2 ################ ########## EL #### Spring Bean #########

Spring # `DelegatingVariableResolver` # Spring # JSF ######## Spring ##################### ## `VariableResolver` ### Bean ID #### EL ##### Spring Bean ################## `DelegatingVariableResolver` # `faces-config.xml` ##############

```
<application>
   <variable-resolver>
      org.springframework.web.jsf.DelegatingVariableResolver
   </variable-resolver>
</application
>
```

`@In` #### Spring Bean

```
@In("#{bookingService}")
private BookingService bookingService;
```

EL ################ Spring Bean ################# Seam # EL ################ Spring Bean ############## ############## ##################

27.3. Spring Bean # Seam

`<seam:component/>` ################ ### Spring Bean ## Seam ################## Seam ########### Bean ##### `<seam:component/>` ############

```
<bean id="someSpringBean" class="SomeSpringBeanClass" scope="prototype">
   <seam:component/>
</bean
```

```
>
```

######## <seam:component/># Bean ################## STATELESS############## #####FactoryBean ########### Spring Bean #### # Bean ###################### ######## class ################### ################ Seam ########################

Spring Bean #### Seam ############# <seam:component/> # scope ############### ##### Seam ##### STATELESS ####### Spring Bean # prototype ################ #### Spring Bean ########################### ##################

27.4. Seam ##### Spring Bean

Seam ########## Seam ######## Spring 2.0 ################################## ######### Seam ############ Spring Bean ################## #### Spring ### ###################### ### ############# Spring Bean ################## ####### Bean ################ Bean ###################

Spring Bean factory ###<seam:configure-scopes/> ######### #### Seam ################ Spring Bean ########### Spring Bean #### Seam ############ Bean ### scope ### Seam ##############

```
<!-- Only needs to be specified once per bean factory-->
<seam:configure-scopes/>

...

<bean id="someSpringBean" class="SomeSpringBeanClass"
 scope="seam.CONVERSATION"/>
```

configure-scopes #### prefix ############# ########################## (############## seam. ###)

################ Spring ############### @In ######################### ############################## @In(create=true) ############## Bean ####### configure-scopes # default-auto-create ###### Seam ######### Spring Bean ##################

########## Seam ##### Spring Bean ## <seam:instance/> ########### Spring Bean ############ #### ################################ Spring ############## Bean ##### <aop:scoped-proxy/> ###### #### Seam##### Spring Bean # <aop:scoped-proxy/> ############ ###### ### Bean # Seam #### Spring Bean################## <seam:instance/>###############

```
<bean                    id="someSpringBean"                    class="SomeSpringBeanClass"
 scope="seam.CONVERSATION"/>

...

<bean id="someSingleton">
   <property name="someSeamScopedSpringBean">
      <seam:instance name="someSpringBean" proxy="true"/>
   </property>
</bean
>
```

27.5. Spring # PlatformTransactionManagement

Spring ######################## ########### API (JPA# Hibernate# JDO# JTA ##) ######## ### Spring # Websphere # Weblogic ################### TransactionManagers ############### Spring ## REQUIRES_NEW # NOT_SUPPORTED ###### Java EE ###################### ######## Spring ######## ## [http://static.springframework.org/spring/docs/2.0.x/reference/transaction.html] ##########

Seam # Spring ######################## SpringTransaction ######################

```
<spring:spring-transaction platform-transaction-manager="#{transactionManager}"/>
```

`spring:spring-transaction` ################### Spring ######################

27.6. Spring # Seam

Seam ################################ EntityManager ################## ################################### ### `LazyInitializationException` ################ Spring # Web ###################################### (`OpenEntityManagerInViewFilter`)# ##### Spring # JPA ######################## Seam ################### Spring ###################### (`PersistenceAnnotationBeanPostProcessor`# `JpaTemplate` ##)#

Seam ###### JPA ######## Spring # Seam ########################### ##### Spring ####################################

########################

- Spring ######### Seam ####################
- Web ###### Seam ##################### (#### quartz #####)

- Seam ########## Spring ################# (#######################)

Spring ############### EntityManagerFactory ##### EntityManager ############# Seam ### EntityManagerFactory # Seam ##########################

```
<bean                                                    id="seamEntityManagerFactory"
 class="org.jboss.seam.ioc.spring.SeamManagedEntityManagerFactoryBean">
        <property name="persistenceContextName" value="entityManager"/>
</bean
>
```

#persistenceContextName## Seam ############################ ######### EntityManagerFactory ## Seam ############ unitName #### ######entityManager####### ## unitName ############### persistenceUnitName ############

```
<bean                                                    id="seamEntityManagerFactory"
 class="org.jboss.seam.ioc.spring.SeamManagedEntityManagerFactoryBean">
        <property name="persistenceContextName" value="entityManager"/>
      <property name="persistenceUnitName" value="bookingDatabase:extended"/>
</bean
>
```

EntityManagerFactory ##### Spring ########################## ##### Spring # `PersistenceAnnotationBeanPostProcessor`

```
<bean
 class="org.springframework.orm.jpa.support.PersistenceAnnotationBeanPostProcessor"/>
```

Spring ##### EntityManagerFactory ###### Seam ################### `defaultPersistenceUnitName` ##################### persistenctUnitName # `PersistenceAnnotationBeanPostProcessor` #############

`applicationContext.xml` ################

```
<bean                                                    id="entityManagerFactory"
 class="org.springframework.orm.jpa.LocalEntityManagerFactoryBean">
      <property name="persistenceUnitName" value="bookingDatabase"/>
</bean>
<bean                                                    id="seamEntityManagerFactory"
 class="org.jboss.seam.ioc.spring.SeamManagedEntityManagerFactoryBean">
        <property name="persistenceContextName" value="entityManager"/>
```

```
    <property name="persistenceUnitName" value="bookingDatabase:extended"/>
</bean>
<bean
 class="org.springframework.orm.jpa.support.PersistenceAnnotationBeanPostProcessor">
    <property name="defaultPersistenceUnitName" value="bookingDatabase:extended"/>
</bean
>
```

`component.xml` ################

```
<persistence:managed-persistence-context name="entityManager"
    auto-create="true" entity-manager-factory="#{entityManagerFactory}"/>
```

`JpaTemplate` ### `JpaDaoSupport` # Seam ################# Seam ##########################

```
<bean id="bookingService" class="org.jboss.seam.example.spring.BookingService">
    <property name="entityManagerFactory" ref="seamEntityManagerFactory"/>
</bean
>
```

27.7. Spring # Seam ### Hibernate

Seam Spring ##### Spring ######## Seam ## Hibernate ########################## ##### *JPA ##* ##########

Spring # JPA ###### Spring ##################EntityManagerFactory ###### EntityManager ## Spring ######################### ##### Seam Session ### proxy SessionFactory # Seam ### Hibernate ############################

```
<bean                                                    id="seamSessionFactory"
 class="org.jboss.seam.ioc.spring.SeamManagedSessionFactoryBean">
    <property name="sessionName" value="hibernateSession"/>
</bean
>
```

#sessionName## `persistence:managed-hibernate-session` ################ ##### SessionFactory ##### Spring ########################## ##### `SeamManagedSessionFactory` # getCurrentInstance() ############## `SessionFactory.getCurrentInstance()` ################

27.8. Seam ########### Spring Application Context

########## Spring # ApplicationContext ######## Spring # `ContextLoaderListener` ############## ############

- Spring ApplicationContext ## `SeamListener`#################
- Seam ###################### Spring ApplicationContext ###############

2 ############ Spring #### Spring ApplicationContext ##### Seam ################ ## Seam ############### `<spring:context-loader/>` #### `components.xml` ######## `config-locations` ######## Spring ################### ################ ###### `<spring:config-locations/>` ################## ######## `components.xml`

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:spring="http://jboss.com/products/seam/spring"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://jboss.com/products/seam/components
                    http://jboss.com/products/seam/components-2.2.xsd
                    http://jboss.com/products/seam/spring
                    http://jboss.com/products/seam/spring-2.2.xsd">

    <spring:context-loader config-locations="/WEB-INF/applicationContext.xml"/>

</components>
```

27.9. @Asynchronous # Spring # TaskExecutor

Spring ################ `TaskExecutor` ############## Spring Seam #### `@Asynchronous` ################## Spring # `TaskExecutor` ######## ########### `SpringTaskExecutorDispatchor` ################ Spring Bean ### taskExecutor ######

```
<spring:task-executor-dispatcher task-executor="#{springThreadPoolTaskExecutor}"/>
```

Spring # `TaskExecutor` ########################### ##### Seam `Dispatcher` ######################################

```
<!-- Install a ThreadPoolDispatcher to handle scheduled asynchronous event -->
<core:thread-pool-dispatcher name="threadPoolDispatcher"/>
```

```
<!-- Install the SpringDispatcher as default -->
<spring:task-executor-dispatcher task-executor="#{springThreadPoolTaskExecutor}"
 schedule-dispatcher="#{threadPoolDispatcher}"/>
```

Guice integration

Google Guice is a library that provides lightweight dependency injection through type-safe resolution. The Guice integration (part of the Seam IoC module) allows use of Guice injection for all Seam components annotated with the `@Guice` annotation. In addition to the regular bijection that Seam performs (which becomes optional), Seam also delegates to known Guice injectors to satisfy the dependencies of the component. Guice may be useful to tie non-Seam parts of large or legacy applications together with Seam.

##

The Guice integration is bundled in the jboss-seam-ioc library. This dependency is required for all integration techniques covered in this chapter. You will also need the Guice JAR file on the classpath.

28.1. Creating a hybrid Seam-Guice component

The goal is to create a hybrid Seam-Guice component. The rule for how to do this is very simple. If you want to use Guice injection in your Seam component, annotate it with the `@Guice` annotation (after importing the type `org.jboss.seam.ioc.guice.Guice`).

```
@Name("myGuicyComponent")
@Guice public class MyGuicyComponent
{
  @Inject MyObject myObject;
  @Inject @Special MyObject mySpecialObject;
  ...
}
```

This Guice injection will happen on every method call, just like with bijection. Guice injects based on type and binding. To satisfy the dependencies in the previous example, you might have bound the following implementations in a Guice module, where `@Special` is an annotation you define in your application.

```
public class MyGuicyModule implements Module
{
  public void configure(Binder binder)
  {
    binder.bind(MyObject.class)
      .toInstance(new MyObject("regular"));

    binder.bind(MyObject.class).annotatedWith(Special.class)
```

```
        .toInstance(new MyObject("special"));
  }
}
```

Great, but which Guice injector will be used to inject the dependencies? Well, you need to perform some setup first.

28.2. Configuring an injector

You tell Seam which Guice injector to use by hooking it into the injection property of the Guice initialization component in the Seam component descriptor (components.xml):

```
<components xmlns="http://jboss.com/products/seam/components"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:guice="http://jboss.com/products/seam/guice"
  xsi:schemaLocation="
    http://jboss.com/products/seam/guice
    http://jboss.com/products/seam/guice-2.2.xsd
    http://jboss.com/products/seam/components
    http://jboss.com/products/seam/components-2.2.xsd">

  <guice:init injector="#{myGuiceInjector}"/>

</components>
```

`myGuiceInjector` must resolve to a Seam component that implements the Guice `Injector` interface.

Having to create an injector is boiler-plate code, though. What you really want to be able to do is simply hook up Seam to your Guice modules. Fortunately, there is a built-in Seam component that implements the `Injector` interface to do exactly that. You can configure it in the Seam component descriptor with this additional stanza.

```
<guice:injector name="myGuiceInjector">
  <guice:modules>
    <value>com.example.guice.GuiceModule1</value>
    <value>com.example.guice.GuiceModule2</value>
  </guice:modules>
</guice:injector>
```

Of course you can also use an injector that is already used in other, possibly non-Seam part of you application. That's one of the main motivations for creating this integration. Since the injector

is defined with EL expression, you can obtain it in whatever way you like. For instance, you may use the Seam factory component pattern to provide injector.

```
@Name("myGuiceInjectorFactory")
public InjectorFactory
{
  @Factory(name = "myGuiceInjector", scope = APPLICATION, create = true)
  public Injector getInjector()
  {
    // Your code that returns injector
  }
}
```

28.3. Using multiple injectors

By default, an injector configured in the Seam component descriptor is used. If you really need to use multiple injectors (AFAIK, you should use multiple modules instead), you can specify different injector for every Seam component in the `@Guice` annotation.

```
@Name("myGuicyComponent")
@Guice("myGuiceInjector")
public class MyGuicyComponent
{
  @Inject MyObject myObject;
  ...
}
```

That's all there is to it! Check out the guice example in the Seam distribution to see the Seam Guice integration in action!

Hibernate Search

29.1.

Apache Lucene™ ## Apache Lucene ####### Hibernate Search##

Hibernate Search has been designed to integrate nicely and as naturally as possible with JPA and Hibernate. As a natural extension, JBoss Seam provides an Hibernate Search integration.

Please refer to the *Hibernate Search documentation* [] for information specific to the Hibernate Search project.

29.2.

Hibernate Search ##`META-INF/persistence.xml` ### `hibernate.cfg.xml` ##################

Hibernate Search ##

```
<persistence-unit name="sample">
  <jta-data-source>java:/DefaultDS</jta-data-source>
  <properties>
    [...]
    <!-- use a file system based index -->
    <property name="hibernate.search.default.directory_provider"
      value="org.hibernate.search.store.FSDirectoryProvider"/>
    <!-- directory where the indexes will be stored -->
    <property name="hibernate.search.default.indexBase"
      value="/Users/prod/apps/dvdstore/dvdindexes"/>
  </properties>
</persistence-unit>
```

If you plan to target Hibernate Annotations or EntityManager 3.2.x (embedded into JBoss AS 4.2.x and later), you also need to configure the appropriate event listeners.

```
<persistence-unit name="sample">
  <jta-data-source>java:/DefaultDS</jta-data-source>
  <properties>
    [...]
```

```
    <!-- use a file system based index -->
    <property name="hibernate.search.default.directory_provider"
            value="org.hibernate.search.store.FSDirectoryProvider"/>
    <!-- directory where the indexes will be stored -->
    <property name="hibernate.search.default.indexBase"
            value="/Users/prod/apps/dvdstore/dvdindexes"/>

    <property name="hibernate.ejb.event.post-insert"
            value="org.hibernate.search.event.FullTextIndexEventListener"/>
    <property name="hibernate.ejb.event.post-update"
            value="org.hibernate.search.event.FullTextIndexEventListener"/>
    <property name="hibernate.ejb.event.post-delete"
            value="org.hibernate.search.event.FullTextIndexEventListener"/>

  </properties>
</persistence-unit>
```

##

It is not longer necessary the register the event listeners if Hibernate Annotations or EntityManager 3.3.x are used. When using Hibernate Search 3.1.x more eventlisteners are needed, but these are registered automatically by Hibernate Annotations; refer to the Hibernate Search reference for configuring it without EntityManager and Annotations.

################ jar ##############:

- hibernate-search.jar
- hibernate-commons-annotations.jar
- lucene-core.jar

`hibernate-commons-annotations.jar` is not needed in JBossAS6. Some Hibernate Search extensions require additional dependencies, a commonly used is `hibernate-search-analyzers.jar`. For details, see the *Hibernate Search documentation* [http://www.hibernate.org/subprojects/search/docs] for details.

##

EAR ################`application.xml`

29.3.

Hibernate Search uses annotations to map entities to a Lucene index, check the *reference documentation* [http://www.hibernate.org/subprojects/search/docs] for more informations.

Hibernate Search ##API # JPA/Hibernate ######################HQL ## API ##`FullTextSession` API ###(Hibernate # `Session` ########)#

Hibernate Search #########JBoss Seam # FullTextSession #######

```
@Stateful
@Name("search")
public class FullTextSearchAction implements FullTextSearch, Serializable {

  @In FullTextSession session;

  public void search(String searchString) {
    org.apache.lucene.search.Query luceneQuery = getLuceneQuery();
    org.hibernate.Query query session.createFullTextQuery(luceneQuery, Product.class);
    searchResults = query
        .setMaxResults(pageSize + 1)
        .setFirstResult(pageSize * currentPage)
        .list();
  }
  [...]
}
```

##

`FullTextSession` # `org.hibernate.Session` ############ Hibernate Session #################

Java Persistance API ############################

```
@Stateful
@Name("search")
public class FullTextSearchAction implements FullTextSearch, Serializable {

  @In FullTextEntityManager em;

  public void search(String searchString) {
```

```
    org.apache.lucene.search.Query luceneQuery = getLuceneQuery();
    javax.persistence.Query query = em.createFullTextQuery(luceneQuery, Product.class);
    searchResults = query
        .setMaxResults(pageSize + 1)
        .setFirstResult(pageSize * currentPage)
        .getResultList();
  }
  [...]
}
```

Hibernate Search #########JBoss Seam # `FullTextEntityManager` #######`FullTextEntityManager` ################ `EntityManager` ################`FullTextSession` # `Session` #######

EJB 3.0 Session ###########Bean(Message Driven Bean) ###########(#####@PersistenceContext ##########)###### `EntityManager` ######## `FullTextEntityManager` ## `FullTextEntityManager` ############################

```
@Stateful
@Name("search")
public class FullTextSearchAction implements FullTextSearch, Serializable {

  @PersistenceContext EntityManager em;

  public void search(String searchString) {
    org.apache.lucene.search.Query luceneQuery = getLuceneQuery();
    FullTextEntityManager ftEm = (FullTextEntityManager) em;
    javax.persistence.Query query = ftEm.createFullTextQuery(luceneQuery, Product.class);
    searchResults = query
        .setMaxResults(pageSize + 1)
        .setFirstResult(pageSize * currentPage)
        .getResultList();
  }
  [...]
}
```

##

For people accustomed to Hibernate Search out of Seam, note that using `Search.getFullTextSession` is not necessary.

DVDStore # JBoss Seam ######## blog ############Hibernate Search ###################

Seam #### Seam

Configuration is a very boring topic and an extremely tedious pastime. Unfortunately, several lines of XML are required to integrate Seam into your JSF implementation and servlet container. There's no need to be too put off by the following sections; you'll never need to type any of this stuff yourself, since you can just use seam-gen to start your application or you can copy and paste from the example applications!

30.1. Seam

JSF # Seam

30.1.1. Seam # JSF#

faces

```
<servlet>
   <servlet-name
>Faces Servlet</servlet-name>
   <servlet-class
>javax.faces.webapp.FacesServlet</servlet-class>
   <load-on-startup
>1</load-on-startup>
</servlet>

<servlet-mapping>
   <servlet-name
>Faces Servlet</servlet-name>
   <url-pattern
>*.seam</url-pattern>
</servlet-mapping
>
```

(## URL ##################)

Seam ## `web.xml`##################

```
<listener>
   <listener-class
>org.jboss.seam.servlet.SeamListener</listener-class>
</listener
>
```

######## Seam ##

JSF ###### Seam ###################################### ## `web.xml` ###########

```
<context-param>
   <param-name
>javax.faces.STATE_SAVING_METHOD</param-name>
   <param-value
>client</param-value>
</context-param
>
```

############### JSF ################ Seam # JSF ############ PAGE ############ ################### JSF-RI ############### PAGE ##### Bean ## ######## ###################### PAGE ############################## ################################## (######## [https://javaserverfaces-spec-public.dev.java.net/issues/show_bug.cgi?id=295] ###)# ##### JSF######################## ####################

```
<context-param>
   <param-name>com.sun.faces.serializeServerState</param-name>
   <param-value>true</param-value>
</context-param>
```

30.1.2. Using Facelets

If you want follow our advice and use Facelets instead of JSP, add the following lines to `faces-config.xml`:

```
<application>
   <view-handler
>com.sun.facelets.FaceletViewHandler</view-handler>
</application
>
```

`web.xml`###########

```
<context-param>
```

```
   <param-name
>javax.faces.DEFAULT_SUFFIX</param-name>
   <param-value
>.xhtml</param-value>
</context-param
>
```

If you are using facelets in JBoss AS, you'll find that Facelets logging is broken (the log messages don't make it to the server log). Seam provides a bridge to fix this, to use it copy `lib/interop/jboss-seam-jul.jar` to `$JBOSS_HOME/server/default/deploy/jboss-web.deployer/jsf-libs/` and include the `jboss-seam-ui.jar` in the `WEB-INF/lib` of your application. The Facelets logging catagories are itemized in the *Facelets Developer Documentation* [https://facelets.dev.java.net/nonav/docs/dev/docbook.html#config-logging].

30.1.3. Seam

Seam ########## Seam Remoting ###### (##########) # JSF # UI ######################## Seam ############### `web.xml` ############

```
<servlet>
  <servlet-name
>Seam Resource Servlet</servlet-name>
  <servlet-class
>org.jboss.seam.servlet.SeamResourceServlet</servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name
>Seam Resource Servlet</servlet-name>
  <url-pattern
>/seam/resource/*</url-pattern>
</servlet-mapping
>
```

30.1.4. Seam##########

Seam ############################## #### ######################## Seam ################ Seam ## ############ `web.xml` ##########################

```
<filter>
   <filter-name
```

```
>Seam Filter</filter-name>
   <filter-class
>org.jboss.seam.servlet.SeamFilter</filter-class>
</filter>

<filter-mapping>
   <filter-name
>Seam Filter</filter-name>
   <url-pattern
>/*</url-pattern>
</filter-mapping
>
```

Seam ######### `web.xml` ###### ################## ################################

Seam ##################### ##################### `components.xml` #############

- `url-pattern` — ###################### ############ `url-pattern` ################## Tomcat ###############
- `regex-url-pattern` — ###################### ############ `regex-url-pattern` ####################
- `disabled` — ####################

######## URI ################# (`HttpServletRequest.getURIPath()` ###)# ##

##

30.1.4.1.

####### `pages.xml` ################ (############################)# ### ## (Java EE ######### Web ######################## ## ### Tomcat ##############################)

`components.xml#<web:exception-filter>`######

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:web="http://jboss.com/products/seam/web">
```

```
    <web:exception-filter url-pattern="*.seam"/>

</components
>
```

30.1.4.2.

#######Seam#################################### ################### Seam ### ID ##########################

################################## `components.xml`#########################

```
<web:redirect-filter url-pattern="*.seam"/>
```

30.1.4.3. URL

######### Seam # `pages.xml` ################## URL ######################## ############################## `components.xml` ############################

```
<web:rewrite-filter view-mapping="*.seam"/>
```

`view-mapping` ###### `web.xml` ######## Faces ## ################### `*.seam` ##########

30.1.4.4.

Seam ########### JSF ###################### ################### RFC-2388 (multipart/form-data ##) ########### ###################### `components.xml`##############

```
<web:multipart-filter create-temp-files="true"
                      max-request-size="1000000"
                      url-pattern="*.seam"/>
```

- `create-temp-files` — `true` ## ### ######### `false` ###
- `max-request-size` — ########################## (#### `Content-Length` ###############) ############ ######## 0 ## (########)#

30.1.4.5.

##################################

################################components.xml ############

```
<web:character-encoding-filter encoding="UTF-16"
                               override-client="true"
                               url-pattern="*.seam"/>
```

- encoding — ###############
- override-client — true ## encoding ################### false ## ######## false ###

30.1.4.6. RichFaces

RichFaces ############# Seam # RichFaces Ajax ## web.xml ##### RichFaces Ajax ##################

RichFaces Ajax ##### RichFaces jar ############################

####################### components.xml ####### ###### RichFaces Developer Guide ################

```
<web:ajax4jsf-filter force-parser="true"
                     enable-cache="true"
                     log4j-init-file="custom-log4j.xml"
                     url-pattern="*.seam"/>
```

- force-parser — JSF ###### Richfaces # XML ########################## false ## AJAX ############## XML ######## force-parser # false ################### AJAX ########################
- enable-cache — ############################ (javascript# CSS# ######)# ##### javascript # CSS ########## true ################################
- log4j-init-file — ############################ log4j.xml ##

30.1.4.7.

################## log4j ################## ##### %X{username} ##

########################## ########## `<web:logging-filter>` ###### `components.xml` ######################

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:web="http://jboss.com/products/seam/web">
   <web:logging-filter url-pattern="*.seam"/>
</components
>
```

30.1.4.8.

JSF ##########################JSF################## ####Seam# Seam#################### ###################################

############################## Seam######################### #########Seam### JSF# `FacesServlet` ###################### Seam #JSF############## phase listener#######

################################`components.xml` ############

```
<web:context-filter url-pattern="/media/*"/>
```

###########`conversationId`################ ID ###################### ID###############

######## ID####################### Seam############ `conversation`########## ID#########

30.1.4.9. Enabling HTTP cache-control headers

Seam does *not* automatically add `cache-control` HTTP headers to any resources served by the Seam resource servlet, or directly from your view directory by the servlet container. This means that your images, Javascript and CSS files, and resource representations from Seam resource servlet such as Seam Remoting Javascript interfaces are usually not cached by the browser. This is convenient in development but should be changed in production when optimizing the application.

You can configure a Seam filter to enable automatic addition of `cache-control` headers depending on the requested URI in `components.xml`:

```
<web:cache-control-filter name="commonTypesCacheControlFilter"
                          regex-url-pattern=".*(\.gif|\.png|\.jpg|\.jpeg|\.css|\.js)"
                          value="max-age=86400"/> <!-- 1 day -->

<web:cache-control-filter name="anotherCacheControlFilter"
                          url-pattern="/my/cachable/resources/*"
                          value="max-age=432000"/> <!-- 5 days -->
```

You do not have to name the filters unless you have more than one filter enabled.

30.1.4.10.

Seam ######################### ############### ## ####### (############### `web.xml` ####################)# `@Filter` ######## Seam ####### (`javax.servlet.Filter` ############) ##########

```
@Startup
@Scope(APPLICATION)
@Name("org.jboss.seam.web.multipartFilter")
@BypassInterceptors
@Filter(within="org.jboss.seam.web.ajax4jsfFilter")
public class MultipartFilter extends AbstractFilter {
```

`@Startup` ############ Seam ###################### ################### (`@BypassInterceptors`)# ##### RichFaces ################## (`@Filter(within="org.jboss.seam.web.ajax4jsfFilter")`)#

30.1.5. EJB ##### Seam

In a Seam application, EJB components have a certain duality, as they are managed by both the EJB container and Seam. Actually, it's more that Seam resolves EJB component references, manages the lifetime of stateful session bean components, and also participates in each method call via interceptors. Let's start with the configuration of the Seam interceptor chain.

We need to apply the `SeamInterceptor` to our Seam EJB components. This interceptor delegates to a set of built-in server-side interceptors that handle such concerns as bijection, conversation demarcation, and business process signals. The simplest way to do this across an entire application is to add the following interceptor configuration in `ejb-jar.xml`:

```
<interceptors>
  <interceptor>
    <interceptor-class
```

```
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
   </interceptor>
</interceptors>

<assembly-descriptor>
   <interceptor-binding>
      <ejb-name
>*</ejb-name>
      <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
   </interceptor-binding>
</assembly-descriptor>
```

Seam needs to know where to go to find session beans in JNDI. One way to do this is specify the `@JndiName` annotation on every session bean Seam component. However, this is quite tedious. A better approach is to specify a pattern that Seam can use to calculate the JNDI name from the EJB name. Unfortunately, there is no standard mapping to global JNDI defined in the EJB3 specification, so this mapping is vendor-specific (and may depend on your own naming conventions as well). We usually specify this option in `components.xml`.

JBoss##############################

```
<core:init jndi-name="earName/#{ejbName}/local" />
```

In this case, `earName` is the name of the EAR in which the bean is deployed, Seam replaces `#{ejbName}` with the name of the EJB, and the final segment represents the type of interface (local or remote).

Outside the context of an EAR (when using the JBoss Embeddable EJB3 container), the first segment is dropped since there is no EAR, leaving us with the following pattern:

```
<core:init jndi-name="#{ejbName}/local" />
```

How these JNDI names are resolved and somehow locate an EJB component might appear a bit like black magic at this point, so let's dig into the details. First, let's talk about how the EJB components get into JNDI.

The folks at JBoss don't care much for XML, if you can't tell. So when they designed JBoss AS, they decided that EJB components would get assigned a global JNDI name automatically, using the pattern just described (i.e., EAR name/EJB name/interface type). The EJB name is the first non-empty value from the following list:

- The value of the `<ejb-name>` element in ejb-jar.xml

- The value of the `name` attribute in the @Stateless or @Stateful annotation
- The simple name of the bean class

Let's look at an example. Assume that you have the following EJB bean and interface defined.

```
package com.example.myapp;

import javax.ejb.Local;

@Local
public interface Authenticator
{
   boolean authenticate();
}

package com.example.myapp;

import javax.ejb.Stateless;

@Stateless
@Name("authenticator")
public class AuthenticatorBean implements Authenticator
{
   public boolean authenticate() { ... }
}
```

Assuming your EJB bean class is deployed in an EAR named myapp, the global JNDI name myapp/AuthenticatorBean/local will be assigned to it on JBoss AS. As you learned, you can reference this EJB component as a Seam component with the name `authenticator` and Seam will take care of finding it in JNDI according to the JNDI pattern (or `@JndiName` annotation).

So what about the rest of the application servers? Well, according to the Java EE spec, which most vendors try to adhere to religiously, you have to declare an EJB reference for your EJB in order for it to be assigned a JNDI name. That requires some XML. It also means that it is up to you to establish a JNDI naming convention so that you can leverage the Seam JNDI pattern. You might find the JBoss convention a good one to follow.

There are two places you have to define the EJB reference when using Seam on non-JBoss application servers. If you are going to be looking up the Seam EJB component through JSF (in a JSF view or as a JSF action listener) or a Seam JavaBean component, then you must declare the EJB reference in web.xml. Here is the EJB reference for the example component just shown:

```
<ejb-local-ref>
```

```
    <ejb-ref-name>myapp/AuthenticatorBean/local</ejb-ref-name>
    <ejb-ref-type>Session</ejb-ref-type>
    <local>org.example.vehicles.action.Authenticator</local>
</ejb-local-ref>
```

This reference will cover most uses of the component in a Seam application. However, if you want to be able to inject a Seam EJB component into another Seam EJB component using `@In`, you need to define this EJB reference in another location. This time, it must be defined in ejb-jar.xml, and it's a bit tricker.

Within the context of an EJB method call, you have to deal with a somewhat sheltered JNDI context. When Seam attempts to find another Seam EJB component to satisfy an injection point defined using `@In`, whether or not it finds it depends on whether an EJB reference exists in JNDI. Strictly speaking, you cannot simply resolve JNDI names as you please. You have to define the references explicitly. Fortunately, JBoss recognized how aggrevating this would be for the developer and all versions of JBoss automatically register EJBs so they are always available in JNDI, both to the web container and the EJB container. So if you are using JBoss, you can skip the next few paragraphs. However, if you are deploying to GlassFish, pay close attention.

For application servers that stubbornly adhere to the EJB specification, EJB references must always be defined explicitly. But unlike with the web context, where a single resource reference covers all uses of the EJB from the web environment, you cannot declare EJB references globally in the EJB container. Instead, you have to specify the JNDI resources for a given EJB component one-by-one.

Let's assume that we have an EJB named RegisterAction (the name is resolved using the three steps mentioned previously). That EJB has the following Seam injection:

```
@In(create = true)
Authenticator authenticator;
```

In order for this injection to work, the link must be established in the ejb-jar.xml file as follows:

```
<ejb-jar>
  <enterprise-beans>
    <session>
      <ejb-name>RegisterAction</ejb-name>
      <ejb-local-ref>
        <ejb-ref-name>myapp/AuthenticatorAction/local</ejb-ref-name>
        <ejb-ref-type>Session</ejb-ref-type>
        <local>com.example.myapp.Authenticator</local>
      </ejb-local-ref>
    </session>
```

```
    </enterprise-beans>

    ...

</ejb-jar>
```

Notice that the contents of the `<ejb-local-ref>` are identical to what we defined in web.xml. What we are doing is bringing the reference into the EJB context where it can be used by the RegisterAction bean. You will need to add one of these references for any injection of a Seam EJB compoenent into another Seam EJB component using `@In`. (You can see an example of this setup in the jee5/booking example).

But what about `@EJB`? It's true that you can inject one EJB into another using `@EJB`. However, by doing so, you are injecting the actual EJB reference rather than the Seam EJB component instance. In this case, some Seam features will work, while others won't. That's because Seam's interceptor is invoked on any method call to an EJB component. But that only invokes Seam's server-side interceptor chain. What you lose is Seam's state management and Seam's client-side interceptor chain. Client-side interceptors handle concerns such as security and concurrency. Also, when injecting a SFSB, there is no guarantee that you will get the SFSB bound to the active session or conversation, whatever the case may be. Thus, you definitely want to inject the Seam EJB component using `@In`.

That covers how JNDI names are defined and used. The lesson is that with some application servers, such as GlassFish, you are going to have to specify JNDI names for all EJB components explicitly, and sometimes twice! And even if you are following the same naming convention as JBoss AS, the JNDI pattern in Seam may need to be altered. For instance, the global JNDI names are automatically prefixed with java:comp/env on GlassFish, so you need to define the JNDI pattern as follows:

```
<core:init jndi-name="java:comp/env/earName/#{ejbName}/local" />
```

Finally, let's talk about transactions. In an EJB3 environment, we recommend the use of a special built-in component for transaction management, that is fully aware of container transactions, and can correctly process transaction success events registered with the `Events` component. If you don't add this line to your `components.xml` file, Seam won't know when container-managed transactions end:

```
<transaction:ejb-transaction/>
```

30.1.6.

#################### Seam############################ `seam.properties#META-INF/seam.properties` #### `META-INF/components.xml`############## (################)# Seam##### Seam#########################`seam.properties`#########

Seam##############web##### (WAR) `#WEB-INF/classes`###### `seam.properties`################

Seam ######## `seam.properties` ################

You might think this is silly and what kind of idiot framework designers would make an empty file affect the behavior of their software?? Well, this is a workaround for a limitation of the JVM — if we didn't use this mechanism, our next best option would be to force you to list every component explicitly in `components.xml`, just like some other competing frameworks do! I think you'll like our way better.

30.2. ### JPA

Seam ######## JPA ######## Hibernate ################## ## JPA ################## `seam` ##############

###########

JPA ###################### ######################################

Seam #### JPA ########## 2 ####################

######### `components.xml` ####### ####### `PersistenceProvider` # hibernate ############### ##################

```
<component name="org.jboss.seam.persistence.persistenceProvider"
        class="org.jboss.seam.persistence.PersistenceProvider"
        scope="stateless">
</component
>
```

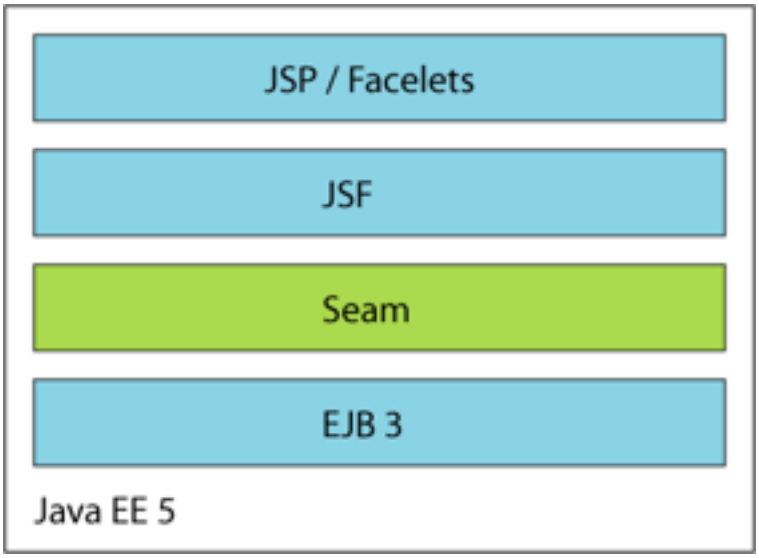
JPA ###################### `PersistenceProvider` ################ `HibernatePersistenceProvider` ########## (######################)# ######### `seam` ###################

```
<component name="org.jboss.seam.persistence.persistenceProvider"
```

```
        class="org.your.package.YourPersistenceProvider">
</component
>
```

###################################### `persistence.xml` ########## ######### jar ##

30.3. Java EE 5 # Seam



Java EE 5 ############ Seam ######################

30.3.1.

EAR ####################################

```
my-application.ear/
    jboss-seam.jar
    lib/
        jboss-el.jar
    META-INF/
        MANIFEST.MF
        application.xml
    my-application.war/
        META-INF/
            MANIFEST.MF
        WEB-INF/
            web.xml
            components.xml
            faces-config.xml
            lib/
```

```
            jsf-facelets.jar
            jboss-seam-ui.jar
        login.jsp
        register.jsp
        ...
    my-application.jar/
        META-INF/
            MANIFEST.MF
            persistence.xml
        seam.properties
        org/
            jboss/
                myapplication/
                    User.class
                    Login.class
                    LoginBean.class
                    Register.class
                    RegisterBean.class
                    ...
```

`jboss-seam.jar` # ejb ######## `META-INF/application.xml` ############### `jboss-el.jar` # EAR # lib ################ (EAR ###########)#

jBPM ### Drools ######### EAR # lib ########## jar ###############

facelets ####### (##) # WAR#`WEB-INF/lib`####### `jsf-facelets.jar`############

Seam ################ (##### Seam ###############)# WAR ##### `WEB-INF/lib` ####### `jboss-seam-ui.jar` ############ PDF# email ################## `WEB-INF/lib` # `jboss-seam-pdf.jar` ### `jboss-seam-mail.jar` ############

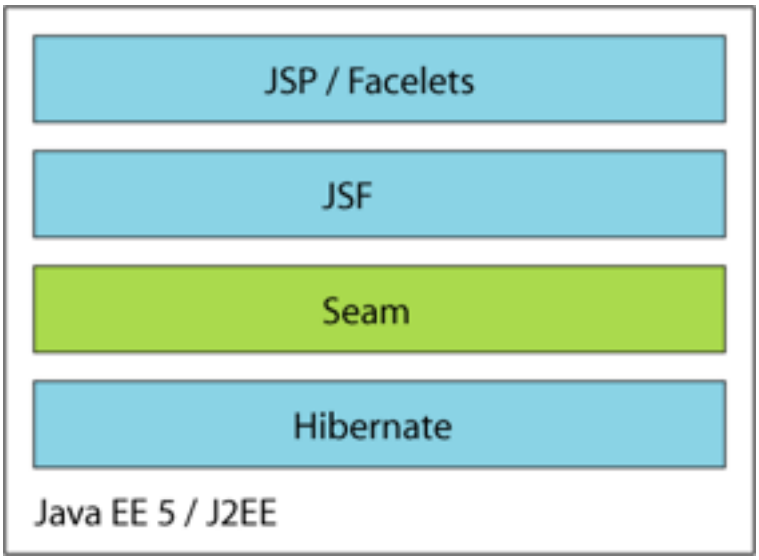
Seam ############ (facelets ################) #### WAR# `WEB-INF/lib`#######`jboss-seam-debug.jar`######

############## EJB 3.0######## Java EE################## Seam ##################

################################## ####3 ############## ##

30.4. J2EE## Seam

Seam # EJB 3.0 ####################### ######### EJB 3.0 ####### Hibernate3 # JPA ##### ##### Bean ######### JavaBean ############# ##### Bean ###################### ######### EJB 3.0 ################# ####### Seam ################################



Seam JavaBean ############ Bean ########################## #### JTA `UserTransaction` ####### #### Seam # `@Transactional` ################## ##### ### JavaBean # Hibernate ###################### Seam ##################

Seam ##############EJB3 ##### Hibernate # JavaBean ##### ######################## ###################J2EE############## ############

30.4.1. Seam ## Hibernate#########

Seam ########################## `hibernate.cfg.xml` ###### Hibernate # `SessionFactory` ############

```
<persistence:hibernate-session-factory name="hibernateSessionFactory"/>
```

#################Seam##### Hibernate#`Session`########## *managed session*############

```
<persistence:managed-hibernate-session name="hibernateSession"
                    session-factory="#{hibernateSessionFactory}"/>
```

30.4.2. Seam ## JPA#########

Seam ############################`persistence.xml` ##`EntityManagerFactory` JPA############

```
<persistence:entity-manager-factory name="entityManagerFactory"/>
```

######### Seam ###JPA `EntityManager` ############ ########## ##############

```
<persistence:managed-persistence-context name="entityManager"
                    entity-manager-factory="#{entityManagerFactory}"/>
```

30.4.3.

######### WAR#################################

```
my-application.war/
   META-INF/
      MANIFEST.MF
   WEB-INF/
      web.xml
      components.xml
      faces-config.xml
      lib/
         jboss-seam.jar
         jboss-seam-ui.jar
         jboss-el.jar
         jsf-facelets.jar
         hibernate3.jar
         hibernate-annotations.jar
         hibernate-validator.jar
         ...
         my-application.jar/
            META-INF/
               MANIFEST.MF
            seam.properties
            hibernate.cfg.xml
            org/
               jboss/
                  myapplication/
                     User.class
                     Login.class
                     Register.class
                     ...
   login.jsp
   register.jsp
   ...
```

Hibernate # Tomcat # TestNG #### EE ################################

30.5. JBoss Embedded ### Java SE # Seam

Seam #### EE ################## ##### ##### JTA ##### Seam ################################## JPA ########## Seam # JPA ################## `EntityTransaction` ####################

```
<transaction:entity-transaction entity-manager="#{entityManager}"/>
```

Hibernate ########## Seam ###### Hibernate ######## API ###################

```
<transaction:hibernate-transaction session="#{session}"/>
```

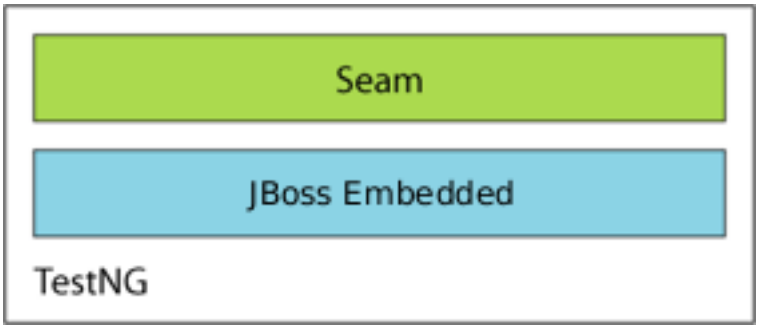
###################

########## JBoss Embedded ##### EE # API #########

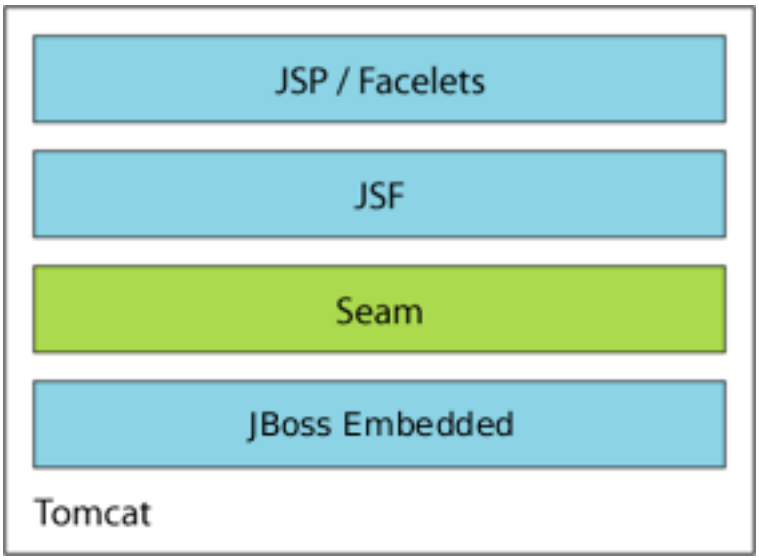
30.6. JBoss Embedded #### Java SE # Seam

JBoss Embedded ### Java EE 5 ###################### EJB 3 ############################ ############ ############

Seam ################ TestNG ############## `SeamTest` #### JBoss Embedded #######



################ Tomcat ###########



30.6.1. Embedded JBoss

Embedded JBoss must by installed into Tomcat for Seam applications to run correctly on it. Embedded JBoss runs with JDK 5 or JDK 6 (see *#42.1. #JDK #####* for details on using JDK 6). Embedded JBoss can be downloaded *here* [http://sourceforge.net/projects/jboss/files/Embedded%20JBoss/Embedded%20JBoss%20Beta%203/]. The process for installing Embedded JBoss into Tomcat 6 is quite simple. First, you should copy the Embedded JBoss JARs and configuration files into Tomcat.

- `jndi.properties` ######## Embedded JBoss # `bootstrap` ####### `lib` ########################## Tomcat # `lib` ###############
- Tomcat # `lib` ######## `annotations-api.jar` ###########

Embedded JBoss ############ 2

- Embedded JBoss #### `EmbeddedJBossBootstrapListener` # `conf/server.xml` ####### ###################################

```
<Server port="8005" shutdown="SHUTDOWN">

  <!-- Comment these entries out to disable JMX MBeans support used for the
       administration web application -->
  <Listener className="org.apache.catalina.core.AprLifecycleListener" />
  <Listener className="org.apache.catalina.mbeans.ServerLifecycleListener" />
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />
  <Listener className="org.apache.catalina.storeconfig.StoreConfigLifecycleListener" />
```

```
<!-- Add this listener -->
<Listener className="org.jboss.embedded.tomcat.EmbeddedJBossBootstrapListener" />
```

- WAR ########## `WebinfScanner` ##### `conf/context.xml` ##################

```
<Context>
  <!-- Default set of monitored resources -->
  <WatchedResource>WEB-INF/web.xml</WatchedResource>

  <!-- Uncomment this to disable session persistence across Tomcat restarts -->
  <!--
  <Manager pathname="" />
  -->
```

```
<!-- Add this listener -->
<Listener className="org.jboss.embedded.tomcat.WebinfScanner" />
```

```
</Context
>
```

- If you are using Sun JDK 6, you need to set the Java option `sun.lang.ClassLoader.allowArraySyntax` to `true` in the JAVA_OPTS environment variable used by the Catalina startup script (catalina.bat on Windows or catalina.sh on Unix).

  Open the script appropriate for your operating system in a text editor. Add a new line immediately below the comments at the top of the file where you will define the JAVA_OPTS environment variable. On Windows, use the following syntax:

```
set JAVA_OPTS=%JAVA_OPTS% -Dsun.lang.ClassLoader.allowArraySyntax=true
```

  On Unix, use this syntax instead:

```
JAVA_OPTS="$JAVA_OPTS -Dsun.lang.ClassLoader.allowArraySyntax=true"
```

############### Embedded JBoss Tomcat ## *wiki* #### [http://wiki.jboss.org/wiki/Wiki.jsp?page=EmbeddedAndTomcat] ##########

30.6.2.

Tomcat #### ############ WAR ##################################

```
my-application.war/
   META-INF/
      MANIFEST.MF
   WEB-INF/
      web.xml
      components.xml
      faces-config.xml
      lib/
         jboss-seam.jar
         jboss-seam-ui.jar
         jboss-el.jar
         jsf-facelets.jar
         jsf-api.jar
         jsf-impl.jar
         ...
         my-application.jar/
            META-INF/
               MANIFEST.MF
               persistence.xml
            seam.properties
            org/
               jboss/
                  myapplication/
                     User.class
                     Login.class
                     LoginBean.class
                     Register.class
                     RegisterBean.class
                     ...
   login.jsp
   register.jsp
   ...
```

Seam ##############`ant deploy.tomcat` ############ Tomcat

30.7. Seam##jBPM##

Seam # jBPM ## jBPM ############## ### ################## `components.xml` #################

```
<bpm:jbpm>
   <bpm:pageflow-definitions>
      <value
>createDocument.jpdl.xml</value>
      <value
>editDocument.jpdl.xml</value>
      <value
>approveDocument.jpdl.xml</value>
   </bpm:pageflow-definitions>
   <bpm:process-definitions>
      <value
>documentLifecycle.jpdl.xml</value>
   </bpm:process-definitions>
</bpm:jbpm
>
```

jBPM ##### jBPM ## Hibernate ################Seam DVD Store demo# Seam ##### `jbpm.cfg.xml#hibernate.cfg.xml`####

```
<jbpm-configuration>

 <jbpm-context>
   <service name="persistence">
     <factory>
       <bean class="org.jbpm.persistence.db.DbPersistenceServiceFactory">
         <field name="isTransactionEnabled"
><false/></field>
       </bean>
     </factory>
   </service>
   <service name="tx" factory="org.jbpm.tx.TxServiceFactory" />
   <service name="message" factory="org.jbpm.msg.db.DbMessageServiceFactory" />
   <service name="scheduler" factory="org.jbpm.scheduler.db.DbSchedulerServiceFactory" />
   <service name="logging" factory="org.jbpm.logging.db.DbLoggingServiceFactory" />
   <service name="authentication"
         factory="org.jbpm.security.authentication.DefaultAuthenticationServiceFactory" />
 </jbpm-context>

</jbpm-configuration
>
```

###############jBPM############################
Seam#####EJB3#JTA##################

30.7.1.

jBPM ## Seam ###################### EAR ###################### ## EAR ############

```
my-application.ear/
    jboss-seam.jar
    lib/
      jboss-el.jar
      jbpm-3.1.jar
    META-INF/
       MANIFEST.MF
       application.xml
    my-application.war/
       META-INF/
          MANIFEST.MF
       WEB-INF/
          web.xml
          components.xml
          faces-config.xml
          lib/
             jsf-facelets.jar
             jboss-seam-ui.jar
       login.jsp
       register.jsp
       ...
    my-application.jar/
       META-INF/
          MANIFEST.MF
          persistence.xml
       seam.properties
       org/
          jboss/
             myapplication/
                User.class
                Login.class
                LoginBean.class
                Register.class
                RegisterBean.class
                ...
```

```
jbpm.cfg.xml
hibernate.cfg.xml
createDocument.jpdl.xml
editDocument.jpdl.xml
approveDocument.jpdl.xml
documentLifecycle.jpdl.xml
```

30.8. JBoss AS## SFSB###############

########### Bean ######## HTTP ################################## ####### SFSB ###### HTTP ############################## JBoss Application Server ###### Bean ############## 30 ##### server/default/conf/standardjboss.xml (default ####################) ##########

SFSB ####### `LRUStatefulContextCachePolicy` ######### `max-bean-life`

```
<container-cache-conf>
   <cache-policy
>org.jboss.ejb.plugins.LRUStatefulContextCachePolicy</cache-policy>
   <cache-policy-conf>
      <min-capacity
>50</min-capacity>
      <max-capacity
>1000000</max-capacity>
      <remover-period
>1800</remover-period>

      <!-- SFSB timeout in seconds; 1800 seconds == 30 minutes -->
      <max-bean-life
>1800</max-bean-life
>

      <overager-period
>300</overager-period>
      <max-bean-age
>600</max-bean-age>
      <resizer-period
>400</resizer-period>
      <max-cache-miss-period
>60</max-cache-miss-period>
      <min-cache-miss-period
>1</min-cache-miss-period>
```

```
      <cache-load-factor
>0.75</cache-load-factor>
   </cache-policy-conf>
</container-cache-conf
>
```

The default HTTP session timeout can be modified in `server/default/deploy/jbossweb-tomcat55.sar/conf/web.xml` for JBoss 4.0.x, or in `server/default/deploy/jboss-web.deployer/conf/web.xml` for JBoss 4.2.x or later. The following entry in this file controls the default session timeout for all web applications:

```
<session-config>
   <!-- HTTP Session timeout, in minutes -->
   <session-timeout
>30</session-timeout>
</session-config
>
```

`web.xml`

30.9. Portlet # Seam

Seam ######### porlet ########## Seam ### RichFaces ####### portlet ## JSF ####### JSR-301 ### JBoss Portlet Bridge ########## ### *http://labs.jboss.com/portletbridge* ##########

30.10.

Seam ########## `/seam.properties#` `/META-INF/components.xml#` ### `/META-INF/seam.properties` ####### jar ######### ##### `@Name` ################### Seam ########## Seam ########

Seam ########################## ############################## Seam ###################### ## Seam # `/META-INF/seam-deployment.properties`

```
# A colon-separated list of annotation types to handle
org.jboss.seam.deployment.annotationTypes=com.acme.Foo:com.acme.Bar
```

################ `@Foo` ##############################

```
@Name("fooStartup")
@Scope(APPLICATION)
@Startup
public class FooStartup {

  @In("#{deploymentStrategy.annotatedClasses['com.acme.Foo']}")
  private Set<Class<Object>> fooClasses;

  @In("#{hotDeploymentStrategy.annotatedClasses['com.acme.Foo']}")
  private Set<Class<Object>> hotFooClasses;

  @Create
  public void create() {
    for (Class clazz: fooClasses) {
      handleClass(clazz);
    }
    for (Class clazz: hotFooClasses) {
      handleClass(clazz);
    }
  }

  public void handleClass(Class clazz) {
     // ...
  }

}
```

.foo.xml ######################

```
public class FooDeploymentHandler implements DeploymentHandler {
   private static DeploymentMetadata FOO_METADATA = new DeploymentMetadata()
   {

      public String getFileNameSuffix() {
         return ".foo.xml";
      }
   };

  public String getName() {
    return "fooDeploymentHandler";
  }
```

```
    public DeploymentMetadata getMetadata() {
        return FOO_METADATA;
    }
}
```

########## `.foo.xml` ##########################

Then, we need to register the deployment handler with Seam in `/META-INF/seam-deployment.properties`. You can register multiple deployment handler using a comma separated list.

```
# For standard deployment
org.jboss.seam.deployment.deploymentHandlers=com.acme.FooDeploymentHandler
# For hot deployment
org.jboss.seam.deployment.hotDeploymentHandlers=com.acme.FooDeploymentHandler
```

Seam uses deployment handlers internally to install components and namespaces. You can easily access the deployment handler during an `APPLICATION` scoped component's startup:

```
@Name("fooStartup")
@Scope(APPLICATION)
@Startup
public class FooStartup {

   @In("#{deploymentStrategy.deploymentHandlers['fooDeploymentHandler']}")
   private FooDeploymentHandler myDeploymentHandler;

   @In("#{hotDeploymentStrategy.deploymentHandlers['fooDeploymentHandler']}")
   private FooDeploymentHandler myHotDeploymentHandler;

   @Create
   public void create() {
      for (FileDescriptor fd: myDeploymentHandler.getResources()) {
          handleFooXml(fd);
      }

      for (FileDescriptor f: myHotDeploymentHandler.getResources()) {
          handleFooXml(fd);
      }
   }
```

```
    public void handleFooXml(FileDescriptor fd) {
        // ...
    }
}
```

Seam

Seam ## Seam ## EJB 3.0 ############################ Hibernate Validator ################### Seam # Seam ####################################

#################### `org.jboss.seam.annotations` ########

31.1.

################## Seam ##

`@Name`

```
@Name("componentName")
```

####### Seam ######################### # Seam ##################

`@Scope`

```
@Scope(ScopeType.CONVERSATION)
```

########################### ##### `ScopeType` ######### `EVENT`, `PAGE`, `CONVERSATION`, `SESSION`, `BUSINESS_PROCESS`, `APPLICATION`, `STATELESS` ########

#################### ############################ ########### Bean #### ###### `STATELESS` ###### ###### bean ############### Bean ### ###### `CONVERSATION` #### JavaBeans ######## `EVENT` ######

`@Role`

```
@Role(name="roleName", scope=ScopeType.SESSION)
```

Seam ################################## `@Name`/`@Scope` ########################## # `@Role` ####################

- `name` — ############
- `scope` — ################ ################## ##############################

`@Roles`

```
@Roles({
      @Role(name="user", scope=ScopeType.CONVERSATION),
      @Role(name="currentUser", scope=ScopeType.SESSION)
  })
```

##########################

`@BypassInterceptors`

```
@BypassInterceptors
```

##

`@JndiName`

```
@JndiName("my/jndi/name")
```

Seam # EJB ################### JNDI ######## JNDI ############### Seam # `org.jboss.seam.core.init.jndiPattern` ###### JNDI ###########

`@Conversational`

```
@Conversational
```

###

`@PerNestedConversation`

```
@PerNestedConversation
```

###

###

`@Startup`

```
@Scope(APPLICATION) @Startup(depends="org.jboss.seam.bpm.jbpm")
```

JNDI#

```
@Scope(SESSION) @Startup
```

##

- `depends` — ##

`@Install`

```
@Install(false)
```

`@Install`

```
@Install(dependencies="org.jboss.seam.bpm.jbpm")
```

###

```
@Install(genericDependencies=ManagedQueueSender.class)
```

###
############################

```
@Install(classDependencies="org.hibernate.Session")
```

###

```
@Install(precedence=BUILT_IN)
```

################## ################### ############################## ################ (##) :

- `BUILT_IN` — ########Seam###########
- `FRAMEWORK` — Seam############################
- `APPLICATION` — Precedence of application components (the default precedence)

- `DEPLOYMENT` — ##
- `MOCK` — ########################

`@Synchronized`

```
@Synchronized(timeout=1000)
```

################################## Seam ###################### ##

`@ReadOnly`

```
@ReadOnly
```

JavaBean ##

`@AutoCreate`

```
@AutoCreate
```

################################## `create=true` ##################

31.2.

############################## ##

`@In`

```
@In
```

null #########

```
@In(required=false)
```

null

```
@In(create=true)
```

null ################### Seam

```
@In(value="contextVariableName")
```

############################## ####################

```
@In(value="#{customer.addresses['shipping']}")
```

############################ JSF EL ############################

- `value` — ### `#{...}` ##### JSF EL ########
- `create` — ######################## (null) #### Seam ## false ###
- `required` — ############################ Seam ################

`@Out`

```
@Out
```

Seam ## ### null #### #########

```
@Out(required=false)
```

Seam ## ### null ########

```
@Out(scope=ScopeType.SESSION)
```

Seam ##

#################### #### `@Out` ########################### (#################### `EVENT`) #

```
@Out(value="contextVariableName")
```

############################ ####################

- `value` — ######################################
- `required` — ##################### null ###### Seam #################

###################### #:

```
@In(create=true) @Out private User currentUser;
```

The next annotation supports the *manager component* pattern; a Seam component manages the lifecycle of an instance of some other class that is to be injected. It appears on a component getter method.

`@Unwrap`

```
@Unwrap
```

################ getter ##

The next annotation supports the *factory component* pattern; a Seam component is responsible for initializing the value of a context variable. This is especially useful for initializing any state needed for rendering the response to a non-faces request. It appears on a component method.

`@Factory`

```
@Factory("processInstance") public void createProcessInstance() { ... }
```

################ ## ####### `void` ##############

```
@Factory("processInstance", scope=CONVERSATION) public ProcessInstance createProcessInstance() { ... }
```

################ Seam ## ###################### ################### `@Factory`

############################## (########################
##################EVENT ############) #

- `value` — ######################## getter ######## JavaBeans ####################
- `scope` — Seam###
- `autoCreate` — ##@In # `create=true` ##################

`Log` ################:

`@Logger`

```
@Logger("categoryName")
```

############ `org.jboss.seam.log.Log` ################## ###### Bean #### ######## static ################

- `value` — ######################################

###############################:

`@RequestParameter`

```
@RequestParameter("parameterName")
```

############################## ##################

- `value` — ######################################

31.3.

############### ##
#####################################

`@Create`

```
@Create
```

################ Seam ################################## create ##### JavaBeans ############## Bean ##########################

`@Destroy`

```
@Destroy
```

destroy ##### JavaBeans ############## Bean

Destroy ############################## *Seam # destroy* ######################################

`@Observer`

```
@Observer("somethingChanged")
```

##

```
@Observer(value="somethingChanged",create=false)
```

###
create # false ##################create ######## true ###

31.4.

############################ #### Seam ######################################

Web ############################ ######################## ####################@Begin ################## #### *(long-running conversation)*

`@Begin`

```
@Begin
```

############ null ##### (outcome) ########################

```
@Begin(join=true)
```

################## ########################

```
@Begin(nested=true)
```

################## ################################ ## @End
##

```
@Begin(pageflow="process definition name")
```

################## jBPM ################

```
@Begin(flushMode=FlushModeType.MANUAL)
```

Seam ########################## `flushMode=FlushModeType.MANUAL` # ########
(atomic conversation) ######### ##### `flush()` (### ############)
################ ################################

- `join` — ##########################`true` ################## `false`
  ###################### `false` ###`nested=true` ########################
- `nested` — ##
- `flushMode` — ########## Seam ### Hibernate ######### JPA
  ##########################
- `pageflow` — `org.jboss.seam.bpm.jbpm.pageflowDefinitions` ########### jBPM
  ################

`@End`

```
@End
```

############ null ##### (outcome) ###### ################

- `beforeRedirect` —
  ####################################`beforeRedirect=true`
  ##
- `root` — ##`root=true`
  #########
  ##

`@StartTask`

```
@StartTask
```

jBPM ############ ############ null ##### (outcome) ###### ################## ###################### jBPM ############ ############## ############################## ######################

- The jBPM `TaskInstance` will be available in a request context variable named `taskInstance`. The jBPM `ProcessInstance` will be available in a request context variable named `processInstance`. (Of course, these objects are available for injection via `@In`.)
- `taskIdParameter` — ####ID###################### `"taskId"` ###### Seam `taskList` JSF ##########################
- `flushMode` — ########## Seam ### Hibernate ######### JPA ##########################

`@BeginTask`

```
@BeginTask
```

####### jBPM ############# ############ null ##### (outcome) ###### ################## ###################### jBPM ############ ############## ############################## ######################

- The jBPM `org.jbpm.taskmgmt.exe.TaskInstance` will be available in a request context variable named `taskInstance`. The jBPM `org.jbpm.graph.exe.ProcessInstance` will be available in a request context variable named `processInstance`.
- `taskIdParameter` — ####ID###################### `"taskId"` ###### Seam `taskList` JSF ##########################
- `flushMode` — ########## Seam ### Hibernate ######### JPA ##########################

`@EndTask`

```
@EndTask
```

jBPM ############ ####### null ##### (outcome) ###### ######### ################### jBPM ########### ######### `transition` ################ `Transition.setName ()` ########## ############################

```
@EndTask(transition="transitionName")
```

jBPM

- `transition` — ################## jBPM ######## ####################
- `beforeRedirect` — ######################################`beforeRedirect=true` ##

`@CreateProcess`

```
@CreateProcess(definition="process definition name")
```

########## null ##### (outcome) ###### ### jBPM ################ `ProcessInstance` ####### `processInstance` ####################

- `definition` — `org.jboss.seam.bpm.jbpm.processDefinitions` ########## jBPM ############

`@ResumeProcess`

```
@ResumeProcess(processIdParameter="processId")
```

########## null ##### (outcome) ###### ### jBPM ###################### `ProcessInstance` ####### `processInstance` ##################

- `processIdParameter` — #### ID ################### `"processId"` ###

`@Transition`

```
@Transition("cancel")
```

null ################## jBPM

31.5. J2EE ### Seam JavaBean

Seam ################ (outcome) #### JTA ################################

`@Transactional`

```
@Transactional
```

JavaBean ############# Bean ##
################################
############################ ###################################

EJB 3.0 ######################### `@TransactionAttribute` #########

`@ApplicationException`

```
@ApplicationException
```

javax.ejb.ApplicationException #####Java EE 5
###(#########)#######################

EJB 3.0 ##########################`@javax.ejb.ApplicationException`
##############

- `rollback` — ####### `false` ###`true` ################### rollback only ########
- `end` — ####### `false` ###`true` ########################

`@Interceptors`

```
@Interceptors({DVDInterceptor, CDInterceptor})
```

javax.interceptors.Interceptors #####Java EE 5
###

EJB 3.0 ##########################`@javax.interceptor.Interceptors`
##############

############## JavaBean Seam ################EJB 3.0 ################ Java EE5
################

31.6.

############## Seam ###################################

`@Redirect`

```
@Redirect(viewId="error.jsp")
```

########################## ID ################

- `viewId` — ######## JSF ### ID ###### EL ########
- `message` — ############ ###################
- `end` — ################# ###### `false` ###

`@HttpError`

```
@HttpError(errorCode=404)
```

################### HTTP ###########

- `errorCode` — HTTP ######### ###### `500` ###
- `message` — HTTP ################# ###################
- `end` — ################# ###### `false` ###

31.7. Seam Remoting#########

Seam Remoting################ ##### Bean ################

`@WebRemote`

```
@WebRemote(exclude="path.to.exclude")
```

Indicates that the annotated method may be called from client-side JavaScript. The `exclude` property is optional and allows objects to be excluded from the result's object graph (see the *# 25. #######* chapter for more details).

31.8. Seam

############Seam ################

EJB ########################## EJB 3.0 ###################

`@Interceptor`

```
@Interceptor(stateless=true)
```

############################## Seam ####################

```
@Interceptor(type=CLIENT)
```

######### EJB ##

```
@Interceptor(around={SomeInterceptor.class, OtherInterceptor.class})
```

##

```
@Interceptor(within={SomeInterceptor.class, OtherInterceptor.class})
```

##

31.9.

############################## #:

```
@Asynchronous public void scheduleAlert(Alert alert, @Expiration Date date) { ... }
```

```
@Asynchronous public Timer scheduleAlerts(Alert alert,
  @Expiration Date date,
  @IntervalDuration long interval) { ... }
```

`@Asynchronous`

```
@Asynchronous
```

############################

`@Duration`

```
@Duration
```

################ ############################## (############################) #

`@Expiration`

```
@Expiration
```

################ ############ (########################) ##################

`@IntervalDuration`

```
@IntervalDuration
```

######################

31.10. JSF

########## JSF #################

`@Converter`

Seam ######## JSF ###################################### Seam ################### `javax.faces.convert.Converter` ##############

- `id` — JSF #####ID###################
- `forClass` — ##

`@Validator`

Seam ######## JSF ###################################### Seam ################### `javax.faces.validator.Validator` ##############

- `id` — JSF #####ID###################

31.10.1. `dataTable`

####################### Bean ######################### ######################

`@DataModel`

```
@DataModel("variableName")
```

List, Map, Set ### Object[] ######## JSF DataModel ##################################(############### STATELESS #### EVENT ####)#Map ####DataModel #### Map.Entry ###

- value — ##############################
- scope — scope=ScopeType.PAGE ###############DataModel # PAGE ##############

@DataModelSelection

```
@DataModelSelection
```

JSF DataModel ###################(### DataModel # Collection ####### Map ####)############### @DataModel ################ DataModel ############################## @DataModel ############ @DataModelSelection # value ################

####### @DataModel # PAGE ################ DataModel Selection ######################## DataModel ##################@DataModel ##################### getter ################ setter ##### Seam ############API###########

- value — ############################ @DataModel ############

@DataModelSelectionIndex

```
@DataModelSelectionIndex
```

JSF DataModel ################################(### DataModel # Collection ######## Map #####)############### @DataModel ################ DataModel ############################## @DataModel ############ @DataModelSelectionIndex # value ################

- value — ############################ @DataModel ############

31.11.

############################## @DataModel # @DataModelSelection ##################

@DataBinderClass

```
@DataBinderClass(DataModelBinder.class)
```

##

`@DataSelectorClass`

```
@DataSelectorClass(DataModelSelector.class)
```

##################################

31.12.

########## ## ## Java ################

`@Namespace`

```
@Namespace(value="http://jboss.com/products/seam/example/seampay")
```

`components.xml` ###### XML

```
@Namespace(value="http://jboss.com/products/seam/core", prefix="org.jboss.seam.core")
```

############################ ### XML ## ##### ############## `init` ### XML ####### `org.jboss.seam.core.init` ########################

31.13.

############ Seam ################################

`@Filter`

`@Filter` ############## Seam ####### (`javax.servlet.Filter` #######) ####################Seam ################

```
@Filter(around={"seamComponent", "otherSeamComponent"})
```

##

```
@Filter(within={"seamComponent", "otherSeamComponent"})
```

##

Seam

Seam ############ ###################### ############ components.xml ################### ######################### 1 ###################### components.xml

@Name ################################ ###################################

32.1.

################## ## ##### ############################ Seam ##############################

```
@In private Context sessionContext;
```

org.jboss.seam.core.contexts
: Seam ########################## ##### org.jboss.seam.core.contexts.sessionContext['user'] ###

org.jboss.seam.faces.facesContext
: FacesContext ############ (#### Seam ##########) ##########

############################

32.2. JSF-related components

The following set of components are provided to supplement JSF.

org.jboss.seam.faces.dateConverter
: Provides a default JSF converter for properties of type java.util.Date.

  This converter is automatically registered with JSF. It is provided to save a developer from having to specify a DateTimeConverter on an input field or page parameter. By default, it assumes the type to be a date (as opposed to a time or date plus time) and uses the short input style adjusted to the Locale of the user. For Locale.US, the input pattern is mm/DD/yy. However, to comply with Y2K, the year is changed from two digits to four (e.g., mm/DD/yyyy).

  It's possible to override the input pattern globally using component configuration. Consult the JavaDoc for this class to see examples.

org.jboss.seam.faces.facesMessages
: ################## faces ###################

  - add(FacesMessage facesMessage) — faces ############ ###################################

- `add(String messageTemplate)` — faces ########### EL ####################################
- `add(Severity severity, String messageTemplate)` — faces ########### EL ####################################
- `addFromResourceBundle(String key)` — faces ########### EL ########## Seam ####################################
- `addFromResourceBundle(Severity severity, String key)` — faces ########### EL ########## Seam ####################################
- `clear()` — ##############

`org.jboss.seam.faces.redirect`
: ######################## API ## (##########################)#

- `redirect.viewId` — ########## JSF ### ID ###
- `redirect.conversationPropagationEnabled` — ############################
- `redirect.parameters` — ################# ###############
- `execute()` — ################
- `captureCurrentRequest()` — ### GET ## (#########) ############ ID ####### `execute()` ##############

`org.jboss.seam.faces.httpError`
: HTTP ############## API ###

`org.jboss.seam.ui.renderStampStore`
: A component (session-scoped by default) responsible for maintaining a collection of render stamps. A render stamp is an indicator as to whether a form which was rendered has been submitted. This store is useful when the client-side state saving method of JSF is being used because it puts the determination of whether a form has been posted in the control of the server rather than in the component tree which is maintained on the client.

  To unbind this check from the session (which is one of the main design goals of client-side state saving) an implementation must be provided that stores the render stamps in the application (valid as long as the application is running) or the database (valid across server restarts).

  - `maxSize` — The maximum number of stamps to be kept in the store. Default: 100

These components are installed when the class `javax.faces.context.FacesContext` is available on the classpath.

32.3.

############################

`org.jboss.seam.core.events`
`@Observer` ######## `components.xml` ################################ API ###

- `raiseEvent(String type)` — ##############################
- `raiseAsynchronousEvent(String type)` — EJB3 ##############################
- `raiseTimedEvent(String type, ....)` — EJB3 ##################################
- `addListener(String type, String methodBinding)` — ####################

`org.jboss.seam.core.interpolator`
Strings # JFS EL ############ API ###

- `interpolate(String template)` — `#{...}` ### JSF EL ############################

`org.jboss.seam.core.expressions`
#################### API ###

- `createValueBinding(String expression)` — ####################
- `createMethodBinding(String expression)` — ########################

`org.jboss.seam.core.pojoCache`
JBoss Cache `PojoCache` ##################

- `pojoCache.cfgResourceName` — ########## ####### `treecache.xml` ########

############################

32.4.

############## Seam ################################

`org.jboss.seam.core.locale`
Seam #######

`org.jboss.seam.international.timezone`
Seam ########## ##################

`org.jboss.seam.core.resourceBundle`
Seam ########## ##################### Seam ######### Java ############################

`org.jboss.seam.core.resourceLoader`
##

- `resourceLoader.bundleNames` — Seam ################### Java ############ ###### `messages` ######

org.jboss.seam.international.localeSelector
##

- select() — ################
- localeSelector.locale — ### java.util.Locale ###
- localeSelector.localeString — ##############
- localeSelector.language — ###############
- localeSelector.country — ##############
- localeSelector.variant — ##################
- localeSelector.supportedLocales — jsf-config.xml ################## SelectItem #####
- localeSelector.cookieEnabled — ############################

org.jboss.seam.international.timezoneSelector
##

- select() — ################
- timezoneSelector.timezone — ### java.util.TimeZone ###
- timezoneSelector.timeZoneId — ###############
- timezoneSelector.cookieEnabled — ################################

org.jboss.seam.international.messages
Seam ##

org.jboss.seam.theme.themeSelector
##

- select() — ###############
- theme.availableThemes — ##############
- themeSelector.theme — ###########
- themeSelector.themes — ############## SelectItem ######
- themeSelector.cookieEnabled — ##########################

org.jboss.seam.theme.theme
##################

############################

32.5.

##

`org.jboss.seam.core.conversation`
Seam ###################### API

- `getId()` — ##### ID ######
- `isNested()` — ###################?
- `isLongRunning()` — #################?
- `getId()` — ##### ID ######
- `getParentId()` — #### ID ######
- `getRootId()` — ###### ID ######
- `setTimeout(int timeout)` — ####################
- `setViewId(String outcome)` — ######### ## ### ID ########
- `setDescription(String description)` — ######### ####################################
- `redirect()` — ##################### ID ########## (##############)#
- `leave()` — ############################
- `begin()` — ############# (`@Begin` ###)#
- `beginPageflow(String pageflowName)` — ###################### (`@Begin(pageflow="...")` ###)#
- `end()` — ############# (`@End` ###)#
- `pop()` — ####################
- `root()` — ##################
- `changeFlushMode(FlushModeType flushMode)` — ##################

`org.jboss.seam.core.conversationList`
##################

`org.jboss.seam.core.conversationStack`
################## (breadcrumbs)#

`org.jboss.seam.faces.switcher`
conversation switcher ###

##########################

32.6. jBPM

jBPM ##################

`org.jboss.seam.pageflow.pageflow`
Seam ####### API #####

- `isInProcess()` — ##################### `true` ######
- `getProcessInstance()` — ########## jBPM `ProcessInstance` ######
- `begin(String pageflowName)` — ############################
- `reposition(String nodeName)` — ########################

`org.jboss.seam.bpm.actor`
############ jBPM actor ################### API ###

- `setId(String actorId)` — ######## jBPM #### ID ########
- `getGroupActorIds()` — ############### jBPM #### ID ######### `Set` ######

`org.jboss.seam.bpm.transition`
########## jBPM ################### API ###

- `setName(String transitionName)` — ####### `@EndTask` ############# jBPM ###########

`org.jboss.seam.bpm.businessProcess`
############################## API ###

- `businessProcess.taskId` — ################# ID ###
- `businessProcess.processId` — ################## ID ###
- `businessProcess.hasCurrentTask()` — ##########################?
- `businessProcess.hasCurrentProcess()` — ##########################
- `createProcess(String name)` — ################################
- `startTask()` — ######################
- `endTask(String transitionName)` — ######################
- `resumeTask(Long id)` — ### ID ###################
- `resumeProcess(Long id)` — ### ID ###################

- `transition(String transitionName)` — ##########

`org.jboss.seam.bpm.taskInstance`
jBPM `TaskInstance` ##########

`org.jboss.seam.bpm.processInstance`
jBPM `ProcessInstance` ##########

`org.jboss.seam.bpm.jbpmContext`
######## `JbpmContext` ##########

`org.jboss.seam.bpm.taskInstanceList`
jBPM task list ##########

`org.jboss.seam.bpm.pooledTaskInstanceList`
jBPM pooled task list ##########

`org.jboss.seam.bpm.taskInstanceListForType`
jBPM ##################

`org.jboss.seam.bpm.pooledTask`
pooled task ################

`org.jboss.seam.bpm.processInstanceFinder`
######################

`org.jboss.seam.bpm.processInstanceList`
##################

`org.jboss.seam.bpm.jbpm` ##

32.7.

################################

`org.jboss.seam.web.userPrincipal`
####### `Principal` ##########

`org.jboss.seam.web.isUserInRole`
###################### JSF ############################## `<h:commandButton value="edit" rendered="#{isUserInRole['admin']}"/>`

32.8. JMS

################ `TopicPublisher` ### `QueueSender` ########## (####)#

`org.jboss.seam.jms.queueSession`
JMS `QueueSession` ##########

`org.jboss.seam.jms.topicSession`
JMS `TopicSession` ##############

32.9.

Seam Email ######################

`org.jboss.seam.mail.mailSession`
JavaMail `Session` ############# ###### JNDI ############### (`sessionJndiName` ########)####################### ###### `host` #########

- `org.jboss.seam.mail.mailSession.host` — #### SMTP ############
- `org.jboss.seam.mail.mailSession.port` — #### SMTP ############
- `org.jboss.seam.mail.mailSession.username` — SMTP ######################
- `org.jboss.seam.mail.mailSession.password` — SMTP ######################
- `org.jboss.seam.mail.mailSession.debug` — JavaMail ############### (#####)#
- `org.jboss.seam.mail.mailSession.ssl` — SMTP ## SSL ######### (#########465)#

  `org.jboss.seam.mail.mailSession.tls` — ####### true ### ######### TLS ############
- `org.jboss.seam.mail.mailSession.sessionJndiName` — JNDI ######## javax.mail.Session ######## ##########################

32.10.

##################################### ######################### `components.xml` ############ `install="true"` ######################

`org.jboss.seam.core.init`
Seam ######### ##############

- `org.jboss.seam.core.init.jndiPattern` — ##### Bean ############# JNDI #######
- `org.jboss.seam.core.init.debug` — Seam ############### ##### false ########### ##
- `org.jboss.seam.core.init.clientSideConversations` — `true` ####### Seam ############# `HttpSession` ########################

`org.jboss.seam.core.manager`
Seam ############################## #############

- org.jboss.seam.core.manager.conversationTimeout — ############################
- org.jboss.seam.core.manager.concurrentRequestTimeout — ##
- org.jboss.seam.core.manager.conversationIdParameter — ## ID ################ ###### conversationId ###
- org.jboss.seam.core.manager.conversationIsLongRunningParameter — ###################################### ###### conversationIsLongRunning ###
- org.jboss.seam.core.manager.defaultFlushMode — #### Seam ###################################### ###### AUTO ###

org.jboss.seam.navigation.pages

Seam ######################## #############

- org.jboss.seam.navigation.pages.noConversationViewId — ######################## ID ################
- org.jboss.seam.navigation.pages.loginViewId — ################################### ID ################
- org.jboss.seam.navigation.pages.httpPort — http ##############################
- org.jboss.seam.navigation.pages.httpsPort — https ##############################
- org.jboss.seam.navigation.pages.resources — pages.xml ###################### ###### WEB-INF/pages.xml ###

org.jboss.seam.bpm.jbpm

JbpmConfiguration ############ ### org.jboss.seam.bpm.Jbpm ############

- org.jboss.seam.bpm.jbpm.processDefinitions — ################ jPDL ###############
- org.jboss.seam.bpm.jbpm.pageflowDefinitions — ################ jPDL ###############

org.jboss.seam.core.conversationEntries

##

org.jboss.seam.faces.facesPage

##

org.jboss.seam.persistence.persistenceContexts

###################################

`org.jboss.seam.jms.queueConnection`
Manages a JMS `QueueConnection`. Installed whenever managed `QueueSender` is installed.

- `org.jboss.seam.jms.queueConnection.queueConnectionFactoryJndiName` — JMS `QueueConnectionFactory` # JNDI #### ###### `UIL2ConnectionFactory` ###

`org.jboss.seam.jms.topicConnection`
Manages a JMS `TopicConnection`. Installed whenever managed `TopicPublisher` is installed.

- `org.jboss.seam.jms.topicConnection.topicConnectionFactoryJndiName` — JMS `TopicConnectionFactory` # JNDI #### ###### `UIL2ConnectionFactory` ###

`org.jboss.seam.persistence.persistenceProvider`
JPA ############################

`org.jboss.seam.core.validators`
Hibernate Validator `ClassValidator` ################

`org.jboss.seam.faces.validation`
######################################

`org.jboss.seam.debug.introspector`
Seam Debug Page ########

`org.jboss.seam.debug.contexts`
Seam Debug Page ########

`org.jboss.seam.exception.exceptions`
##################

`org.jboss.seam.transaction.transaction`
############ JTA ## API ###

`org.jboss.seam.faces.safeActions`
URL ######################

32.11.

`org.jboss.seam.async.dispatcher`
############################ Bean

`org.jboss.seam.core.image`
##############

`org.jboss.seam.core.pojoCache`
PojoCache ################

`org.jboss.seam.core.uiComponent`
    ####### ID ######### UIComponents ###########

32.12.

Seam ########## Seam ######### name ################### #### `components.xml` ###### Seam ######## 2

```
<component name="bookingDatabase"
        class="org.jboss.seam.persistence.ManagedPersistenceContext">
   <property name="persistenceUnitJndiName"
>java:/comp/emf/bookingPersistence</property>
</component>

<component name="userDatabase"
        class="org.jboss.seam.persistence.ManagedPersistenceContext">
   <property name="persistenceUnitJndiName"
>java:/comp/emf/userPersistence</property>
</component
>
```

Seam ######### `bookingDatabase` # `userDatabase` ###

<*entityManager*>, `org.jboss.seam.persistence.ManagedPersistenceContext`
    ########################## `EntityManager` ##############

- <*entityManager*>.entityManagerFactory — `EntityManagerFactory` ##########################

  <*entityManager*>.persistenceUnitJndiName — ################## JNDI #### ###### java:/<*managedPersistenceContext*> ###

<*entityManagerFactory*>, `org.jboss.seam.persistence.EntityManagerFactory`
    JPA `EntityManagerFactory` ####### EJB 3.0 ######### JPA ################

- `entityManagerFactory.persistenceUnitName` — ##########

  ################ API JavaDoc ########

<*session*>, `org.jboss.seam.persistence.ManagedSession`
    ############ Hibernate `Session` ##############

- <*session*>.sessionFactory — `SessionFactory` ############################

  <*session*>.sessionFactoryJndiName — ########### JNDI #### ###### java:/<*managedSession*> ###

<*sessionFactory*>, `org.jboss.seam.persistence.HibernateSessionFactory`
Hibernate `SessionFactory` #######

- `<sessionFactory>.cfgResourceName` — ################## `hibernate.cfg.xml` ###

################ API JavaDoc ########

<*managedQueueSender*>, `org.jboss.seam.jms.ManagedQueueSender`
############## JMS `QueueSender` ##############

- <*managedQueueSender*>.queueJndiName — JMS #### JNDI ####

<*managedTopicPublisher*>, `org.jboss.seam.jms.ManagedTopicPublisher`
############## JMS `TopicPublisher` ##############

- <*managedTopicPublisher*>.topicJndiName — JMS ##### JMDI ####

<*managedWorkingMemory*>, `org.jboss.seam.drools.ManagedWorkingMemory`
############ Drools `WorkingMemory` ##############

- <*managedWorkingMemory*>.ruleBase — `RuleBase` ####################

<*ruleBase*>, `org.jboss.seam.drools.RuleBase`
############ Drools `RuleBase` ############ ############################ ############################

- <*ruleBase*>.ruleFiles — Drools ################

  <*ruleBase*>.dslFile — Drools DSL #####

<*entityHome*>, `org.jboss.seam.framework.EntityHome`

<*hibernateEntityHome*>, `org.jboss.seam.framework.HibernateEntityHome`

<*entityQuery*>, `org.jboss.seam.framework.EntityQuery`

<*hibernateEntityQuery*>, `org.jboss.seam.framework.HibernateEntityQuery`

Seam JSF

Seam ## Seam ######## JSF ########################## JSF ## Seam ########JBoss RichFaces#Apache MyFaces Trinidad ################## Tomahawk ####################

33.1.

############## ############## "s" ########## (facelets ##)#

```
<html xmlns="http://www.w3.org/1999/xhtml"
    xmlns:s="http://jboss.com/products/seam/taglib"
>
```

UI####(examples/ui)####################

33.1.1. #######

33.1.1.1. `<s:button>`

##

################################ ###############

##

- `value` — ####
- `action` — ########################
- `view` — ##### JSF view id #
- `fragment` — ########### ID #
- `disabled` — ##########
- `propagation` — determines the conversation propagation style: `begin`, `join`, `nested`, `none`, `end` or `endRoot`.
- `pageflow` — ############ (`propagation="begin"` ### `propagation="join"` #######)#
- `includePageParams` — when set to false, page parameters defined in `pages.xml` will be excluded from rendering.

###

```
<s:button id="cancel"
        value="Cancel"
        action="#{hotelBooking.cancel}"/>
```

<s:link /> ### view # action ##########
##

<s:button />##########(######JSF##########)###############

33.1.1.2. <s:conversationId>

##

JSF ############## ID ###### (# <h:commandLink />#<s:button />)#

##

##

33.1.1.3. <s:taskId>

##

#####{task}############(######JSF######)####ID#######

##

##

33.1.1.4. <s:link>

##

################################ ################

<s:link />##########(######JSF##########)###############

##

- value — ####
- action — ######################
- view — ##### JSF view id #

- `fragment` — ########## ID #
- `disabled` — ##########
- `propagation` — determines the conversation propagation style: `begin`, `join`, `nested`, `none`, `end` or `endRoot`.
- `pageflow` — ########## (`propagation="begin"` ### `propagation="join"` #######)
- `includePageParams` — when set to false, page parameters defined in `pages.xml` will be excluded from rendering.

###

```
<s:link id="register" view="/register.xhtml"
      value="Register New User"/>
```

`<s:link /> ### view # action ##########`
`########################################`

33.1.1.5. `<s:conversationPropagation>`

##

########## (###### JSF ######) ################### *Facelets* #####

##

- `type` — determines the conversation propagation style: `begin`, `join`, `nested`, `none`, `end` or `endRoot`.
- `pageflow` — ########## (`propagation="begin"` ### `propagation="join"` #######)

###

```
<h:commandButton value="Apply" action="#{personHome.update}">
  <s:conversationPropagation type="join" />
</h:commandButton
>
```

33.1.1.6. `<s:defaultAction>`

##

enter##

######### (###<h:commandButton />#<a:commandButton /> ### <tr:commandButton />)#################

###########ID##

##

##

###

```
<h:commandButton id="foo" value="Foo" action="#{manager.foo}">
  <s:defaultAction />
</h:commandButton
>
```

33.1.2.

33.1.2.1. <s:convertDateTime>

##

Seam ##########################

##

##

###

```
<h:outputText value="#{item.orderDate}">
  <s:convertDateTime type="both" dateStyle="full"/>
</h:outputText
>
```

33.1.2.2. <s:convertEntity>

##

################################ ###################################

###JSF################

##

##

##

`<s:convertEntity />` # *Seam########## (Seam managed transaction)* (*#9.2. #Seam ########## ##*) ###############

############## (Managed Persistence Context) # `entityManager` ############ components.xml############:

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:ui="http://jboss.com/products/seam/ui">

  <ui:jpa-entity-loader entity-manager="#{em}" />
```

Hibernate ##### #########components.xml############:

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:ui="http://jboss.com/products/seam/ui">

  <ui:hibernate-entity-loader />
```

Hibernate ##### *(Managed Hibernate Session)* # `session` ############components.xml############:

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:ui="http://jboss.com/products/seam/ui">

  <ui:hibernate-entity-loader session="#{hibernateSession}" />
```

##`components.xml`
##

```
<components xmlns="http://jboss.com/products/seam/components"
        xmlns:ui="http://jboss.com/products/seam/ui">

   <ui:entity-converter name="standardEntityConverter" entity-loader="#{standardEntityLoader}"
 />

                        <ui:jpa-entity-loader name="standardEntityLoader" entity-
manager="#{standardEntityManager}" />
```

```
  <ui:entity-converter name="restrictedEntityConverter" entity-loader="#{restrictedEntityLoader}"
 />

                                <ui:jpa-entity-loader          name="restrictedEntityLoader"          entity-
manager="#{restrictedEntityManager}" />
```

```
<h:selectOneMenu value="#{person.continent}">
  <s:selectItems value="#{continents.resultList}" var="continent"
                label="#{continent.name}" />
   <f:converter converterId="standardEntityConverter" />
</h:selectOneMenu
>
```

###

```
<h:selectOneMenu value="#{person.continent}" required="true">
  <s:selectItems value="#{continents.resultList}" var="continent"
                label="#{continent.name}"
                noSelectionLabel="Please Select..."/>
  <s:convertEntity />
</h:selectOneMenu
>
```

33.1.2.3. `<s:convertEnum>`

##

enum ######################## ##

##

##

###

```
<h:selectOneMenu value="#{person.honorific}">
  <s:selectItems value="#{honorifics}" var="honorific"
                label="#{honorific.label}"
                noSelectionLabel="Please select" />
  <s:convertEnum />
</h:selectOneMenu
```

```
>
```

33.1.2.4. <s:convertAtomicBoolean>

##

java.util.concurrent.atomic.AtomicBoolean #### javax.faces.convert.Converter ###

##

##

###

```
<h:outputText value="#{item.valid}">
  <s:convertAtomicBoolean />
</h:outputText
>
```

33.1.2.5. <s:convertAtomicInteger>

##

java.util.concurrent.atomic.AtomicInteger #### javax.faces.convert.Converter ###

##

##

###

```
<h:outputText value="#{item.id}">
  <s:convertAtomicInteger />
</h:outputText
>
```

33.1.2.6. <s:convertAtomicLong>

##

java.util.concurrent.atomic.AtomicLong #### javax.faces.convert.Converter ###

##

##

###

```
<h:outputText value="#{item.id}">
  <s:convertAtomicLong />
</h:outputText
>
```

33.1.2.7. `<s:validateEquality>`

##

Tag to nest inside an input control to validate that its parent's value is equal (or not equal!) to the referenced control's value.

##

- `for` — ################ID
- `message` — ###############
- `required` — False will disable a check that a value at all is inputted in fields.
- `messageId` — ##############ID
- `operator` — What operator to use when comparing the values Valid operators are:
  - `equal` — Validates that value.equals(forValue)
  - `not_equal` — Validates that !value.equals(forValue)
  - `greater` — Validates that ((Comparable)value).compareTo(forValue) > 0
  - `greater_or_equal` — Validates that ((Comparable)value).compareTo(forValue) >= 0
  - `less` — Validates that ((Comparable)value).compareTo(forValue) <0
  - `less_or_equal` — Validates that ((Comparable)value).compareTo(forValue) <= 0

###

```
<h:inputText id="name" value="#{bean.name}"/>
<h:inputText id="nameVerification" >
  <s:validateEquality for="name" />
</h:inputText
>
```

33.1.2.8. `<s:validate>`

##

############### Hibernate Validator ################### JSF ###############

##

##

###

```
<h:inputText id="userName" required="true"
             value="#{customer.userName}">
  <s:validate />
</h:inputText>
<h:message for="userName" styleClass="error" />
```

33.1.2.9. `<s:validateAll>`

##

############### Hibernate Validator ######################### JSF ###############

##

##

###

```
<s:validateAll>
  <div class="entry">
    <h:outputLabel for="username"
>Username:</h:outputLabel>
    <h:inputText id="username" value="#{user.username}"
                 required="true"/>
    <h:message for="username" styleClass="error" />
  </div>
  <div class="entry">
    <h:outputLabel for="password"
>Password:</h:outputLabel>
    <h:inputSecret id="password" value="#{user.password}"
                   required="true"/>
    <h:message for="password" styleClass="error" />
  </div>
  <div class="entry">
```

```
   <h:outputLabel for="verify"
>Verify Password:</h:outputLabel>
   <h:inputSecret id="verify" value="#{register.verify}"
                 required="true"/>
   <h:message for="verify" styleClass="error" />
 </div>
</s:validateAll
>
```

33.1.3.

33.1.3.1. `<s:decorate>`

##

`############ required="true" ########### JSF ######## "##" ####`

##

- `template` — ################facelet######
- `enclose` — if true, the template used to decorate the input field is enclosed by the element specified with the "element" attribute. By default this is a div element.
- `element` — the element to enclose the template used to decorate the input field. By default, the template is enclosed with a div element.

`#{invalid} # #{required} # s:decorate ##########; ########################## #{required} # true ######################## #{invalid} # true ########`

###

```
<s:decorate template="edit.xhtml">
  <ui:define name="label"
>Country:</ui:define>
    <h:inputText value="#{location.country}" required="true"/>
  </s:decorate
>
```

```
<ui:composition xmlns="http://www.w3.org/1999/xhtml"
  xmlns:ui="http://java.sun.com/jsf/facelets"
  xmlns:h="http://java.sun.com/jsf/html"
  xmlns:f="http://java.sun.com/jsf/core"
  xmlns:s="http://jboss.com/products/seam/taglib">
```

```
  <div
>

    <s:label styleClass="#{invalid?'error':''}">
      <ui:insert name="label"/>
      <s:span styleClass="required" rendered="#{required}"
>*</s:span>
    </s:label>

    <span class="#{invalid?'error':''}">
      <s:validateAll>
        <ui:insert/>
      </s:validateAll>
    </span>

    <s:message styleClass="error"/>

  </div
>

</ui:composition
>
```

33.1.3.2. `<s:div>`

##

HTML `<div>` ##########

##

##

###

```
<s:div rendered="#{selectedMember == null}">
  Sorry, but this member does not exist.
</s:div
>
```

33.1.3.3. `<s:span>`

##

HTML ###########

##

- title — span #####

###

```
<s:span styleClass="required" rendered="#{required}" title="Small tooltip"
>*</s:span
>
```

33.1.3.4. <s:fragment>

##

##/#############

##

##

###

```
<s:fragment rendered="#{auction.highBidder ne null}">
  Current bid:
</s:fragment
>
```

33.1.3.5. <s:label>

##

JSF############ "##" ########HTML## <label> ############JSF#################### ## <s:decorate> ###########

##

- style — ############
- styleClass — ###############

###

```
<s:label styleClass="label">
  Country:
</s:label>
<h:inputText value="#{location.country}" required="true"/>
```

33.1.3.6. `<s:message>`

##

############ JSF ######## "##" ####

##

##

###

```
<f:facet name="afterInvalidField">
  <s:span>
     Error: 
    <s:message/>
  </s:span>
</f:facet
>
```

33.1.4. Seam Text

33.1.4.1. `<s:validateFormattedText>`

##

##############Seam Text###############

##

##

33.1.4.2. `<s:formattedText>`

##

Seam Text #######Seam Text #####wiki ## Seam Text ############

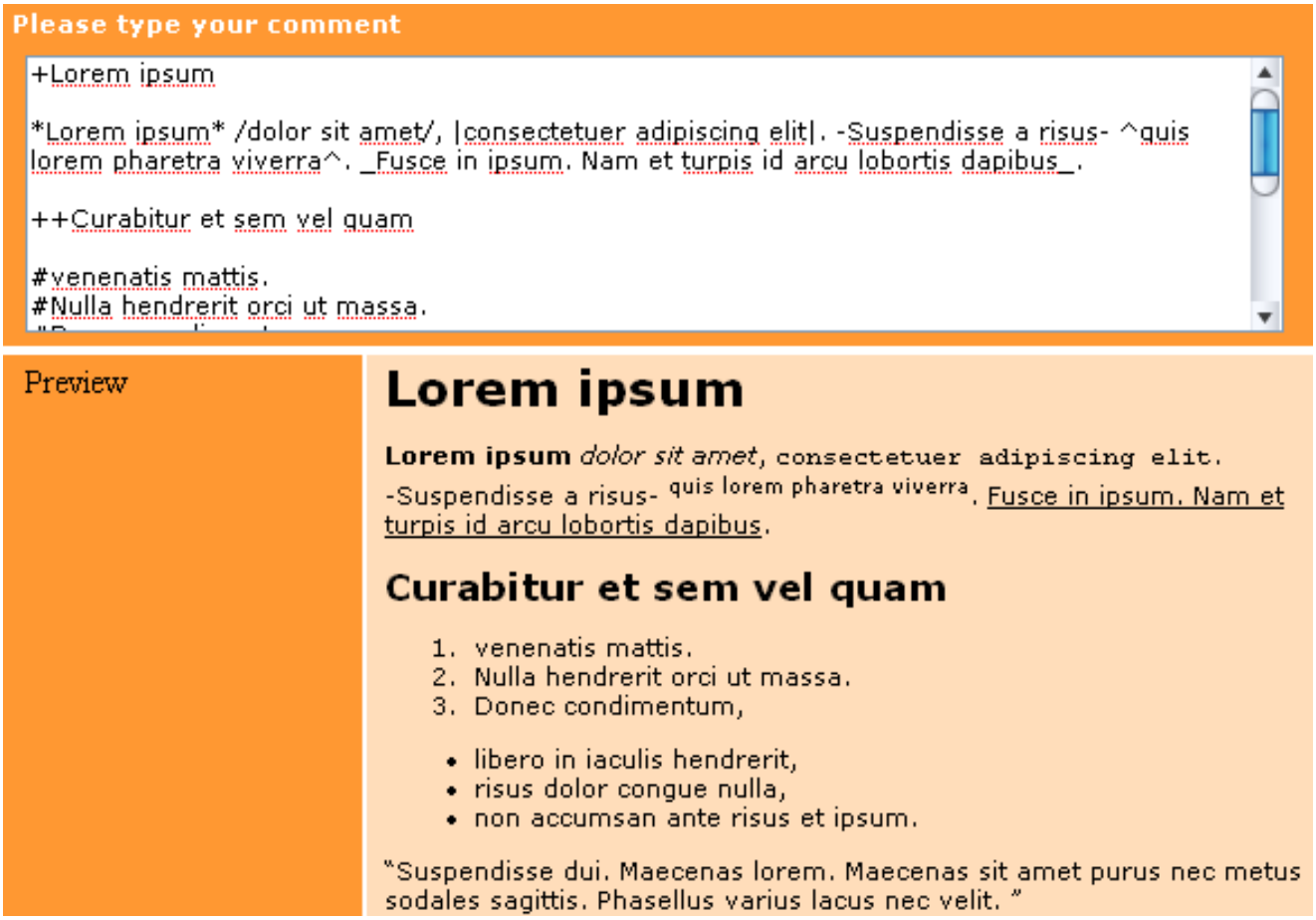
##

- `value` — ############################EL#

###

```
<s:formattedText value="#{blog.text}"/>
```

#



33.1.5. Form support

33.1.5.1. `<s:token>`

##

Produces a random token that is inserted into a hidden form field to help to secure JSF form posts against cross-site request forgery (XSRF) attacks. Note that the browser must have cookies enabled to submit forms that include this component.

##

- `requireSession` — indicates whether the session id should be included in the form signature, hence binding the token to the session. This value can be set to false if the "build before restore" mode of Facelets is activated (the default in JSF 2.0). (default: false)
- `enableCookieNotice` — indicates that a JavaScript check should be inserted into the page to verify that cookies are enabled in the browser. If cookies are not enabled, present a notice to the user that form posts will not work. (default: false)
- `allowMultiplePosts` — indicates whether to allow the same form to be submitted multiple times with the same signature (as long as the view does not change). This is a common need if the form is perform Ajax calls but not rerendering itself or, at the very least, the UIToken component. The preferred approach is to have the UIToken component rerendered on any Ajax call where the UIToken component would be processed. (default: false)

###

```
<h:form>
  <s:token enableCookieNotice="true" requireSession="false"/>
  ...
</h:form>
```

33.1.5.2. `<s:enumItem>`

##

enum #### `SelectItem` #######

##

- `enumValue` — #########
- `label` — `SelectItem` ##################

###

```
<h:selectOneRadio id="radioList"
                  layout="lineDirection"
                  value="#{newPayment.paymentFrequency}">
  <s:convertEnum />
  <s:enumItem enumValue="ONCE"         label="Only Once" />
  <s:enumItem enumValue="EVERY_MINUTE" label="Every Minute" />
  <s:enumItem enumValue="HOURLY"       label="Every Hour" />
  <s:enumItem enumValue="DAILY"        label="Every Day" />
  <s:enumItem enumValue="WEEKLY"       label="Every Week" />
</h:selectOneRadio
```

```
>
```

33.1.5.3. <s:selectItems>

##

List# Set# DataModel ### Array ## `List<SelectItem>` #######

##

- `value` — `List<SelectItem>` ###############EL##
- `var`— ####################################
- `label` — `SelectItem` ################### `var` ##########
- `itemValue` — ## `var` ##############`var` ##########
- `disabled` — true #### `SelectItem` #############`var` ##########
- `noSelectionLabel` — ############ (######) ######### (`required="true"` ################################) #
- `hideNoSelectionLabel` — true ################ `noSelectionLabel` #######

###

```
<h:selectOneMenu value="#{person.age}"
                 converter="ageConverter">
  <s:selectItems value="#{ages}" var="age" label="#{age}" />
</h:selectOneMenu
>
```

33.1.5.4. <s:fileUpload>

##

############################ ################### `multipart/form-data` ######################

```
<h:form enctype="multipart/form-data"
>
```

############ Seam Multipart ########### `web.xml` ###############

```
<filter>
  <filter-name
>Seam Filter</filter-name>
  <filter-class
>org.jboss.seam.servlet.SeamFilter</filter-class>
</filter>

<filter-mapping>
  <filter-name
>Seam Filter</filter-name>
  <url-pattern
>/*</url-pattern>
</filter-mapping
>
```

##

components.xml ### ##############################

- `createTempFiles` — true ##
- `maxRequestSize` — ##############################

#:

```
<component class="org.jboss.seam.web.MultipartFilter">
  <property name="createTempFiles"
>true</property>
  <property name="maxRequestSize"
>1000000</property>
</component
>
```

##

- `data` — ## `byte[]` ### `InputStream` ##################(##)#
- `contentType` — ############################## (#####)#
- `fileName` — ######################## (#####)#
- `fileSize` — ######################### (#####)#

- `accept` — ## ## "images/png,images/jpg"# "images/*"#
- `style` — ############
- `styleClass` — ###############

###

```
<s:fileUpload id="picture" data="#{register.picture}"
          accept="image/png"
          contentType="#{register.pictureContentType}" />
```

33.1.6.

33.1.6.1. `<s:cache>`

##

JBoss Cache ######################## Cache ### `<s:cache>` ########## `pojoCache` ############## JBoss Cache ########################

##

- `key` — ## `key="Document-#{document.id}"`#########
- `enabled` — ####################
- `region` — #### JBoss Cache ####(############################)#

###

```
<s:cache key="entry-#{blogEntry.id}" region="pageFragments">
 <div class="blogEntry">
   <h3
>#{blogEntry.title}</h3>
   <div>
     <s:formattedText value="#{blogEntry.body}"/>
   </div>
   <p>
     [Posted on 
     <h:outputText value="#{blogEntry.date}">
       <f:convertDateTime timezone="#{blog.timeZone}" locale="#{blog.locale}"
                  type="both"/>
```

```
    </h:outputText
>]
   </p>
  </div>
</s:cache
>
```

33.1.6.2. `<s:resource>`

##

A tag that acts a file download provider. It must be alone in the JSF page. To be able to use this contol, web.xml must be set up as follows.

##

```
<servlet>
  <servlet-name>Document Store Servlet</servlet-name>
  <servlet-class>org.jboss.seam.document.DocumentStoreServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>Document Store Servlet</servlet-name>
  <url-pattern>/seam/docstore/*</url-pattern>
</servlet-mapping>
```

##

- `data` — Data that should be downloaded. May be a java.util.File, an InputStream or a byte array.
- `fileName` — Filename of the file to be served
- `contentType` — content type of the file to be downloaded
- `disposition` — disposition to use. Default is inline

###

Here is an example on how to use the tag:

```
<s:resource xmlns="http://www.w3.org/1999/xhtml"
      xmlns:s="http://jboss.com/products/seam/taglib"
      data="#{resources.data}"
      contentType="#{resources.contentType}"
      fileName="#{resources.fileName}" />
```

The bean named `resources` is some backing bean that given some request parameters servers a specific file, see `s:download`.

33.1.6.3. <s:download>

##

Builds a RESTful link to a `<s:resource>`. Nested `f:param` build up the url.

- `src` — Resource file serving files.

##

```
<s:download src="/resources.xhtml">
   <f:param name="fileId" value="#{someBean.downloadableFileId}"/>
</s:download>
```

Will produce something like: `http://localhost/resources.seam?fileId=1`

33.1.6.4. <s:graphicImage>

##

`<h:graphicImage>` ##Seam##########################

`<h:graphicImage>` ################## ##############

##

- `value` — image to display. Can be a path `String` (loaded from the classpath), a `byte[]`, a `java.io.File`, a `java.io.InputStream` or a `java.net.URL`. Currently supported image formats are `image/png`, `image/jpeg`, `image/gif` and `image/bmp`.
- `fileName` — ##

##

############### ###################### Seam #### ##################

`<s:transformImageSize>`

  - `width` — ###########
  - `height` — ############
  - `maintainRatio` — `true` ######## `width` # `height` # ## ##

- `factor` — ####################

`<s:transformImageBlur>`

- `radius` — ############################

`<s:transformImageType>`

- `contentType` — ####### `image/jpeg` ### `image/png` #######

`##############` `-` `org.jboss.seam.ui.graphicImage.ImageTransform` `#####` `UIComponent` `#######` `applyTransform()######` `image.getBufferedImage()` `##############image.setBufferedImage()` ######################################

###

```
<s:graphicImage rendered="#{auction.image ne null}"
            value="#{auction.image.data}">
  <s:transformImageSize width="200" maintainRatio="true"/>
</s:graphicImage
>
```

33.1.6.5. `<s:remote>`

##

Seam Remoting ######## Javascript ###########

##

- `include` — ######## (############)################### Seam Remoting # Javascript ############### *# 25. #######* ##########

###

```
<s:remote include="customerAction,accountAction,com.acme.MyBean"/>
```

33.2.

Seam#####Seam########JSF###################################:

`@Converter`

```
@Name("itemConverter")
```

```
@BypassInterceptors
@Converter
public class ItemConverter implements Converter {

  @Transactional
  public Object getAsObject(FacesContext context, UIComponent cmp, String value) {
                                    EntityManager entityManager = (EntityManager)
 Component.getInstance("entityManager");
    entityManager.joinTransaction();
    // Do the conversion
  }

  public String getAsString(FacesContext context, UIComponent cmp, Object value) {
    // Do the conversion
  }

}
```

```
<h:inputText value="#{shop.item}" converter="itemConverter" />
```

Seam########JSF##JTA##########JPA##########

`@Validator`

```
@Name("itemValidator")
@BypassInterceptors
@org.jboss.seam.annotations.faces.Validator
public class ItemValidator implements javax.faces.validator.Validator {

  public void validate(FacesContext context, UIComponent cmp, Object value)
      throws ValidatorException {
   ItemController ItemController = (ItemController) Component.getInstance("itemController");
    boolean valid = itemController.validate(value);
    if (!valid) {
      throw ValidatorException("Invalid value " + value);
    }
  }
}
```

```
<h:inputText value="#{shop.item}" validator="itemValidator" />
```

Seam########JSF############################Seam##

JBoss EL

Seam # ### Unified Expression Language (EL) ######## JBoss EL ####### JBoss EL # EL ##################################

34.1.

EL ################################## ### JSF ######### (`valueChangeListener`) # JSF

JBoss EL ############### #######

```
<h:commandButton action="#{hotelBooking.bookHotel(hotel)}" value="Book Hotel"/>
```

```
@Name("hotelBooking")
public class HotelBooking {

  public String bookHotel(Hotel hotel) {
    // Book the hotel
  }
}
```

34.1.1.

Java ## ################

```
<h:commandButton action="#{hotelBooking.bookHotel(hotel, user)}" value="Book Hotel"/>
```

`hotel` # `user` ################## `bookHotel()`

################################

```
<h:commandButton
  action="#{hotelBooking.bookHotel(hotel.id, user.username)}"
  value="Book Hotel"/>
```

EL ###################################### ################### ## ##### (## `hotel.id` # `user.username`)# ######################

##
################################## `null` ##############

##########################

```
<h:commandLink action="#{printer.println('Hello world!')}" value="Hello"/>
```

Unified EL ######### ########### Bean ################## ### JavaBean ######### getter # setter ############# JSF ########## (get) ######################## (`rendered` ##)# ######## ###################################### ################################

JBoss EL ################################## #######

```
<h:outputText value="#{person.name}" rendered="#{person.name.length()
> 5}" />
```

1

```
#{searchResults.size()}
```

#{obj.property} ###### #{obj.getProperty()}

################ ################## `productsByColorMethod` ##########

```
#{controller.productsByColor('blue')}
```

34.1.2.

JBoss EL ####################

- *JSP 2.1* ###### — JBoss EL ### JSP 2.1 ########## ######################### JSF 1.2 ############## Facelets ############# ##### JSP 2.0 ##########
- ############## — `<c:forEach />` # `<ui:repeat />` ############ List ### `<h:commandButton />` # `<h:commandLink />` ######################

```
@Factory("items")
public List<Item
```

```
> getItems() {
  return entityManager.createQuery("select ...").getResultList();
}
```

```
<h:dataTable value="#{items}" var="item">
  <h:column>
    <h:commandLink value="Select #{item.name}" action="#{itemSelector.select(item})" />
  </h:column>
</h:dataTable
>
```

<s:link /> # <s:button /> ############## DataModel ####### <dataTable /> (### <rich:dataTable /> > ####################) ### ########## <s:link /> #### <s:button /> ################## (####################) ## DataModel

- *Java ##### MethodExpression ##### —* ### MethodExpression ################ JSF ########## ################ JSF ################### ####### ###################################### ############### 2 ###############
  - Java#### MethodExpression ######### ######################## ######################
  - ### methodExpression.getMethodInfo().getParamTypes() ################### ############## ## MethodExpression ######### getParamTypes() ###############

  ################## Java #### MethodExpression ##########################

34.2.

JBoss EL ###################### ################### (#### #####) ############### #######

```
#{company.departments}
```

############### ### JBoss EL ########################

```
#{company.departments.{d|d.name}}
```

################ ########## d.name ####### d ########################## ####################

###################################### ############### (###) ############################

```
#{company.departments.{d|d.size()}}
```

######################## ############################

```
#{company.departments.{d|d.employees.{emp|emp.lastName}}}
```

############################ ########################

```
#{company.departments.{d|d.employees}}
```

############################### ############################

```
#{company.departments.{d|d.employees.{e|e}}}
```

Facelets # JSP ######### xhtml ### JSP ######################## ########## JBoss EL

Clustering and EJB Passivation

Please note that this chapter is still being reviewed. Tread carefully.

This chapter covers two distinct topics that happen share a common solution in Seam, (web) clustering and EJB passivation. Therefore, they are addressed together in this reference manual. Although performance tends to be grouped in this category as well, it's kept separate because the focus of this chapter is on the programming model and how it's affected by the use of the aforementioned features.

In this chapter you will learn how Seam manages the passivation of Seam components and entity instances, how to activate this feature, and how this feature is related to clustering. You will also learn how to deploy a Seam application into a cluster and verify that HTTP session replication is working properly. Let's start with a little background on clustering and see an example of how you deploy a Seam application to a JBoss AS cluster.

35.1. Clustering

Clustering (more formally web clustering) allows an application to run on two or more parallel servers (i.e., nodes) while providing a uniform view of the application to clients. Load is distributed across the servers in such a way that if one or more of the servers fails, the application is still accessible via any of the surviving nodes. This topology is crucial for building scalable enterprise applications as performance and availability can be improved simply by adding nodes. But it brings up an important question. *What happens to the state that was on the server that failed?*

Since day one, Seam has always provided support for stateful applications running in a cluster. Up to this point, you have learned that Seam provides state management in the form of additional scopes and by governing the life cycle of stateful (scoped) components. But state management in Seam goes beyond creating, storing and destroying instances. Seam tracks changes to JavaBean components and stores the changes at strategic points during the request so that the changes can be restored when the request shifts to a secondary node in the cluster. Fortunately, monitoring and replication of stateful EJB components is already handled by the EJB server, so this feature of Seam is intended to put stateful JavaBeans on par with their EJB cohorts.

But wait, there's more! Seam also offers an incredibly unique feature for clustered applications. In addition to monitoring JavaBean components, Seam ensures that managed entity instances (i.e. JPA and Hibernate entities) don't become detached during replication. Seam keeps a record of the entities that are loaded and automatically loads them on the secondary node. You must, however, be using a Seam-managed persistence context to get this feature. More in depth information about this feature is provided in the second half of this chapter.

Now that you understand what features Seam offers to support a clustered environment, let's look at how you program for clustering.

35.1.1. Programming for clustering

Any session- or conversation-scoped mutable JavaBean component that will be used in a clustered environment must implement the `org.jboss.seam.core.Mutable` interface from the Seam API. As part of the contract, the component must maintain a dirty flag that is reported and reset by the `clearDirty()` method. Seam calls this method to determine if it is necessary to replicate the component. This avoids having to use the more cumbersome Servlet API to add and remove the session attribute on every change of the object.

You also must ensure that all session- and conversation-scoped JavaBean components are Serializable. Additional, all fields of a stateful component (EJB or JavaBean) must Serializable unless the field is marked transient or set to null in a `@PrePassivate` method. You can restore the value of a transient or nullified field in a `@PostActivate` method.

One area where people often get bitten is by using `List.subList` to create a list. The resulting list is not Serializable. So watch out for situations like that. If hit a `java.io.NotSerializableException` and cannot locate the culprit at first glance, you can put a breakpoint on this exception, run the application server in debug mode and attach a debugger (such as Eclipse) to see what deserialization is choking on.

##

Please note that clustering does not work with hot deployable components. But then again, you shouldn't be using hot deployable components in a non-development environment anyway.

35.1.2. Deploying a Seam application to a JBoss AS cluster with session replication

The procedure outlined in this tutorial has been validated with an seam-gen application and the Seam booking example.

In the tutorial, I assume that the IP addresses of the master and slave servers are 192.168.1.2 and 192.168.1.3, respectively. I am intentionally not using the mod_jk load balancer so that it's easier to validate that both nodes are responding to requests and can share sessions.

I'm using the farm deployment method in these instructions, though you could also deploy the application normally and allow the two servers to negotiate a master/slave relationship based on startup order.

##

JBoss AS clustering relies on UDP multicasting provided by jGroups. The SELinux configuration that ships with RHEL/Fedora blocks these packets by default. You

can allow them to pass by modifying the iptables rules (as root). The following commands apply to an IP address that matches 192.168.1.x.

```
/sbin/iptables -I RH-Firewall-1-INPUT 5 -p udp -d 224.0.0.0/4 -j ACCEPT
/sbin/iptables -I RH-Firewall-1-INPUT 9 -p udp -s 192.168.1.0/24 -j ACCEPT
/sbin/iptables -I RH-Firewall-1-INPUT 10 -p tcp -s 192.168.1.0/24 -j ACCEPT
/etc/init.d/iptables save
```

Detailed information can be found on *this page* [http://www.jboss.org/community/docs/DOC-11935] on the JBoss Wiki.

- Create two instances of JBoss AS (just extract the zip twice)
- Deploy the JDBC driver to server/all/lib/ on both instances if not using HSQLDB
- Add `<distributable/>` as the first child element in WEB-INF/web.xml
- Set the `distributable` property on `org.jboss.seam.core.init` to true to enable the ManagedEntityInterceptor (i.e., `<core:init distributable="true"/>`)
- Ensure you have two IP addresses available (two computers, two network cards, or two IP addressses bound to the same interface). I'll assume the two IP address are 192.168.1.2 and 192.168.1.3
- Start the master JBoss AS instance on the first IP

```
./bin/run.sh -c all -b 192.168.1.2
```

  The log should report that there are 1 cluster members and 0 other members.

- Verify that the server/all/farm directory is empty in the slave JBoss AS instance
- Start the slave JBoss AS instance on the second IP

```
./bin/run.sh -c all -b 192.168.1.3
```

  The log should report that there are 2 cluster members and 1 other members. It should also show the state being retrieved from the master.

- Deploy the -ds.xml to server/all/farm of the master instance

  In the log of the master you should see acknowledgement of the deployment. In the log of the slave you should see a corresponding message acknowledging the deployment to the slave.

- Deploy the application to the server/all/farm directory

  In the log of the master you should see acknowledgement of the deployment. In the log of the slave you should see a corresponding message acknowledging the deployment to the slave. Note that you may have to wait up to 3 minutes for the deployed archive to be transfered.

You're application is now running in a cluster with HTTP session replication! But, of course, you are going to want to validate that the clustering actually works.

35.1.3. Validating the distributable services of an application running in a JBoss AS cluster

It's all well and fine to see the application start successfully on two different JBoss AS servers, but seeing is believing. You likely want to validate that the two instances are exchanging HTTP sessions to allow the slave to take over when the master instance is stopped.

Start off by visiting the application running on the master instance in your browser. That will produce the first HTTP session. Now, open up the JBoss AS JMX console on that instance and navigate to the following MBean:

- *Category:* jboss.cache
- *Entry:* service=TomcatClusteringCache
- *Method:* printDetails()

Invoke the printDetails() method. You will see a tree of active HTTP sessions. Verify that the session your browser is using corresponds to one of the sessions in this tree.

Now switch over to the slave instance and invoke the same method in the JMX console. You should see an identical list (at least underneath this application's context path).

So you can see that at least both servers claim to have identical sessions. Now, time to test that the data is serializing and unserializing properly.

Sign in using using the URL of the master instance. Then, construct a URL for the second instance by putting the ;jsessionid=XXXX immediately after the servlet path and changing the IP address. You should see that the session has carried over to the other instance. Now kill the master instance and see that you can continue to use the application from the slave instance. Remove the deployments from the server/all/farm directory and start the instance again. Switch the IP in the URL back to that of the master instance and visit the URL. You'll see that the original session is still being used.

One way to watch objects passivate and activate is to create a session- or conversation-scoped Seam component and implement the appropriate life-cycle methods. You can either use methods from the HttpSessionActivationListener interface (Seam automatically registers this interface on all non-EJB components):

```
public void sessionWillPassivate(HttpSessionEvent e);
public void sessionDidActivate(HttpSessionEvent e);
```

Or you can simply mark two no-argument public void methods with `@PrePassivate` and `@PostActivate`, respectively. Note that the passivation step occurs at the end of every request, while the activation step occurs when a node is called upon.

Now that you understand the big picture of running Seam in a cluster, it's time to address Seam's most mysterious, yet remarkable agent, the ManagedEntityInterceptor.

35.2. EJB Passivation and the ManagedEntityInterceptor

The ManagedEntityInterceptor (MEI) is an optional interceptor in Seam that gets applied to conversation-scoped components when enabled. Enabling it is simple. You just set the distributable property on the org.jboss.seam.init.core component to true. More simply put, you add (or update) the following component declaration in the component descriptor (i.e., components.xml).

```
<core:init distributable="true"/>
```

Note that this doesn't enable replication of HTTP sessions, but it does prepare Seam to be able to deal with passivation of either EJB components or components in the HTTP session.

The MEI serves two distinct scenarios (EJB passivation and HTTP session passivation), although to accomplish the same overall goal. It ensures that throughout the life of a conversation using at least one extended persistence context, the entity instances loaded by the persistence context(s) remain managed (they do not become detached prematurally by a passivation event). In short, it ensures the integrity of the extended persistence context (and therefore its guarantees).

The previous statement implies that there is a challenge that threatens this contract. In fact, there are two. One case is when a stateful session bean (SFSB) that hosts an extended persistence context is passivated (to save memory or to migrate it to another node in the cluster) and the second is when the HTTP session is passivated (to prepare it to be migrated to another node in the cluster).

I first want to discuss the general problem of passivation and then look at the two challenges cited individually.

35.2.1. The friction between passivation and persistence

The persistence context is where the persistence manager (i.e., JPA EntityManager or Hibernate Session) stores entity instances (i.e., objects) it has loaded from the database (via the object-

relational mappings). Within a persistence context, there is no more than one object per unique database record. The persistence context is often referred to as the first-level cache because if the application asks for a record by its unique identifier that has already been loaded into the persistence context, a call to the database is avoided. But it's about more than just caching.

Objects held in the persistence context can be modified, which the persistence manager tracks. When an object is modified, it's considered "dirty". The persistence manager will migrate these changes to the database using a technique known as write-behind (which basically means only when necessary). Thus, the persistence context maintains a set of pending changes to the database.

Database-oriented applications do much more than just read from and write to the database. They capture transactional bits of information that need to be tranfered into the database atomically (at once). It's not always possible to capture this information all on one screen. Additionally, the user might need to make a judgement call about whether to approve or reject the pending changes.

What we are getting at here is that the idea of a transaction from the user's perspective needs to be extended. And that is why the extended persistence context fits so perfectly with this requirement. It can hold such changes for as long as the application can keep it open and then use the built-in capabilities of the persistence manager to push these pending changes to the database without requiring the application developer to worry about the low-level details (a simple call to `EntityManager#flush()` does the trick).

The link between the persistence manager and the entity instances is maintained using object references. The entity instances are serializable, but the persistence manager (and in turn its persistence context) is not. Therefore, the process of serialization works against this design. Serialization can occur either when a SFSB or the HTTP session is passivated. In order to sustain the activity in the application, the persistence manager and the entity instances it manages must weather serialization without losing their relationship. That's the aid that the MEI provides.

35.2.2. Case #1: Surviving EJB passivation

Conversations were initially designed with stateful session beans (SFSBs) in mind, primarily because the EJB 3 specification designates SFSBs as hosts of the extended persistence context. Seam introduces a complement to the extended persistence context, known as a Seam-managed persistence context, which works around a number of limitations in the specification (complex propagation rules and lack of manual flushing). Both can be used with a SFSB.

A SFSB relies on a client to hold a reference to it in order to keep it active. Seam has provided an ideal place for this reference in the conversation context. Thus, for as long as the conversation context is active, the SFSB is active. If an EntityManager is injected into that SFSB using the annotation @PersistenceContext(EXTENDED), then that EntityManager will be bound to the SFSB and remain open throughout its lifetime, the lifetime of the conversation. If an EntityManager is injected using @In, then that EntityManager is maintained by Seam and stored directly in the conversation context, thus living for the lifetime of the conversation independent of the lifetime of the SFSB.

With all of that said, the Java EE container can passivate a SFSB, which means it will serialize the object to an area of storage external to the JVM. When this happens depends on the settings of the individual SFSB. This process can even be disabled. However, the persistence context is not serialized (is this only true of SMPC?). In fact, what happens depends highly on the Java EE container. The spec is not very clear about this situation. Many vendors just tell you not to let it happen if you need the guarantees of the extended persistence context. Seam's approach is more conservative. Seam basically doesn't trust the SFSB with the persistence context or the entity instances. After each invocation of the SFSB, Seam moves the reference to entity instance held by the SFSB into the current conversation (and therefore into the HTTP session), nullifying those fields on the SFSB. It then restores this references at the beginning of the next invocation. Of course, Seam is already storing the persistence manager in the conversation. Thus, when the SFSB passivates and later activates, it has absolutely no averse affect on the application.

##

If you are using SFSBs in your application that hold references to extended persistence contexts, and those SFSBs can passivate, then you must use the MEI. This requirement holds even if you are using a single instance (not a cluster). However, if you are using clustered SFSB, then this requirement also applies.

It is possible to disable passivation on a SFSB. See the *Ejb3DisableSfsbPassivation* [http://www.jboss.org/community/docs/DOC-9656] page on the JBoss Wiki for details.

35.2.3. Case #2: Surviving HTTP session replication

Dealing with passivation of a SFSB works by leveraging the HTTP session. But what happens when the HTTP session passivates? This happens in a clustered environment with session replication enabled. This case is much tricker to deal with and is where a bulk of the MEI infrastructure comes into play. In thise case, the persistence manager is going to be destroyed because it cannot be serialized. Seam handles this deconstruction (hence ensuring that the HTTP session serializes properly). But what happens on the other end. Well, when the MEI sticks an entity instance into the conversation, it embeds the instance in a wrapper that provides information on how to reassociate the instance with a persistence manager post-serialization. So when the application jumps to another node in the cluster (presumably because the target node went down) the MEI infrastruture essentially reconstructs the persistence context. The huge drawback here is that since the persistence context is being reconstructed (from the database), pending changes are dropped. However, what Seam does do is ensure that if the entity instance is versioned, that the guarantees of optimistic locking are upheld. (why isn't the dirty state transfered?)

##

If you are deploying your application in a cluster and using HTTP session replication, you must use the MEI.

35.2.4. ManagedEntityInterceptor wrap-up

The important point of this section is that the MEI is there for a reason. It's there to ensure that the extended persistence context can retain intact in the face of passivation (of either a SFSB or the HTTP session). This matters because the natural design of Seam applications (and conversational state in general) revolve around the state of this resource.

Performance Tuning

This chapter is an attempt to document in one place all the tips for getting the best performance from your Seam application.

36.1. Bypassing Interceptors

For repetitive value bindings such as those found in a JSF dataTable or other iterative control (like `ui:repeat`), the full interceptor stack will be invoked for every invocation of the referenced Seam component. The effect of this can result in a substantial performance hit, especially if the component is accessed many times. A significant performance gain can be achieved by disabling the interceptor stack for the Seam component being invoked. To disable interceptors for the component, add the `@BypassInterceptors` annotation to the component class.

##

It is very important to be aware of the implications of disabling interceptors for a Seam component. Features such as bijection, annotated security restrictions, synchronization and others are unavailable for a component marked with `@BypassInterceptors`. While in most cases it is possible to compensate for the loss of these features (e.g. instead of injecting a component using `@In`, you can use `Component.getInstance()` instead) it is important to be aware of the consequences.

The following code listing demonstrates a Seam component with its interceptors disabled:

```
@Name("foo")
@Scope(EVENT)
@BypassInterceptors
public class Foo
{
  public String getRowActions()
  {
     // Role-based security check performed inline instead of using @Restrict or other security
 annotation
    Identity.instance().checkRole("user");

    // Inline code to lookup component instead of using @In
    Bar bar = (Bar) Component.getInstance("bar");

    String actions;
    // some code here that does something
    return actions;
```

```
  }
}
```

Seam############

Seam####################2############### ###Seam#########################
##############Java# (#############) ####################

######################

37.1. Seam################

####Seam########POJO#################################Seam#################################

############################Seam#################

```
@Stateless
@Scope(EVENT)
@Name("statementOfAccount")
public class StatementOfAccount {

  @In(create=true) EntityManager entityManager

  private double statementTotal;

  @In
  private Customer customer;

  @Create
  public void create() {
    List<Invoice
> invoices = entityManager
      .createQuery("select invoice from Invoice invoice where invoice.customer = :customer")
      .setParameter("customer", customer)
      .getResultList();
    statementTotal = calculateTotal(invoices);
  }

  public double calculateTotal(List<Invoice
> invoices) {
    double total = 0.0;
    for (Invoice invoice: invoices)
    {
      double += invoice.getTotal();
    }
    return total;
  }
```

```
  // getter and setter for statementTotal

}
```

calculateTotal###

```
public class StatementOfAccountTest {

   @Test
   public testCalculateTotal {
     List<Invoice
> invoices = generateTestInvoices(); // A test data generator
     double statementTotal = new StatementOfAccount().calculateTotal(invoices);
     assert statementTotal = 123.45;
   }
}
```

You'll notice we aren't testing retrieving data from or persisting data to the database; nor are we testing any functionality provided by Seam. We are just testing the logic of our POJOs. Seam components don't usually depend directly upon container infrastructure, so most unit testing are as easy as that!

###################################

37.2. Seam##############

###

Seam###Seam#JBoss############*#30.6.1. #Embedded JBoss ##########*#################

```
public class RegisterTest extends SeamTest
{

  @Test
  public void testRegisterComponent() throws Exception
  {

    new ComponentTest() {

      protected void testComponents() throws Exception
      {
```

```
            setValue("#{user.username}", "1ovthafew");
            setValue("#{user.name}", "Gavin King");
            setValue("#{user.password}", "secret");
            assert invokeMethod("#{register.register}").equals("success");
            assert getValue("#{user.username}").equals("1ovthafew");
            assert getValue("#{user.name}").equals("Gavin King");
            assert getValue("#{user.password}").equals("secret");
        }

    }.run();

  }

  ...

}
```

37.2.1.

########################Seam################## ##############################
########################Seam##########

```
@Name("paymentProcessor")
public class PaymentProcessor {
   public boolean processPayment(Payment payment) { .... }
}
```

####################################

```
@Name("paymentProcessor")
@Install(precedence=MOCK)
public class MockPaymentProcessor extends PaymentProcessor {
   public boolean processPayment(Payment payment) {
      return true;
   }
}
```

MOCK###################### ################################Seam# ############
######################################

37.3.

##W

SeamTest########JSF#####################################Seam################################JS

#################################

######################JSF##############

```
<html>
 <head>
  <title
>Register New User</title>
 </head>
 <body>
  <f:view>
   <h:form>
    <table border="0">
     <tr>
      <td
>Username</td>
      <td
><h:inputText value="#{user.username}"/></td>
     </tr>
     <tr>
      <td
>Real Name</td>
      <td
><h:inputText value="#{user.name}"/></td>
     </tr>
     <tr>
      <td
>Password</td>
      <td
><h:inputSecret value="#{user.password}"/></td>
     </tr>
    </table>
    <h:messages/>
    <h:commandButton type="submit" value="Register" action="#{register.register}"/>
   </h:form>
  </f:view>
 </body>
</html
```

```
>
```

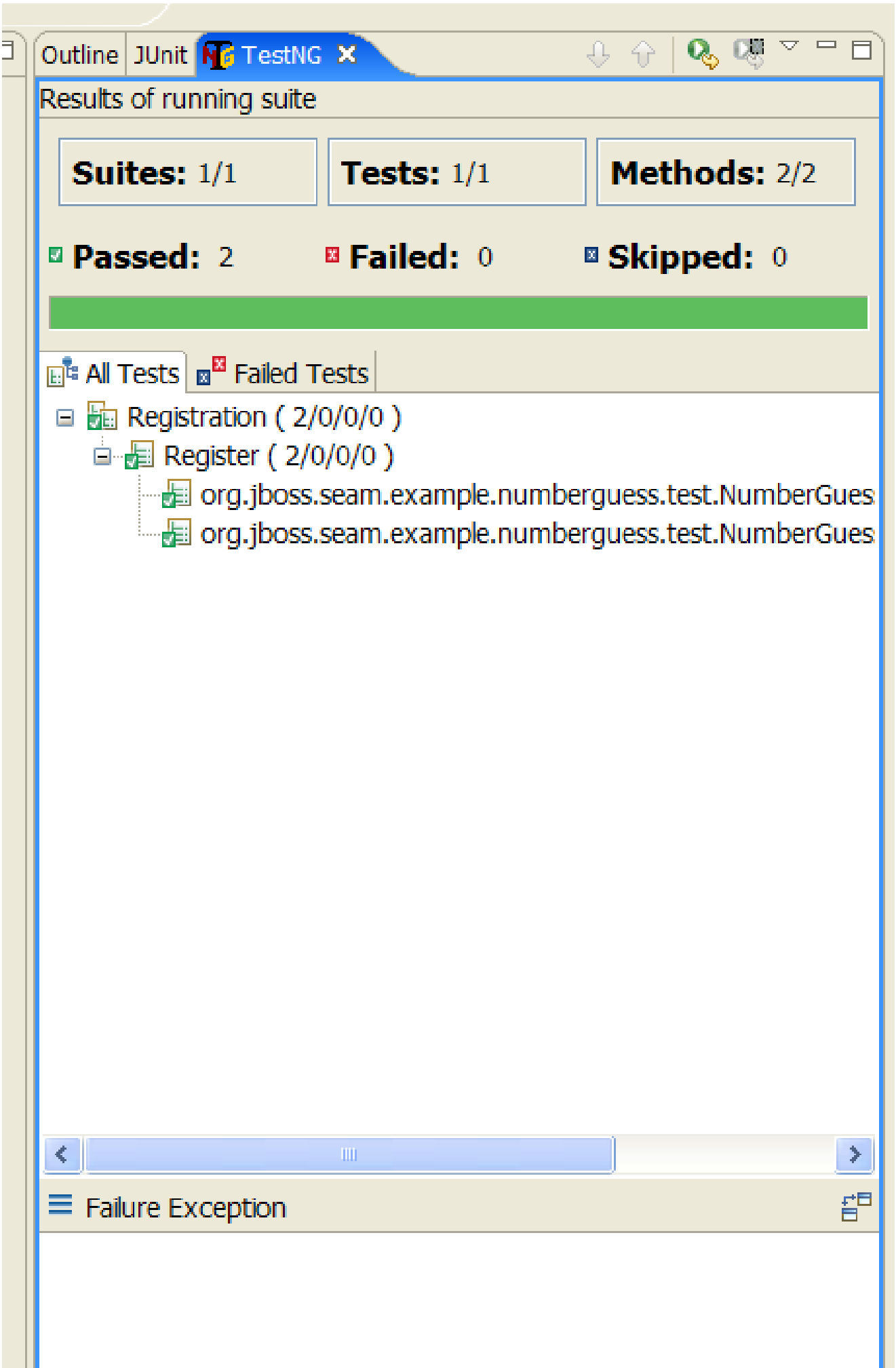
####################Register########################TestNG#######JSF####################

```
public class RegisterTest extends SeamTest
{

  @Test
  public void testRegister() throws Exception
  {

    new FacesRequest() {

      @Override
      protected void processValidations() throws Exception
      {
        validateValue("#{user.username}", "1ovthafew");
        validateValue("#{user.name}", "Gavin King");
        validateValue("#{user.password}", "secret");
        assert !isValidationFailure();
      }

      @Override
      protected void updateModelValues() throws Exception
      {
        setValue("#{user.username}", "1ovthafew");
        setValue("#{user.name}", "Gavin King");
        setValue("#{user.password}", "secret");
      }

      @Override
      protected void invokeApplication()
      {
        assert invokeMethod("#{register.register}").equals("success");
      }

      @Override
      protected void renderResponse()
      {
        assert getValue("#{user.username}").equals("1ovthafew");
        assert getValue("#{user.name}").equals("Gavin King");
        assert getValue("#{user.password}").equals("secret");
      }
```

```
      }.run();

   }

   ...

}
```

########Seam#######SeamTest#####JSF###################SeamTest.FacesRequest#################

Seam##Ant#################Eclipse#TestNG##############


Outline
JUnit
TestNG
Results of running suite
Suites: 1/1
Tests: 1/1
Methods: 2/2
Passed: 2
Failed: 0
Skipped: 0
All Tests
Failed Tests
Registration (2/0/0/0)
Register (2/0/0/0)
org.jboss.seam.example.numberguess.test.NumberGues
org.jboss.seam.example.numberguess.test.NumberGues
Failure Exception

37.3.1.

seam-gen###ant, maven, Eclipse######################

######################

37.1.

####ID	########ID	Seam ####
org.jboss.seam.embedded	hibernate-all	lib/test/hibernate-all.jar
org.jboss.seam.embedded	jboss-embedded-all	lib/test/jboss-embedded-all.jar
org.jboss.seam.embedded	thirdparty-all	lib/test/thirdparty-all.jar
org.jboss.seam.embedded	jboss-embedded-api	lib/jboss-embedded-api.jar
org.jboss.seam	jboss-seam	lib/jboss-seam.jar
org.jboss.el	jboss-el	lib/jboss-el.jar
javax.faces	jsf-api	lib/jsf-api.jar
javax.el	el-api	lib/el-api.jar
javax.activation	javax.activation	lib/activation.jar

#####JBoss####################JBoss AS############jboss-system.jar##lib/ ################################Drools#jBPM#############

#####JBoss######bootstrap/######################

############jar###JPA#Seam###

####################java:/ DefaultDS######JBoss######HSQL#####################################foo-ds.xml#bootstrap/deploy###############

37.3.2. ##########SeamTest###

Seam##TestNG###############JUnit############################

######AbstractSeamTest##########

- ##############super.begin()####
- ##############super.end()####

- ################super.setupClass()#####################################
- ############super.cleanupClass()####
- ##########super.startSeam()####Seam######
- ##########super.stopSeam()####Seam######

37.3.3.

If you want to insert or clean data in your database before each test you can use Seam's integration with DBUnit. To do this, extend `DBUnitSeamTest` rather than `SeamTest`.

You have to provide a dataset for DBUnit.

##

DBUnit supports two formats for dataset files, flat and XML. Seam's `DBUnitSeamTest` assumes the flat format is used, so make sure that your dataset is in this format.

```
<dataset>

  <ARTIST
    id="1"
    dtype="Band"
    name="Pink Floyd" />

  <DISC
    id="1"
    name="Dark Side of the Moon"
    artist_id="1" />

</dataset
>
```

In your test class, configure your dataset with overriding `prepareDBUnitOperations()`:

```
protected void prepareDBUnitOperations() {
   beforeTestOperations.add(
     new DataSetOperation("my/datasets/BaseData.xml")
   );
}
```

`DataSetOperation####################################DatabaseOperation.CLEAN_INSERT##############`

`############################afterTestOperations#####################`

You need to tell DBUnit which datasource you are using. This is accomplished by defining a *test parameter* [http://testng.org/doc/documentation-main.html#parameters-testng-xml] named `datasourceJndiName` in testng.xml as follows:

```
<parameter name="datasourceJndiName" value="java:/seamdiscsDatasource"/>
```

DBUnitSeamTest has support for MySQL and HSQL - you need to tell it which database is being used, otherwise it defaults to HSQL:

```
<parameter name="database" value="MYSQL" />
```

It also allows you to insert binary data into the test data set (n.b. this is untested on Windows). You need to tell it where to locate these resources on your classpath:

```
<parameter name="binaryDir" value="images/" />
```

You do not have to configure any of these parameters if you use HSQL and have no binary imports. However, unless you specify `datasourceJndiName` in your test configuration, you will have to call `setDatabaseJndiName()` before your test runs. If you are not using HSQL or MySQL, you need to override some methods. See the Javadoc of `DBUnitSeamTest` for more details.

37.3.4. Seam#########

##
################

Seam##################

```
public class MailTest extends SeamTest {

  @Test
  public void testSimpleMessage() throws Exception {

    new FacesRequest() {
```

```
        @Override
        protected void updateModelValues() throws Exception {
          setValue("#{person.firstname}", "Pete");
          setValue("#{person.lastname}", "Muir");
          setValue("#{person.address}", "test@example.com");
        }

        @Override
        protected void invokeApplication() throws Exception {
          MimeMessage renderedMessage = getRenderedMailMessage("/simple.xhtml");
          assert renderedMessage.getAllRecipients().length == 1;
          InternetAddress to = (InternetAddress) renderedMessage.getAllRecipients()[0];
          assert to.getAddress().equals("test@example.com");
        }

      }.run();
    }
}
```

#####FacesRequest#######invokeApplication######viewId##################getRenderedMailMessage(vie

##JSF###

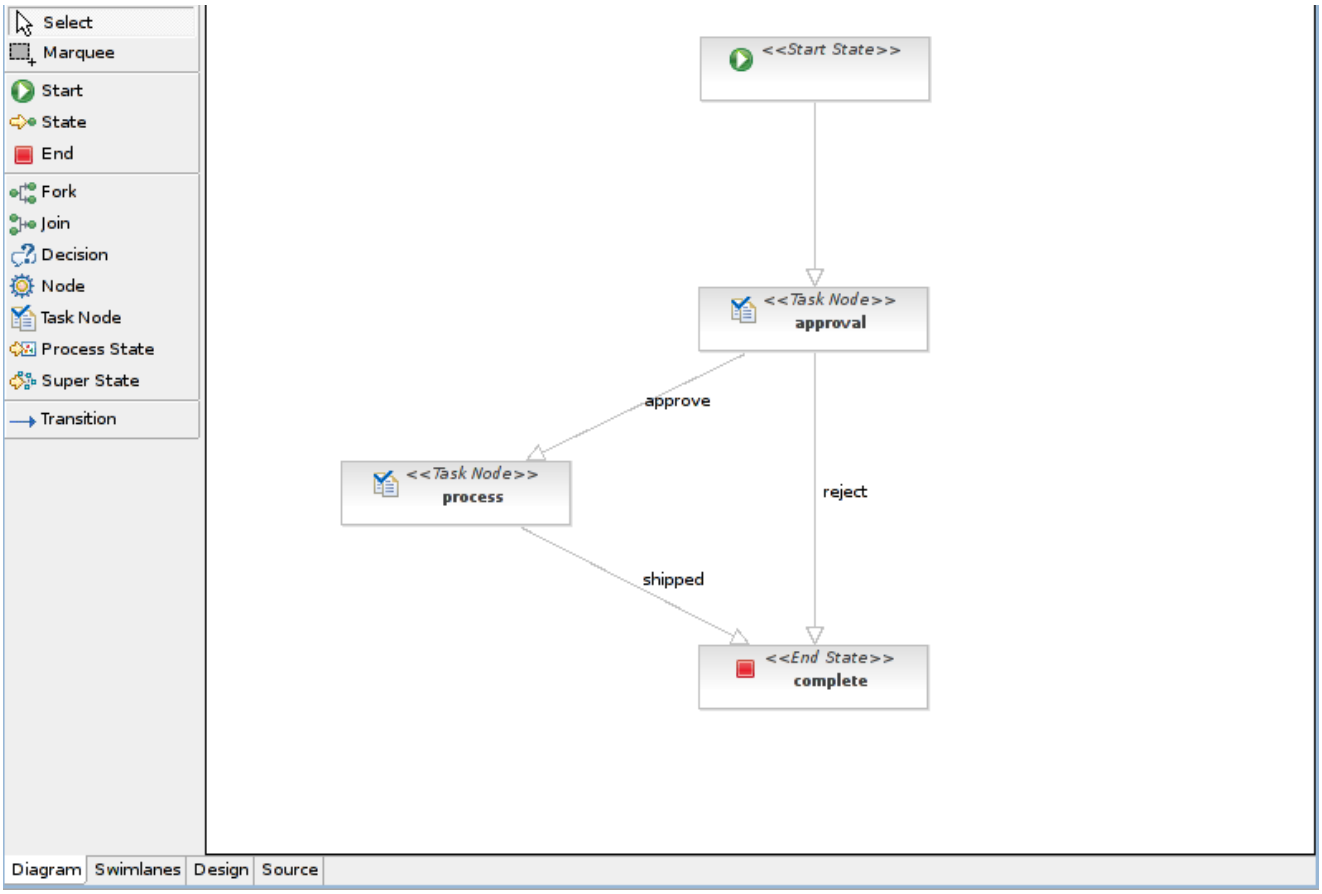
Seam

38.1. jBPM

The jBPM designer and viewer will let you design and view in a nice way your business processes and your pageflows. This convenient tool is part of JBoss Eclipse IDE and more details can be found in the jBPM's *documentation* [http://docs.jboss.com/jbpm/v3/gpd/]

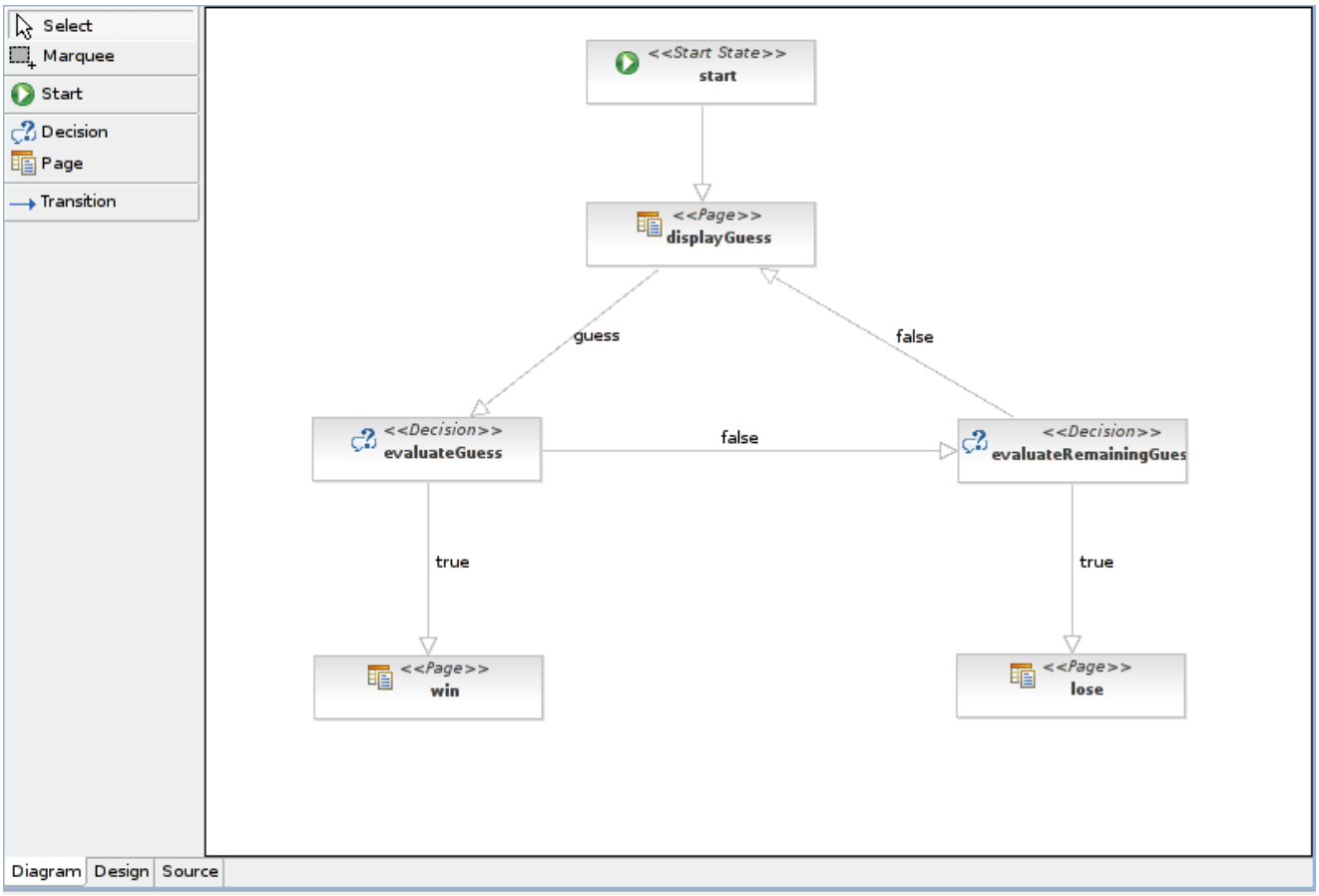
38.1.1.

##



38.1.2.

########################## ##########################
##


Select
Marquee
Start
Decision
Page
Transition
<<Start State>>
start
<<Page>>
displayGuess
guess
false
<<Decision>>
evaluateGuess
false
<<Decision>>
evaluateRemainingGues
true
true
<<Page>>
win
<<Page>>
lose
Diagram
Design
Source

BEA Weblogic #### Seam

WebLogic 10.3
#BEA###############J2EE######Seam#########WebLogic#################################WebLog

###WebLogic##########################Seam#JEE5####################JPA#####################
`seam-gen` ##

39.1. WebLogic##########

##10.3###################BEA#####################

- `Weblogic 10.0.MP1` — ###### ### [http://www.oracle.com/technology/software/products/ias/htdocs/wls_main.html]

  10.0.MP1##EJB###`varargs`##########`transie`
  #Weblogic##EJB3####

- `Weblogic 10.3` — ######### [http://www.oracle.com/technology/software/products/ias/htdocs/wls_main.html]

  ###WebLogic###10.0.MP1#######EJB#########
  ###############*#39.2.1. #Weblogic##EJB3####*
  ######################WebLogic##jar###################

Weblogic EJB ######### `jboss-seam.jar`

Seam 2.0.2.CR2
##########WebLogic####jar#################`TimerServiceDispatcher`
#############EJB################EJB##2################BEA#############`jee5/booking` ##########jar#################

39.1.1. 10.3#######

####Weblogic
10.3##BEA################# *Weblogic 10.3 #########* [http://edocs.bea.com/wls/docs103/] . ###################### RHEL #####5
###########

1. 10.3
   ##

2. #####`server103_XX.bin` ######################

```
chmod a+x server103_XX.bin############
```

3. ##############

```
./server103_XX.bin
```

4. ####################BEA##################################BEA##############################
   $BEA_HOME ############

```
/jboss/apps/bea
```

5. ##############Complete#######################################struts#beehive####################

6. ##

39.1.2. Weblogic############

WebLogic######JBoss###

1. Weblogic############

```
$BEA_HOME/wlserver_10.3/common/bin/config.sh
```

2. ####################Weblogic
   Server###

3. ################################

4. ##################Development Mode#######JDK###########

5. ################################ No###########

6. #########seam_examples ###################################

39.1.3. ##### ##/##/####

###

- #######

  ############ $BEA_HOME/user_projects/domains/seam_examples/bin ######### ./
  startWeblogic.sh################

- ############

  #########http://127.0.0.1:7001/
  console###

- #######

  ####2#3############

  - ##########################

    1. ###########seam_examples##########
    2. ##########seam_examples############
    3. ##########AdminServer##################
    4. ############## Shutdown################ When work completes #Force shutdown now#########

  - ###############Ctrl-C####

    ##

- 

  Weblogic###############

  ###############/autodeploy
  ####################NoClassDefFound##########################Weblogic############ deployed EAR/WAR files###

39.1.4. Weblogic#JSF#######

#################Weblogic#JSF 1.2###################################Weblogic######JSF#####################
Weblogic 10.3 Configuring JSF and JSTL Libraries [http://edocs.bea.com/wls/docs103/webapp/configurejsfandjtsl.html]##########

1. ########################Deployments #########
2. ####################Install#########
3. ###############$BEA_HOME/wlserver_10.3/common/deployable-libraries ############ jsf-1.2.war############Next #############
4. Install this deployment as a library ####################Install Application Assistant ###### Next ###############

5. Optional Settings######Next###############

6. Yes, take me to the deployment's configuration screen.####################
   Review your choices and click Finish###### Finish###############

7. ######################################Settings for jsf(1.2,1.2.3.1)####
   Deployment Order# 99############### Save ###############

JSF## jsf-api.jar
##jsf-1.2.war## javax.jsf_1.2.0.0.jar (jsf-api.jar)############################

39.2. jee5/booking####

Weblogic#EJB####Seam######################################BEA#################################booking###################################

39.2.1. Weblogic##EJB3###

Weblogic####################Weblogic#########################EJB##############################W9.X#10.0.MP1####################10.3#####################################

39.2.1.1. varargs###

#######Weblogic#EJB######transient#########varargs################################BEA########TimerServiceDispatcher)##10.0.MP1#####################

```
java.io.IOException: Compiler failed executable.exec:
/jboss/apps/bea/wlserver_10.0/user_projects/domains/seam_examples/servers/AdminServer
/cache/EJBCompilerCache/5yo5dk9ti3yo/org/jboss/seam/async/
TimerServiceDispatcher_qzt5w2_LocalTimerServiceDispatcherImpl.java:194: modifier transient
not allowed here
  public transient javax.ejb.Timer scheduleAsynchronousEvent(java.lang.String arg0,
  java.lang.Object[] arg1)
                       ^
/jboss/apps/bea/wlserver_10.0/user_projects/domains/seam_examples/servers/AdminServer
/cache/EJBCompilerCache/5yo5dk9ti3yo/org/jboss/seam/async/
TimerServiceDispatcher_qzt5w2_LocalTimerServiceDispatcherImpl.java:275: modifier transient
not allowed here
  public transient javax.ejb.Timer scheduleTimedEvent(java.lang.String arg0,
  org.jboss.seam.async.TimerSchedule arg1, java.lang.Object[] arg2)
```

######Weblogic 10.3##############BEA##############Weblogic 10.0.MP1#########(CR327275)########BEA###############

##########BEA###############

39.2.1.2. ######EJB#######

#####10.0.MP1######CR327275##########################BEA#########BEA#10.0.MP1###############

#####Weblogic#########################EJB##

```
<<Error
> <EJB
> <BEA-012036
> <Compiling generated EJB classes produced the following Java compiler error message:
<Compilation Error
> TimerServiceDispatcher_qzt5w2_Impl.java: The type TimerServiceDispatcher_qzt5w2_Impl
must implement the inherited abstract method
TimerServiceDispatcher_qzt5w2_Intf.scheduleTimedEvent(String, Schedule, Object[])
<Compilation Error
> TimerServiceDispatcher_qzt5w2_LocalTimerServiceDispatcherImpl.java: Type mismatch:
cannot convert from Object to Timer
<Compilation Error
> TimerServiceDispatcher_qzt5w2_LocalTimerServiceDispatcherImpl.java: Type mismatch:
cannot convert from Object to Timer
>
<Error
> <Deployer
> <BEA-149265
> <Failure occurred in the execution of deployment request with ID '1223409267344' for task '0'.
Error is: 'weblogic.application.ModuleException: Exception preparing module: EJBModule(jboss-
seam.jar)
```

##############Weblogic 10.3#################################Weblogic 10.0.MP1############10.3##########Seam#jar###BEA###JBoss###############

Seam 2.0.2.CR2#########Seam######EJB#########Weblogic############Weblogic#jar###############jar#######lib/interop###########jar#####jboss-seam-wls-compatible.jar###########jar#####jboss-seam.jar#######TimerServiceDispatcher EJB###########jar#################jboss-seam-wls-compatible.jar# jboss-seam.jar#################################jee5/booking#################################jar#####TimerServiceDispatcher#################

39.2.2. jee5/booking

jee5/booking#########################

39.2.2.1. hsql##########

##

1. Weblogic##############hsqldb.jar############### `cp $SEAM_HOME/lib/hsqldb.jar $BEA_HOME/user_projects/domains/seam_examples/lib`
2. ############################## *#39.1.3. ###### ##/##/#### ###*
3. #########seam_examples - Services- JDBC - Data Sources#########
4. ##################New##############
5. ###############
   a. Name: `seam-jee5-ds`
   b. JNDI Name: `seam-jee5-ds`
   c. Database Type and Driver: `other`
   d. `Next` ##############
6. `Transaction Options` ######Next##############
7. `Connection Properties` ################
   a. Database Name: `hsqldb`
   b. Host Name: `127.0.0.1`
   c. Port: `9001`
   d. Username:`sa` #################
   e. Password: ###
   f. `Next` ##############
8. `Connection Properties` ################
   a. Driver Class Name: `org.hsqldb.jdbcDriver`
   b. URL: `jdbc:hsqldb:.`
   c. Username: `sa`
   d. Password: ###
   e. ########################
   f. `Next` ##############
9. #####AdminServer################################

39.2.2.2.

###Weblogic##############Seam#############################

`resources/META-INF/persistence.xml`

- `jta-data-source#####################`

```
        <jta-data-source
>seam-jee5-ds</jta-data-source
>
```

- ####glassfish############################
- weblogic############################

```
<property name="hibernate.dialect"
        value="org.hibernate.dialect.HSQLDialect"/>
<property name="hibernate.transaction.manager_lookup_class"
        value="org.hibernate.transaction.WeblogicTransactionManagerLookup"/>
```

`resources/META-INF/weblogic-application.xml`

- ######################

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<weblogic-application>
   <library-ref>
      <library-name
>jsf</library-name>
      <specification-version
>1.2</specification-version>
      <implementation-version
>1.2</implementation-version>
      <exact-match
>false</exact-match>
   </library-ref>
   <prefer-application-packages>
      <package-name
>antlr.*</package-name>
```

```
    </prefer-application-packages>
</weblogic-application>
```

- ############################## `library-ref`#weblogic###########JSF## `prefer-application-packages` weblogic#`antlr`#jar#####################hibernate################

`resources/META-INF/ejb-jar.xml`

- ###########Weblogic######`sessionBeanInterceptor`################Bean#####################

  `assembly-descriptor` #################

```
<assembly-descriptor>
  <interceptor-binding
>
    <ejb-name
>AuthenticatorAction</ejb-name>
    <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
  </interceptor-binding>
  <interceptor-binding
>
    <ejb-name
>BookingListAction</ejb-name>
    <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
  </interceptor-binding>
  <interceptor-binding
>
    <ejb-name
>RegisterAction</ejb-name>
    <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
  </interceptor-binding>
  <interceptor-binding
>
    <ejb-name
>ChangePasswordAction</ejb-name>
    <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
  </interceptor-binding>
  <interceptor-binding
```

```
>
    <ejb-name
>HotelBookingAction</ejb-name>
    <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
  </interceptor-binding>
  <interceptor-binding
>
    <ejb-name
>HotelSearchingAction</ejb-name>
    <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
  </interceptor-binding>
  <interceptor-binding
>
    <ejb-name
>EjbSynchronizations</ejb-name>
    <interceptor-class
>org.jboss.seam.ejb.SeamInterceptor</interceptor-class>
  </interceptor-binding>
</assembly-descriptor
>
```

`resources/WEB-INF/weblogic.xml`

- ##########################

```
<?xml version="1.0" encoding="UTF-8"?>

<weblogic-web-app
>
<library-ref>
   <library-name
>jsf</library-name>
   <specification-version
>1.2</specification-version>
   <implementation-version
>1.2</implementation-version>
   <exact-match
>false</exact-match>
 </library-ref>
</weblogic-web-app>
```

- #########library-ref
  #######Weblogic#####################JSF#########################
  weblogic-application.xml
  ##

39.2.2.3.

##############################jboss-seam.jar#######################

build.xml

- ###### weblogic-application.xml ##############

```
<!-- Resources to go in the ear -->
<fileset id="ear.resources" dir="${resources.dir}">
   <include name="META-INF/application.xml" />
   <include name="META-INF/weblogic-application.xml" />
   <include name="META-INF/*-service.xml" />
   <include name="META-INF/*-xmbean.xml" />
   <include name="treecache.xml" />
   <include name="*.jpdl.xml" />
   <exclude name=".gpd.*" />
   <include name="*.cfg.xml" />
   <include name="*.xsd" />
</fileset>
>
```

$SEAM/lib/interop/jboss-seam-wls-compatible.jar

- *#39.2.1. #Weblogic##EJB3####* ######################
- ##jar#################### $SEAM/lib/jboss-seam.jar
  ##############################EAR##############################jboss-seam.jar################
- ###############EAR########## jboss-seam.jar####################################jar###

jboss-seam-wls-compatible.jar########################## jee5/booking###############ant archive#####################

########Weblogic##EAR###############################

```
cp ./dist/jboss-seam-jee5.ear
        $BEA_HOME/user_projects/domains/seam_examples/autodeploy
```

```
```

http://localhost:7001/seam-jee5/#########################

39.3. jpa

####Seam POJO#Hibernate JPA###EJB3####################Weblogic 10.X###

########Weblogic 10.x###Weblogic##############JBoss AS#########

#######Weblogic#JSF######*#39.1.4. # Weblogic#JSF########*################################

39.3.1. jpa

##

39.3.1.1.

#########Weblogic 10.X #######PointBase#####################hsql##############PointBase#################PointBase####### #hibernate#### PointBase#####################jpa/weblogic92######PointBase##################

##########jee5 *#39.2.2.1. #hsql###########*###

- DataSource Name: `seam-jpa-ds`
- JNDI Name: `seam-jpa-ds`

39.3.1.2.

Building it only requires running the correct ant command:

```
ant weblogic10
```

This will create a container specific distribution and exploded archive directories.

39.3.1.3.

#39.1.2. #Weblogic################Weblogic###

```
cp ./dist-weblogic10/jboss-seam-jpa.war
$BEA_HOME/user_projects/domains/seam_examples/autodeploy
```

`http://localhost:7001/jboss-seam-jpa/############################`

39.3.2. Weblogic 10.x####

- Weblogic#10.x#9.2#####################
  - `META-INF/persistence.xml` — 9.2#######`PointBase`##10.x#######`hsql`###########
  - `WEB-INF/weblogic.xml` — #######Weblogic 10.x##########`ANTLR`##############################OC4J ##

```
<?xml version="1.0" encoding="UTF-8"?>
<weblogic-web-app
xmlns="http://www.bea.com/ns/weblogic/90"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.bea.com/ns/weblogic/90
              http://www.bea.com/ns/weblogic/90/weblogic-web-app.xsd">
  <library-ref>
    <library-name
>jsf</library-name>
    <specification-version
>1.2</specification-version>
    <implementation-version
>1.2</implementation-version>
    <exact-match
>false</exact-match>
  </library-ref>
  <container-descriptor>
    <prefer-web-inf-classes
>true</prefer-web-inf-classes>
  </container-descriptor>
</weblogic-web-app
>
```

####Weblogic#####################Web##hibernate##`INF/persistence.xml`##

```
<property name="hibernate.query.factory_class"
    value="org.hibernate.hql.classic.ClassicQueryTranslatorFactory"/>
```

- `WEB-INF/components.xml` — Weblogic 10.x######JPA##################################

```
<transaction:entity-transaction entity-manager="#{em}"/>
```

- `WEB-INF/web.xml` — `jsf-impl.jar`# `WAR`############################

```
<listener>
  <listener-class
>com.sun.faces.config.ConfigureListener</listener-class>
 </listener
>
```

- Weblogic#10.x######JBoss###################################

  - `META-INF/persistence.xml` — ############Weblogic##############

```
<property name="hibernate.transaction.manager_lookup_class"
  value="org.hibernate.transaction.WeblogicTransactionManagerLookup"/>
```

  - `WEB-INF/lib` — Weblogic################################JBoss AS#########################hibernate################

    - JPA#########Hibernate########jar##########

      - `hibernate.jar`
      - `hibernate-annotations.jar`
      - `hibernate-entitymanager.jar`
      - `hibernate-validator.jar`

- jboss-common-core.jar
- commons-logging.jar
- commons-collections.jar
- jboss-common-core.jar

- ##############jar#####Weblogic#######
  - antlr.jar
  - cglib.jar
  - asm.jar
  - dom4j.jar
  - el-ri.jar
  - javassist.jar
  - concurrent.jar

39.4. Weblogic 10.x #seam-gen#########################

seam-gen ###seam-gen#######JBoss AS#############################

seam-gen#######################seam-gen################ Weblogic 10.x##Weblogic 10.x ############jar###############

###########seam-gen WAR###################Seam POJO########Hibernate JPA, Facelets, Drools security, RichFaces, ###################################

39.4.1. seam-gen#p#########

################ ###############seam-gen#############Seam########################./seam setup###

```
./seam setup
Buildfile: build.xml

init:
```

```
setup:
     [echo] Welcome to seam-gen :-)
    [input] Enter your Java project workspace (the directory that contains your
Seam projects) [C:/Projects] [C:/Projects]
/home/jbalunas/workspace
    [input] Enter your JBoss home directory [C:/Program Files/jboss-4.2.3.GA]
[C:/Program Files/jboss-4.2.3.GA]
/jboss/apps/jboss-4.2.3.GA
    [input] Enter the project name [myproject] [myproject]
weblogic-example
     [echo] Accepted project name as: weblogic_example
    [input] Select a RichFaces skin (not applicable if using ICEFaces) [blueSky]
 ([blueSky], classic, ruby, wine, deepMarine, emeraldTown, sakura, DEFAULT)

    [input] Is this project deployed as an EAR (with EJB components) or a WAR
(with no EJB support) [ear]  ([ear], war, )
war
    [input] Enter the Java package name for your session beans [org.jboss.seam.
tutorial.weblogic.action] [org.jboss.seam.tutorial.weblogic.action]
org.jboss.seam.tutorial.weblogic.action
    [input] Enter the Java package name for your entity beans [org.jboss.seam.
tutorial.weblogic.model] [org.jboss.seam.tutorial.weblogic.model]
org.jboss.seam.tutorial.weblogic.model
    [input] Enter the Java package name for your test cases [org.jboss.seam.
tutorial.weblogic.action.test] [org.jboss.seam.tutorial.weblogic.action.test]
org.jboss.seam.tutorial.weblogic.test
    [input] What kind of database are you using? [hsql]  ([hsql], mysql, oracle,
 postgres, mssql, db2, sybase, enterprisedb, h2)

    [input] Enter the Hibernate dialect for your database [org.hibernate.
dialect.HSQLDialect] [org.hibernate.dialect.HSQLDialect]

    [input] Enter the filesystem path to the JDBC driver jar [/tmp/seamlib/hsqldb.jar]
[/tmp/seam/lib/hsqldb.jar]

    [input] Enter JDBC driver class for your database [org.hsqldb.jdbcDriver]
 [org.hsqldb.jdbcDriver]

    [input] Enter the JDBC URL for your database [jdbc:hsqldb:.] [jdbc:hsqldb:.]

    [input] Enter database username [sa] [sa]

    [input] Enter database password [] []
```

```
    [input] Enter the database schema name (it is OK to leave this blank) [] []

    [input] Enter the database catalog name (it is OK to leave this blank) [] []

    [input] Are you working with tables that already exist in the database? [n]
 (y, [n], )

    [input] Do you want to drop and recreate the database tables and data in
import.sql each time you deploy? [n]  (y, [n], )

    [input] Enter your ICEfaces home directory (leave blank to omit ICEfaces) [] []

[propertyfile] Creating new property file:
/rhdev/projects/jboss-seam/cvs-head/jboss-seam/seam-gen/build.properties
     [echo] Installing JDBC driver jar to JBoss server
     [copy] Copying 1 file to /jboss/apps/jboss-4.2.3.GA/server/default/lib
     [echo] Type 'seam create-project' to create the new project

BUILD SUCCESSFUL
```

`./seam new-project#################### cd /home/jbalunas/workspace/weblogic_example######################`

39.4.2. Weblogic 10.X#######

`##########################################################################`

39.4.2.1.

`build.xml`

- `############archive##########`

```
<project name="weblogic_example" default="archive" basedir="."
>
```

`resources/META-INF/persistence-dev.xml`

- `jta-data-source#seam-gen-ds##########Weblogic############################jndi-name##############`
- JPA############################RESOURCE_LOCAL##########

```
<persistence-unit name="weblogic_example" transaction-type="RESOURCE_LOCAL"
>
```

- Weblogic################################

```
<property name="hibernate.cache.provider_class"
   value="org.hibernate.cache.HashtableCacheProvider"/>
<property name="hibernate.transaction.manager_lookup_class"
   value="org.hibernate.transaction.WeblogicTransactionManagerLookup"/>
```

- ##################Weblogic###########`persistence-prod.xml`##############

`resource/WEB-INF/weblogic.xml`
    ########### *description of WEB-INF/weblogic.xml [578]*#################

`resource/WEB-INF/components.xml`
    JPA################Seam#########################

```
<transaction:entity-transaction entity-manager="#{entityManager}"/>
```

    ##

```
xmlns:transaction="http://jboss.com/products/seam/transaction"
```

```
http://jboss.com/products/seam/transaction      http://jboss.com/products/seam/transaction-
2.2.xsd
```

`resource/WEB-INF/web.xml`
    `WEB-INF/web.xml` —— `jsf-impl.jar`# `WAR`############################

```
 <listener>
   <listener-class
>com.sun.faces.config.ConfigureListener</listener-class>
 </listener
```

```
  >
```

`resources/WEB-INF/jboss-web.xml`
JBoss AS#################################`jboss-app.xml`#JBoss AS############################

`resources/*-ds.xml`
JBoss AS###JBoss AS####################Weblogic################

39.4.2.2.

`seam-gen`#############`jpa`################################*#39.3.2. #Weblogic 10.x#####*###

- build.xml — ####`build.xml`#############`war`############################

```
    <copy todir="${war.dir}/WEB-INF/lib">
      <fileset dir="${lib.dir}">
        <!-- Misc 3rd party -->
        <include name="commons-logging.jar" />
        <include name="dom4j.jar" />
        <include name="javassist.jar" />
        <include name="cglib.jar" />
        <include name="antlr.jar" />

        <!-- Hibernate -->
        <include name="hibernate.jar" />
        <include name="hibernate-commons-annotations.jar" />
        <include name="hibernate-annotations.jar" />
        <include name="hibernate-entitymanager.jar" />
        <include name="hibernate-validator.jar" />
        <include name="jboss-common-core.jar" />
        <include name="concurrent.jar" />
      </fileset>
    </copy>
  >
```

39.4.3.

###

39.4.3.1.

##########jee5 *#39.2.2.1. #hsql##########*#####################################

- DataSource Name: `seam-gen-ds`
- JNDI Name: `seam-gen-ds`

39.4.3.2.

##################ant#################

39.4.3.3.

#39.1.2. #Weblogic################Weblogic###

```
cp ./dist/weblogic_example.war /jboss/apps/bea/user_projects/domains/seam_examples/autodeploy
```

`http://localhost:7001/weblogic_example/`############################

Seam on IBM's WebSphere AS v7

40.1. WebSphere AS environment and version recommendation

WebSphere Application Server v7 is IBM's application server offering. This release is fully Java EE 5 certified.

WebSphere AS being a commercial product, we will not discuss the details of its installation. At best, we will instruct you to follow the directions provided by your particular installation type and license.

First, we will go over some basic considerations on how to run Seam applications under WebSphere AS v7. We will go over the details of these steps using the JEE5 booking example. We will also deploy the JPA (non-EJB3) example application.

All of the examples and information in this chapter are based on WebSphere AS v7. A trial version can be downloaded here : *WebSphere Application Server V7* [http://www.ibm.com/developerworks/downloads/ws/was]

WebSphere v7.0.0.5 is the minimal version of WebSphere v7 to use with Seam. WAS v7.0.0.9 is highly recommended. Earlier versions of WebSphere have bugs in the EJB container that will cause various exceptions to occur at runtime.

##

You may encounter two exceptions with Seam on WebSphere v7.0.0.5 :

`EJBContext may only be looked up by or injected into an EJB`

This is a bug in WebSphere v7.0.0.5. WebSphere does not conform to the EJB 3.0 specs as it does not allow to perform a lookup on "java:comp/EJBContext" in callback methods.

This problem is associated with APAR PK98746 at IBM and is corrected in v7.0.0.9.

`NameNotFoundException: Name "comp/UserTransaction" not found in context "java:"`

Another bug in WebSphere v7.0.0.5. This occurs when an HTTP session expires. Seam correctly catches the exception when necessary and performs the correct actions in these cases. The problem is that even if the exception is handled by Seam, WebSphere prints the traceback of the exception in SystemOut. Those messages are harmless and can safely be ignored.

> This problem is associated with APAR PK97995 at IBM and is corrected in v7.0.0.9.

The following sections in this chapter assume that WebSphere is correctly installed and is functional, and a WebSphere "profile" has been successfully created.

This chapter explains how to compile, deploy and run some sample applications in WebSphere. These sample applications require a database. WebSphere comes by default with a set of sample applications called "Default Application". This set of sample applications use a Derby database running on the Derby instance installed within WebSphere. In order to keep this simple we'll use this Derby database created for the "Default Applications". However, to run the sample application with the Derby database "as-is", a patched Hibernate dialect must be used (The patch changes the default "auto" key generation strategy) as explained in *# 41. GlassFish ############## Seam*. If you want to use another database, it's just a matter of creating a connection pool in WebSphere pointing to this database, declare the correct Hibernate dialect and set the correct JNDI name in `persistence.xml`.

40.2. Configuring the WebSphere Web Container

This step is mandatory in order to have Seam applications run with WebSphere v7. Two extra properties must be added to the Web Container. Please refer to the IBM WebSphere Information Center for further explanations on those properties.

To add the extra properties:

- Open the WebSphere administration console
- Select the `Servers/Server Types/WebSphere Application Servers` in the left navigation menu
- Click on the server name (`server1`)
- On the right navigation menu, select `Web Container Settings/Web container`
- On the right navigation menu, select `custom properties` and add the following properties:
  - `prependSlashToResource = true`
  - `com.ibm.ws.webcontainer.invokefilterscompatibility = true`
- Save the configuration and restart the server

40.3. Seam and the WebSphere JNDI name space

In order to use component injection, Seam needs to know how to lookup for session beans bound to the JNDI name space. Seam provides two mechanisms to configure the way it will search for such resources:

- The global `jndi-pattern` switch on the `<core:init>`tag in `components.xml`. The switch can use a special placeholder "`#{ejbName}`" that resolves to the unqualified name of the EJB
- The `@JndiName` annotation

#30.1.5. #EJB ##### Seam #### gives detailed explanations on how those mechanisms work.

By default, WebSphere will bind session beans in its local JNDI name space under a "short" binding name that adheres to the following pattern `ejblocal:<package.qualified.local.interface.name>`.

For a detailed description on how WebSphere v7 organizes and binds EJBs in its JNDI name spaces, please refer to the WebSphere Information Center.

As explained before, Seam needs to lookup for session bean as they appear in JNDI. Basically, there are three strategies, in order of complexity:

- Specify which JNDI name Seam must use for each session bean using the `@JndiName` annotation in the java source file,
- Override the default session bean names generated by WebSphere to conform to the `jndi-pattern` attribute,
- Use EJB references.

40.3.1. Strategy 1: Specify which JNDI name Seam must use for each Session Bean

This strategy is the simplest and fastest one regarding development. It uses the WebSphere v7 default binding mechanism. To use this strategy:

- Add a `@JndiName("ejblocal:<package.qualified.local.interface.name>)` annotation to each session bean that is a Seam component.
- In `components.xml`, add the following line:

```
<core:init jndi-name="java:comp/env/#{ejbName}" />
```

- Add a file named `WEB-INF/classes/seam-jndi.properties` in the web module with the following content:

```
com.ibm.websphere.naming.hostname.normalizer=com.ibm.ws.naming.util.DefaultHostnameNormalizer
java.naming.factory.initial=com.ibm.websphere.naming.WsnInitialContextFactory
com.ibm.websphere.naming.name.syntax=jndi
com.ibm.websphere.naming.namespace.connection=lazy
com.ibm.ws.naming.ldap.ldapinitctxfactory=com.sun.jndi.ldap.LdapCtxFactory
```

```
com.ibm.websphere.naming.jndicache.cacheobject=populated
com.ibm.websphere.naming.namespaceroot=defaultroot
com.ibm.ws.naming.wsn.factory.initial=com.ibm.ws.naming.util.WsnInitCtxFactory
com.ibm.websphere.naming.jndicache.maxcachelife=0
com.ibm.websphere.naming.jndicache.maxentrylife=0
com.ibm.websphere.naming.jndicache.cachename=providerURL
java.naming.provider.url=corbaloc:rir:/NameServiceServerRoot
java.naming.factory.url.pkgs=com.ibm.ws.runtime:com.ibm.ws.naming
```

- At the end of `web.xml`, add the following lines:

```
<ejb-local-ref>
  <ejb-ref-name>EjbSynchronizations</ejb-ref-name>
  <ejb-ref-type>Session</ejb-ref-type>
  <local-home></local-home>
  <local>org.jboss.seam.transaction.LocalEjbSynchronizations</local>
 </ejb-local-ref>
```

That's all folks! No need to update any file during the development, nor to define any EJB to EJB or web to EJB reference!

Compared to the other strategies, this strategy has the advantage to not have to manage any EJBs reference and also to not have to maintain extra files. The only drawback is one extra line in the java source code with the `@JndiName` annotation

40.3.2. Strategy 2: Override the default names generated by WebSphere

There is no simple way to globally override the default naming strategy for session beans in WebSphere. However, WebSphere provides a way to override the name of each bean.

To use this strategy:

- Add a file named `META-INF/ibm-ejb-jar-bnd.xml` in the EJB module and add an entry for each session bean like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar-bnd
  xmlns="http://websphere.ibm.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://websphere.ibm.com/xml/ns/javaee
                      http://websphere.ibm.com/xml/ns/javaee/ibm-ejb-jar-bnd_1_0.xsd"
  version="1.0">
```

```
 <session name="AuthenticatorAction" simple-binding-name="AuthenticatorAction" />
 <session name="BookingListAction" simple-binding-name="BookingListAction" />

</ejb-jar-bnd
>
```

WebSphere will then bind the `AuthenticatorAction` EJB to the `ejblocal:AuthenticatorAction` JNDI name

- In `components.xml`, add the following line:

```
<core:init jndi-name="ejblocal:#{ejbName}" />
```

- Add a file named `WEB-INF/classes/seam-jndi.properties` as described in strategy 1
- In `web.xml`, add the following lines (Note the different `ejb-ref-name` value):

```
<ejb-local-ref>
  <ejb-ref-name>ejblocal:EjbSynchronizations</ejb-ref-name>
  <ejb-ref-type>Session</ejb-ref-type>
  <local-home></local-home>
  <local>org.jboss.seam.transaction.LocalEjbSynchronizations</local>
 </ejb-local-ref>
```

Compared to the first strategy, this strategy requires to maintain an extra file (`META-INF/ibm-ejb-jar-ext.xml`), where a line must be added each time a new session bean is added to the application), but still does not require to maintain EJB reference between beans.

40.3.3. Strategy 3: Use EJB references

This strategy is based on the usage of EJB references, from EJB to EJB and from the web module to EJB. To use it:

- In `components.xml`, add the following line:

```
<core:init jndi-name="java:comp/env/#{ejbName}" />
```

- Follow the instructions in *#30.1.5. #EJB ##### Seam ####* to declare the references from web to EJB and from EJB to EJB

This is the most tedious strategy as each session bean referenced by another session bean (i.e. "injected") as to be declared in `ejb-jar.xml` file. Also, each new session bean has to be added to the list of referenced bean in `web.xml`

40.4. Configuring timeouts for Stateful Session Beans

A timeout value has to be set for each stateful session bean used in the application because stateful bean must not expire in WebSphere while Seam might still need them. At the time of writing this document, WebSphere does not provide a way to configure a global timeout at neither the cluster, server, application nor ejb-jar level. It has to be done for each stateful bean individually. By default, the default timeout is 10 minutes. This is done by adding a file named `META-INF/ibm-ejb-jar-ext.xml` in the EJB module, and declare the timeout value for each bean:

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar-ext
  xmlns="http://websphere.ibm.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://websphere.ibm.com/xml/ns/javaee
                      http://websphere.ibm.com/xml/ns/javaee/ibm-ejb-jar-ext_1_0.xsd"
  version="1.0">

  <session name="BookingListAction"><time-out value="605"/></session>
  <session name="ChangePasswordAction"><time-out value="605"/></session>

</ejb-jar-ext>
```

The `time-out` is expressed in seconds and must be higher than the Seam conversation expiration timeout and a few minutes higher than the user's HTTP session timeout (The session expiration timeout can trigger a few minutes after the number of minutes declared to expire the HTTP session).

40.5. `jee5/booking`

The`jee5/booking`example is based on the Hotel Booking example (which runs on JBoss AS). Out of the box, it is designed to run on Glassfish, but with the following steps, it can be deployed on WebSphere. It is located in the `$SEAM_DIST/examples/jee5/booking` directory.

The example already has a breakout of configurations and build scripts for WebSphere. First thing, we are going to do is build and deploy this example. Then we'll go over some key changes that we needed.

The tailored configuration files for WebSphere use the second JNDI mapping strategy ("Override the default names generated by WebSphere") as the goal was to not change any java code to add the `@JndiName` annotation as in the first strategy.

40.5.1. `jee5/booking`

Building it only requires running the correct ant command:

```
ant -f build-websphere7.xml
```

This will create container specific distribution and exploded archive directories with the `websphere7` label.

40.5.2. Deploying the `jee5/booking` example

The steps below are for the WAS version stated above.The ports are the default values, if you changed them, you must substitute the values.

1. Log in to the administration console

```
http://localhost:9060/admin
```

   Enter your userid annd/or your password if security is enabled for the console.

2. Go to the `WebSphere enterprise applications` menu option under the `Applications --> Application Type` left side menu.
3. At the top of the `Enterprise Applications` table select `Install`. Below are installation wizard pages and what needs to done on each:
   - `##################`
     - Browse to the `examples/jee5/booking/dist-websphere7/jboss-seam-jee5.ear` file using the file upload widget.
     - `Next` `##############`
     - Select the `Fast Path` button.
     - `Next` `##############`
   - `###############`
     - Select the "`Allow EJB reference targets to resolve automatically`" check boxes at the bottom of the page. This will let WebSphere use its simplified JNDI reference mapping.
     - `Next` `##############`
   - `##############`
     - No changes needed here as we only have one server. Select the `Next`button.
   - `Map virtual hosts for Web modules`
     - No changes needed here as we only have one virtual host. Select the `Next`button.

- Summary (##)
  - ################Finish #############
- Installation (######)
  - Now you will see WebSphere installing and deploying your application.
  - When done, select the Save link and you will be returned to the Enterprise Applications table.
- To start the application, select the application in the list, then click on the Start button at the top of the table.

4. You can now access the application at http://localhost:9080/seam-jee5-booking

40.5.3. Deviation from the original base files

Below are the differences between the base configuration files and the WebSphere specific files held in the resources-websphere7 directory.

- META-INF/ejb-jar.xml — Removed all the EJB references
- META-INF/ibm-ejb-jar-bnd.xml — This WebSphere specific file has been added as we use the second JNDI mapping strategy. It defines, for each session bean, the name WebSphere will use to bind it in its JNDI name space
- META-INF/ibm-ejb-jar-ext.xml — This WebSphere specific file defines the timeout value for each stateful bean
- META-INF/persistence.xml — The main changes here are for the datasource JNDI path, switching to the WebSphere transaction manager lookup class, turning off the hibernate.transaction.flush_before_completion toggle, and forcing the Hibernate dialect to be GlassfishDerbyDialect as we are using the integrated Derby database
- WEB-INF/components.xml — the change here is jndi-pattern to use ejblocal:#{ejbname} as using the second JNDI matching strategy
- WEB-INF/web.xml — Remove all the ejb-local ref except the one for EjbSynchronizations bean. Changed the ref fo this bean to ejblocal:EjbSynchronizations
- import.sql — due to the cutomized hibernate Derby dialect, the ID column can not be populated by this file and was removed.

Also the build procedure has been changed to include the log4j.jar file and exclude the concurrent.jar and jboss-common-core.jar files.

40.6. jpa booking

This is the Hotel Booking example implemented in Seam POJOs and using Hibernate JPA with JPA transactions. It does not use EJB3.

The example already has a breakout of configurations and build scripts for many of the common containers including WebSphere.

First thing, we are going to do is build and deploy that example. Then we'll go over some key changes that we needed.

40.6.1. jpa

Building it only requires running the correct ant command:

```
ant websphere7
```

This will create container specific distribution and exploded archive directories with the `websphere7` label.

40.6.2. jpa

Deploying `jpa` application is very similar to the `jee5/booking` example at *#40.5.2. #Deploying the `jee5/booking` example#*. The main difference is, that this time, we will deploy a war file instead of an ear file, and we'll have to manually specify the context root of the application.

Follow the same instructions as for the `jee5/booking` sample. Select the `examples/jpa/dist-websphere7/jboss-seam-jpa.war` file on the first page and on the `Map context roots for Web modules` page (after the `Map virtual host for Web module`), enter the context root you want to use for your application in the `Context Root` input field.

When started, you can now access the application at the `http://localhost:9080/<context root>`.

40.6.3. Deviation from the generic base files

Below are the configuration file differences between the base configuration files and the files customized for WebSphere held in the `resources-websphere7` directory.

- `META-INF/persistence.xml` — The main changes here are for the datasource JNDI path, switching to the WebSphere transaction manager look up class, turning off the `hibernate.transaction.flush_before_completion` toggle, and forcing the Hibernate dialect to be `GlassfishDerbyDialect` how as using the integrated Derby database
- `import.sql` — due to the customized hibernate Derby dialect, the `ID` column can not be populated by this file and was removed.

Also the build procedure have been changed to include the `log4j.jar` file and exclude the `concurrent.jar` and `jboss-common-core.jar` files.

GlassFish ############### Seam

GlassFish ##Java EE 5 ## v2 UR2 ###

GlassFish ################ jee5 ############################jpa ############################seam-gen ######################### GlassFish

41.1. GlassFish

41.1.1.

#################### GlassFish #################

- *GlassFish v2 UR2 - #########* [https://glassfish.dev.java.net/downloads/v2ur2-b04.html]

GlassFish ########################

```
$ java -Xmx256m -jar glassfish-installer-v2ur2-b04-linux.jar
```

############GlassFish ###############

```
$ cd glassfish; ant -f setup.xml
```

##############`domain1` ###

######## JavaDB ###########

```
$ bin/asadmin start-database
```

##

JavaDB ##HSQLDB # JBoss AS ##############GlassFish ###############

####GlassFish #############

```
$ bin/asadmin start-domain domain1
```

The web administration console is available at `http://localhost:4848/`. You can access the web admin console with the default username (`admin`) and password (`adminadmin`). We will be using the the admin console to deploy our examples. You can also copy EAR/WAR files to the `glassfish/domains/domain1/autodeploy` directory to deploy them, although we are not going to cover that.

################################

```
$ bin/asadmin stop-domain domain1; bin/asadmin stop-database
```

41.2. `jee5/booking`

`jee5/booking` ######(JBoss AS ######) ######################## GlassFish ########################## `$SEAM_DIST/examples/jee5/booking` ######

41.2.1. Building the `jee5/booking` example

To build the example, simply execute the default `ant` target:

```
$ ant
```

in the `examples/jee5/booking` directory. This will create the `dist` and `exploded-archives` directories.

41.2.2. GlassFish

GlassFish ############ GlassFish ##################

1. `http://localhost:4848` ##################
2. ########## Applications (########) ################ Enterprise Applications (################) ############
3. Enterprise Application (################) ######### Deploy (##) ##
   - ##################
     - ##### `examples/jee5/booking/dist/jboss-seam-jee5.ear` ##########
     - OK ##############
4. `http://localhost:8081/seam-jee5/` ####################

41.3. jpa booking

####Hibernate JPA # JPA ############ Seam POJO ##EJB3 ############

#######GlassFish ####################################

41.3.1. jpa

###############glassfish ############

```
$ ant glassfish
```

############## dist-glassfish ####### exploded-archives-glassfish ##############

41.3.2. jpa

This is very similar to the jee5 example at *#41.2.2. #GlassFish ################* except that this is a war and not an ear.

- ####################

```
http://localhost:4848
```

- ########## Applications (########) ################ Web Applications (Web ########) ############
  - #################
    - ##### examples/jpa/dist-glassfish/jboss-seam-jpa.war ##########
    - OK ##############
  - http://localhost:8081/jboss-seam-jpa/ ####################

Hypersonic SQL DB ##### Derby ###

In order for the app to work out of the box with GlassFish, we have used the Derby (aka JavaDB) database in GlassFish. However, we strongly recommend that you use another database (e.g. HSQL). examples/jpa/resources-glassfish/WEB-INF/classes/GlassfishDerbyDialect.class is a hack to get around a Derby bug in GlassFish server. You must use it as your Hibernate dialect if you use Derby with GlassFish.

41.3.3. GlassFish v2 UR2

- #########
  - `META-INF/persistence.xml` — the main changes needed are the datasource JNDI, switching to the GlassFish transaction manager lookup class, and changing the hibernate dialect to be `GlassfishDerbyDialect`.
  - `WEB-INF/classes/GlassfishDerbyDialect.class` — this class is needed for the Hibernate dialect change to `GlassfishDerbyDialect`
  - `import.sql` — ######## Derby DB ####### `ID` ##############################

41.4. `seam-gen` ################## GlassFish v2 UR2

`seam-gen` is a very useful tool for developers to quickly get an application up and running, and provides a foundation to add your own functionality. Out of box `seam-gen` will produce applications configured to run on JBoss AS. These instructions will show the steps needed to get it to run on GlassFish.

41.4.1. `seam-gen`

############################ `seam-gen`
Hibernate
##############################

```
$ ./seam setup
Buildfile: build.xml

init:

setup:
[echo] Welcome to seam-gen :-)
[input] Enter your Java project workspace (the directory that contains your
Seam projects) [C:/Projects] [C:/Projects]
/projects
[input] Enter your JBoss home directory [C:/Program Files/jboss-4.2.3.GA]
[C:/Program Files/jboss-4.2.3.GA]

[input] Enter the project name [myproject] [myproject]
seamgen_example
[echo] Accepted project name as: seamgen_example
[input] Do you want to use ICEfaces instead of RichFaces [n] (y, [n])
```

```
[input] skipping input as property icefaces.home.new has already
been set.
[input] Select a RichFaces skin [blueSky] ([blueSky], classic, ruby, wine,
deepMarine, emeraldTown, japanCherry, DEFAULT)

[input] Is this project deployed as an EAR (with EJB components) or a WAR
(with no EJB support) [ear] ([ear], war)

[input] Enter the Java package name for your session beans
[com.mydomain.seamgen_example] [com.mydomain.seamgen_example]
org.jboss.seam.tutorial.glassfish.action
[input] Enter the Java package name for your entity beans
[org.jboss.seam.tutorial.glassfish.action]
[org.jboss.seam.tutorial.glassfish.action]
org.jboss.seam.tutorial.glassfish.model
[input] Enter the Java package name for your test cases
[org.jboss.seam.tutorial.glassfish.action.test]
[org.jboss.seam.tutorial.glassfish.action.test]
org.jboss.seam.tutorial.glassfish.test
[input] What kind of database are you using? [hsql] ([hsql], mysql, oracle,
postgres, mssql, db2, sybase, enterprisedb, h2)

[input] Enter the Hibernate dialect for your database
[org.hibernate.dialect.HSQLDialect]
[org.hibernate.dialect.HSQLDialect]

[input] Enter the filesystem path to the JDBC driver jar
[/tmp/seam/lib/hsqldb.jar] [/tmp/seam/lib/hsqldb.jar]

[input] Enter JDBC driver class for your database [org.hsqldb.jdbcDriver]
[org.hsqldb.jdbcDriver]

[input] Enter the JDBC URL for your database [jdbc:hsqldb:.]
[jdbc:hsqldb:.]

[input] Enter database username [sa] [sa]

[input] Enter database password [] []

[input] Enter the database schema name (it is OK to leave this blank) [] []

[input] Enter the database catalog name (it is OK to leave this
blank) [] []
```

```
[input] Are you working with tables that already exist in the database? [n]
(y, [n])

[input] Do you want to drop and recreate the database tables and data in
import.sql each time you deploy? [n] (y, [n])

[propertyfile] Creating new property file:
/home/mnovotny/workspaces/jboss/jboss-seam/seam-gen/build.properties
[echo] Installing JDBC driver jar to JBoss server
[copy] Copying 1 file to
/home/mnovotny/workspaces/jboss/jboss-seam/seam-gen/C:/Program
Files/jboss-4.2.3.GA/server/default/lib
[echo] Type 'seam create-project' to create the new project

BUILD SUCCESSFUL
Total time: 4 minutes 5 seconds
```

`#################$ ./seam new-project ############## cd /projects/seamgen_example ##############################`

41.4.2. GlassFish

`##########################`

41.4.2.1.

`resources/META-INF/persistence-dev.xml`

- `jta-data-source` # `jdbc/__default` ################ GlassFish Derby DB #######
- Replace all of the properties with the following. The key differences are briefly described in *#41.3.3. #GlassFish v2 UR2 ######*:

```
<property name="hibernate.dialect" value="GlassfishDerbyDialect"/>
<property name="hibernate.hbm2ddl.auto" value="update"/>
<property name="hibernate.show_sql" value="true"/>
<property name="hibernate.format_sql" value="true"/>
<property                                    name="hibernate.cache.provider_class"
 value="org.hibernate.cache.HashtableCacheProvider"/>
<property                                name="hibernate.transaction.manager_lookup_class"
 value="org.hibernate.transaction.SunONETransactionManagerLookup"/>
```

- prod ############ GlassFish ############`persistence-prod.xml` ################

`resources/GlassfishDerbyDialect.class`
: ##jpa ###### seamgen_example/resources ################

```
$ cp \
$SEAM_DIST/examples/jpa/resources-glassfish/WEB-INF/classes/
GlassfishDerbyDialect.class \
./resources
```

`resources/META-INF/jboss-app.xml`
: JBoss AS ######################## (JBoss AS ## `jboss-app.xml` ########################)

`resources/*-ds.xml`
: JBoss AS ######################## (##########JBoss AS ##################GlassFish ######################)

`resources/WEB-INF/components.xml`

- ###################### - `<transaction:ejb-transaction/>` ################ `xmlns:transaction="http://jboss.com/products/seam/transaction"` #########
- Alter the `jndi-pattern` to `java:comp/env/seamgen_example/#{ejbName}`

`resources/WEB-INF/web.xml`
: As with the `jee5/booking` example, we need to add EJB references to web.xml. Technically, the reference type is not required, but we add it here for good measure. Note that these references require the presence of an empty `local-home` element to retain compatibility with a JBoss AS 4.x deployment.

```
<ejb-local-ref>
   <ejb-ref-name>seamgen_example/AuthenticatorAction</ejb-ref-name>
   <ejb-ref-type>Session</ejb-ref-type>
   <local-home/>
   <local>org.jboss.seam.tutorial.glassfish.action.Authenticator</local>
 </ejb-local-ref>

 <ejb-local-ref>
   <ejb-ref-name>seamgen_example/EjbSynchronizations</ejb-ref-name>
   <ejb-ref-type>Session</ejb-ref-type>
   <local-home/>
   <local>org.jboss.seam.transaction.LocalEjbSynchronizations</local>
 </ejb-local-ref>
```

Keep in mind that if you are deploying to JBoss AS 4.x, and have defined the EJB references shown above in your web.xml, you will need to also define local JNDI names for each of them in jboss-web.xml, as shown below. This step is not required when deploying to GlassFish, but it's mentioned here in case you are also deploying the application to JBoss AS 4.x (not required for JBoss AS 5).

```
<ejb-local-ref>
   <ejb-ref-name>seamgen_example/AuthenticatorAction</ejb-ref-name>
   <local-jndi-name>AuthenticatorAction</local-jndi-name>
 </ejb-local-ref>

 <ejb-local-ref>
   <ejb-ref-name>seamgen_example/EjbSynchronizations</ejb-ref-name>
   <local-jndi-name>EjbSynchronizations</local-jndi-name>
 </ejb-local-ref>
```

41.4.2.2. `AuthenticatorAction` EJB

`Authenticator` Seam POJO ##############EJB3

1. • ##### `AuthenticatorAction` #######

   • `@Stateless` ########### `AuthenticatorAction` ##########

   • `AuthenticatorAction` ##### `Authenticator` ###################### (EJB3 ####### Bean ###############)######### `@Local` ############`AuthenticatorAction` # `authenticate` ##########################

```
@Name("authenticator")
@Stateless
public class AuthenticatorAction implements Authenticator {
```

```
@Local
public interface Authenticator {

   public boolean authenticate();
}
```

2. ### `web.xml` ###### EJB #################

41.4.2.3. `build.xml` ## jar

############`jee5/booking` ###################

- ############ `archive` ###### (GlassFish ################)#

```
<project name="seamgen_example" default="archive" basedir=".">
```

- We need to get the `GlassfishDerbyDialect.class` into our application jar. To do that find the `jar` task and add the `GlassfishDerbyDialect.class` line as shown below:

```
<target name="jar" depends="compile,copyclasses" description="Build the distribution .jar
 file">
  <copy todir="${jar.dir}">
    <fileset dir="${basedir}/resources">
      <include name="seam.properties" />
      <include name="*.drl" />
      <include name="GlassfishDerbyDialect.class" />
    </fileset>
  </copy>
  ...
```

- Now we need to get extra jars into the `ear` file. Look for the `<copy todir="${ear.dir}/lib" >` section of the `ear` target. Add the following to the child `<fileset dir="${lib.dir}" >` element.

- • Hibernate ###############

```
<!-- Hibernate and deps -->
<include name="hibernate.jar"/>
<include name="hibernate-commons-annotations.jar"/>
<include name="hibernate-annotations.jar"/>
<include name="hibernate-entitymanager.jar"/>
<include name="hibernate-validator.jar"/>
<include name="jboss-common-core.jar"/>
```

- ######################

```
<!-- 3rd party and supporting jars -->
<include name="javassist.jar"/>
<include name="dom4j.jar"/>
<include name="concurrent.jar" />
```

```
  <include name="cglib.jar"/>
  <include name="asm.jar"/>
  <include name="antlr.jar" />
  <include name="commons-logging.jar" />
  <include name="commons-collections.jar" />
```

################

```
<fileset dir="${lib.dir}">
  <includesfile name="deployed-jars-ear.list" />

  <!-- Hibernate and deps -->
  <include name="hibernate.jar"/>
  <include name="hibernate-commons-annotations.jar"/>
  <include name="hibernate-annotations.jar"/>
  <include name="hibernate-entitymanager.jar"/>
  <include name="hibernate-validator.jar"/>
  <include name="jboss-common-core.jar" />

  <!-- 3rd party and supporting jars -->
  <include name="javassist.jar" />
  <include name="dom4j.jar" />
  <include name="concurrent.jar" />
  <include name="cglib.jar" />
  <include name="asm.jar" />
  <include name="antlr.jar" />
  <include name="commons-logging.jar" />
  <include name="commons-collections.jar" />
</fileset>
```

41.4.2.4. seam-gen ################ GlassFish

- ################(### /projects/seamgen-example)# ant ## dist/seamgen-example.ear ###
- ################## *#41.2.2. #GlassFish ################* ########### ####jboss-seam-jee5 ############## seamgen-example ################
- http://localhost:8081/seamgen_example/ ##########################

###

42.1. JDK

Seam does not work with JDK 1.4 and requires JDK 5 or above as it uses annotations and other JDK 5.0 features. Seam has been thoroughly tested using Sun's JDKs. However there are no known issues specific to Seam with other JDKs.

42.1.1. Sun # JDK 6

Sun # JDK 6 ######### JAXB ################### "endorsed" ######################### Sun # JDK 6 Update 4 ##### JAXB 2.1 ############################ #### #### ##

Seam ################## JBoss Embedded ########## JDK 6 ############################ JBoss Embedded # JDK 6 ######## ## JVM #################

```
-Dsun.lang.ClassLoader.allowArraySyntax=true
```

Seam ################ Seam ############################## #### #### JBoss Embedded ##########################

42.2.

This section both lists the compile-time and runtime dependencies for Seam. Where the type is listed as `ear`, the library should be included in the /lib directory of your application's ear file. Where the type is listed as `war`, the library should be placed in the `/WEB-INF/lib` directory of your application's war file. The scope of the dependency is either all, runtime or provided (by JBoss AS 4.2 or 5.0).

`/dependency-report.txt` ########## ### `/build` ###### Maven POM ######## `ant dependencyReport`

42.2.1. Core

 # 42.1.

##	##	###	##
`jboss-seam.jar`	all	ear	### Seam ####### ##########
`jboss-seam-debug.jar`	runtime	war	Seam #########################
`jboss-seam-ioc.jar`	runtime	war	Spring # Seam ################

##	##	###	##
jboss-seam-pdf.jar	runtime	war	Seam # PDF ##################
jboss-seam-excel.jar	runtime	war	Seam # Microsoft® Excel® ##################
jboss-seam-rss.jar	runtime	war	Seam # RSS ##################
jboss-seam-remoting.jar	runtime	war	Seam Remoting ################
jboss-seam-ui.jar	runtime	war	Seam JSF ################
jsf-api.jar	provided		JSF API
jsf-impl.jar	provided		JSF ########
jsf-facelets.jar	runtime	war	Facelets ###
urlrewrite.jar	runtime	war	URL Rewrite ########
quartz.jar	runtime	ear	Seam ####### Quartz ################

42.2.2. RichFaces

42.2. RichFaces

##	##	###	##
richfaces-api.jar	all	ear	###RichFaces################ API ##########
richfaces-impl.jar	runtime	war	RichFaces ###########
richfaces-ui.jar	runtime	war	RichFaces ########### all # UI ##############

42.2.3. Seam Mail

42.3. Seam Mail

##	##	###	##
activation.jar	runtime	ear	################
mail.jar	runtime	ear	##################
mail-ra.jar	compile		################## mail-ra.rar ##################################

##	##	###	##
jboss-seam-mail.jar	runtime	war	Seam Mail ##############

42.2.4. Seam PDF

42.4. Seam PDF

##	###	##	##
itext.jar	runtime	war	PDF #####
jfreechart.jar	runtime	war	#########
jcommon.jar	runtime	war	JFreeChart #########
jboss-seam-pdf.jar	runtime	war	Seam PDF ##############

42.2.5. Seam Microsoft® Excel®

42.5. Seam Microsoft® Excel®

##	###	##	##
jxl.jar	runtime	war	JExcelAPI ########
jboss-seam-excel.jar	runtime	war	Seam Microsoft® Excel® ##############

42.2.6. Seam RSS

42.6. Seam RSS

##	###	##	##
yarfraw.jar	runtime	war	YARFRAW RSS ########
JAXB	runtime	war	JAXB XML ##########
http-client.jar	runtime	war	Apache HTTP Client #########
commons-io	runtime	war	Apache commons IO ########
commons-lang	runtime	war	Apache commons ##########
commons-codec	runtime	war	Apache commons codec ########
commons-collections	runtime	war	Apache commons collections ########
jboss-seam-rss.jar	runtime	war	Seam RSS ###########

42.2.7. JBoss Rules

JBoss Rules ####### Seam # drools/lib ############

42.7. JBoss Rules

##	##	###	##
antlr-runtime.jar	runtime	ear	ANTLR #######
core.jar	runtime	ear	Eclipse JDT #####
drools-api.jar	runtime	ear	
drools-compiler.jar	runtime	ear	
drools-core.jar	runtime	ear	
drools-decisiontables.jar	runtime	ear	
drools-templates.jar	runtime	ear	
janino.jar	runtime	ear	
mvel2.jar	runtime	ear	

42.2.8. JBPM

42.8. JBPM

##	##	###	##
jbpm-jpdl.jar	runtime	ear	

42.2.9. GWT

########## Seam ######### Google Web Toolkit (GWT) ##################

42.9. GWT

##	##	###	##
gwt-servlet.jar	runtime	war	GWT Servlet ########

42.2.10. Spring

########## Seam ######### Spring Framework ##################

42.10. Spring Framework

##	##	###	##
spring.jar	runtime	ear	Spring Framework ########

42.2.11. Groovy

########## Seam ######### Groovy ##################

42.11. Groovy

##	##	###	##
`groovy-all.jar`	runtime	ear	Groovy #########

42.3. Maven

Maven ########################## Seam ########################## Maven Ant Tasks #### Maven # Ant ########## Maven ################################

Maven ########################## POM

Released versions of Seam are available in *http://repository.jboss.org/maven2* [http://repository.jboss.org/maven2] and nightly snapshots are available in *http://snapshots.jboss.org/maven2* [http://snapshots.jboss.org/maven2].

Seam ########## Maven ######

```
<dependency>
  <groupId
>org.jboss.seam</groupId>
  <artifactId
>jboss-seam</artifactId>
</dependency
>
```

```
<dependency>
  <groupId
>org.jboss.seam</groupId>
  <artifactId
>jboss-seam-ui</artifactId>
</dependency
>
```

```
<dependency>
  <groupId
>org.jboss.seam</groupId>
  <artifactId
>jboss-seam-pdf</artifactId>
</dependency
>
```

```
<dependency>
  <groupId
>org.jboss.seam</groupId>
  <artifactId
>jboss-seam-remoting</artifactId>
</dependency
>
```

```
<dependency>
  <groupId
>org.jboss.seam</groupId>
  <artifactId
>jboss-seam-ioc</artifactId>
</dependency
>
```

```
<dependency>
  <groupId
>org.jboss.seam</groupId>
  <artifactId
>jboss-seam-ioc</artifactId>
</dependency
>
```

This sample POM will give you Seam, JPA (provided by Hibernate), Hibernate Validator and Hibernate Search:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0  http://maven.apache.org/maven-
v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.jboss.seam.example/groupId>
  <artifactId>my-project</artifactId>
  <version>1.0</version>
  <name>My Seam Project</name>
  <packaging>jar</packaging>
  <repositories>
```

```
  <repository>
    <id>repository.jboss.org</id>
    <name>JBoss Repository</name>
    <url>http://repository.jboss.org/maven2</url>
  </repository>
 </repositories>

 <dependencies>

  <dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-validator</artifactId>
    <version>3.1.0.GA</version>
  </dependency>

  <dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-annotations</artifactId>
    <version>3.4.0.GA</version>
  </dependency>

  <dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-entitymanager</artifactId>
    <version>3.4.0.GA</version>
  </dependency>

  <dependency>
      <groupId>org.hibernate</groupId>
      <artifactId>hibernate-search</artifactId>
      <version>3.1.1.GA</version>
  </dependency>

  <dependency>
    <groupId>org.jboss.seam</groupId>
    <artifactId>jboss-seam</artifactId>
    <version>2.2.0.GA</version>
  </dependency>

 </dependencies>

</project>
```