

JBoss Seam Framework

[Seam](#) es un proyecto desarrollado por JBoss, cuyo líder es Gavin King. Es un completo framework para la creación de aplicaciones web 2.0 que unifica varias tecnologías como AJAX, Enterprise Java Beans (EJB3), Java Server Faces (JSF), Java Portlets and Business Process Management (BPM), Hibernate, y mucho más.

De la Wikipedia:

Seam introduce el concepto de contextos. Cada componente de Seam existe dentro de un contexto. El contexto conversacional por ejemplo captura todas las acciones del usuario hasta que éste sale del sistema o cierra el navegador - inclusive puede llevar un control de múltiples pestañas y mantiene un comportamiento consistente cuando se usa el botón de regresar de el navegador.

Tú puedes automáticamente generar una aplicación web de altas, bajas, cambio y modificaciones a partir de una base de datos existente utilizando una herramienta de línea de comandos llamada seam-gen incluida con el framework.

El desarrollo WYSIWYG es facilitado a través del uso de las JBoss Tools, que es un conjunto de plug-ins diseñados para el entorno integrado de desarrollo Eclipse. Seam puede ser integrado con las bibliotecas de componentes JSF JBoss RichFaces o con ICEsoft ICEFaces. Ambas bibliotecas poseen soporte para AJAX.

Actualmente soporta varios contenedores de aplicaciones como JBoss 4 o 5, IBM Websphere, BEA WebLogic, Oracle OC4J y por supuesto Apache Tomcat.

Puedes encontrar ejemplos de proyectos creados con seam y listos para ejecutar dentro del directorio examples de la aplicación.

Referencias y documentación:

- [Página oficial de Seam Framework](#)
- [Página del producto JBoss Seam](#)
- [Documentación Online mantenida por la comunidad](#)
- [Documentación de JBoss Seam](#)
- [Introduction to JBoss Seam \[artículo en InfoQ\]](#)

Laboratorio de JBoss Seam Framework



Con el siguiente laboratorio se pretende dar un punto de partida a la creación de aplicaciones usando [Seam](#). El laboratorio está enfocado al desarrollo sobre Eclipse, que es el IDE que recomiendo para trabajar con Seam.

Este laboratorio está enfocado al uso de Seam para JBoss, aunque con pocas o sin modificaciones puede funcionar en Apache Tomcat. Recordemos además que Seam también soporta los contenedores IBM Websphere, BEA WebLogic y Oracle OC4J.

Nota: todas las pruebas se realizaron sobre Linux (Gentoo amd64) y OpenSolaris 9.6. Sin embargo, hacerlo en Windows con los mismos pasos no debería presentar problemas; solo se debe tener cuidado con las rutas donde se guarden los archivos usados.

Prerequisitos

- [Eclipse 3.4 o 3.5](#)
- [La última versión de JBoss Seam](#) (la más reciente al momento de escribir este documento es la 2.2.0).
- [La última versión de Jboss Server](#) (la más reciente al momento de escribir este documento es la 5.1.0).
- Un gestor de bases de datos. En mi caso he usado MySQL, aunque hay soporte para muchos otros gestores. No olvides además descargar el conector JDBC, en mi caso el `mysql-connector-java-bin.jar`.

Instalación

Para instalar Seam y JBoss basta con descomprimir los paquetes que descarguemos de Internet. Yo recomiendo hacerlo en la carpeta `/opt` en sistemas UNIX/Linux, o directamente en `C:/` en sistemas Windows.

Creación de un proyecto base para Eclipse

SeamFramework nos proporciona una herramienta de línea de comandos para la generación de proyectos base. Vamos al directorio donde lo instalamos y ejecutamos:

```
./seam setup
```

Ahora debemos responder a cada una de las preguntas que nos hace; muchas de ellas las debemos dejar por defecto, así que nos concentraremos solo en aquellas en donde tengamos que cambiar. Resumiendo:

- El directorio donde se creará el proyecto:
[input] Enter the directory where you want the project to be created (should not contain spaces) [/home/funtoo/projects] [/home/funtoo/projects]
`/home/funtoo/Polli/Componentes/seam-framework/workspace`
- El directorio donde se encuentra el JBoss AS:
[input] Enter your JBoss AS home directory [C:/Program Files/jboss-5.1.0.GA]
[C:/Program Files/jboss-5.1.0.GA]
`/opt/jboss-5.1.0.GA/`
- El nombre del proyecto:
[input] Enter the project name [myproject] [myproject]
`holaseam`
- Vamos a desarrollar nuestra aplicación como EAR:
[input] Is this project deployed as an EAR (with EJB components) or a WAR (with no EJB support)? [war] (ear, [war])
`ear`
- Definimos el nombre del paquete en base de la aplicación:
[input] Enter the base package name for your Java classes [com.mydomain.holaseam]
[com.mydomain.holaseam]
`org.ejemplo.seam`

Los que hacen referencia a los Session Beans o Test Cases los podemos dejar como nos lo pongan.

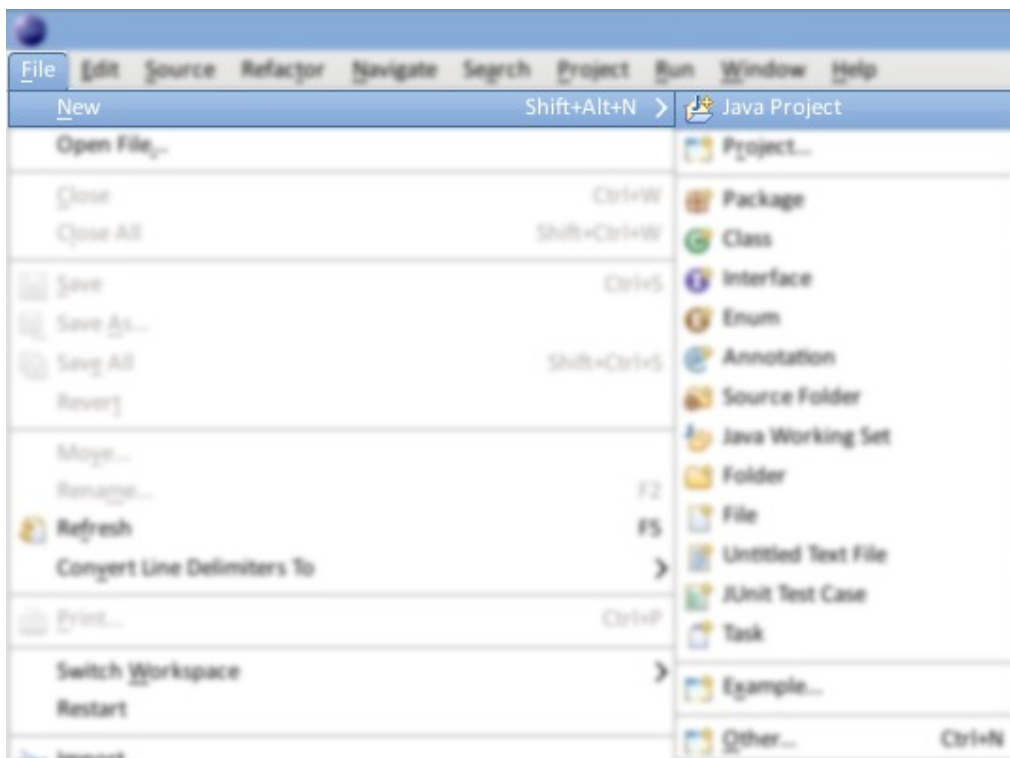
- El gestor de base de datos; en este caso usé mysql aunque debería ser prácticamente transparente:
[input] What kind of database are you using? [hsq1] ([hsq1], mysql, derby, oracle, postgres, mssql, db2, sybase, enterprisedb, h2)
mysql
- Ingresamos la ruta en donde se encuentra el driver JDBC:
[input] Enter the filesystem path to the JDBC driver jar [] []
/opt/conectores/mysql-connector-java-5.0.8-bin.jar
- Ahora debemos definir la URL de conexión a la base de datos. En este caso, he creado una base de datos llamada *ejemplo*, y por lo tanto la URL quedaría así:
[input] Enter the JDBC URL for your database [jdbc:mysql:///test]
[jdbc:mysql:///test]
jdbc:mysql://localhost:3306/ejemplo
- Con esta opción se ejecutará un script llamado import.sql cada vez que hagamos un deploy del proyecto. Dicho archivo deberá tener los scripts necesarios para crear tablas y registros. Puesto que en este laboratorio poco vamos a usar MySQL, puedes poner lo que quieras; pero es importante tener en cuenta esta opción a la hora de desarrollar un proyecto.
[input] Do you want to recreate the database tables and execute import.sql each time you deploy? [n] (y, [n])
y

Ya que hemos configurado lo que será el proyecto, es hora de crearlo. Ejecutamos el comando:

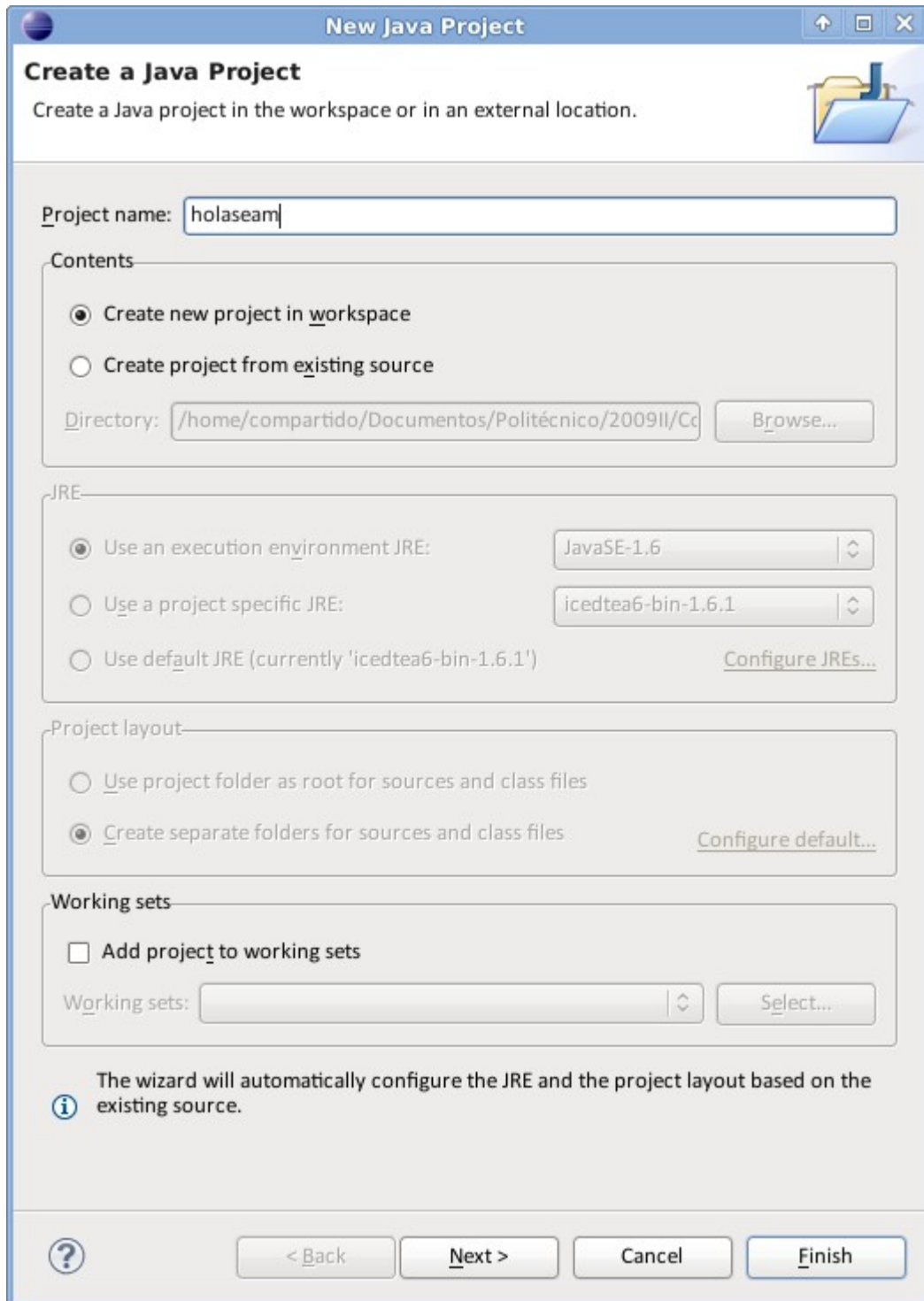
```
./seam create-project
```

Esto creará un proyecto base que puedes abrir con Eclipse, y eso es justo lo que haremos a continuación.

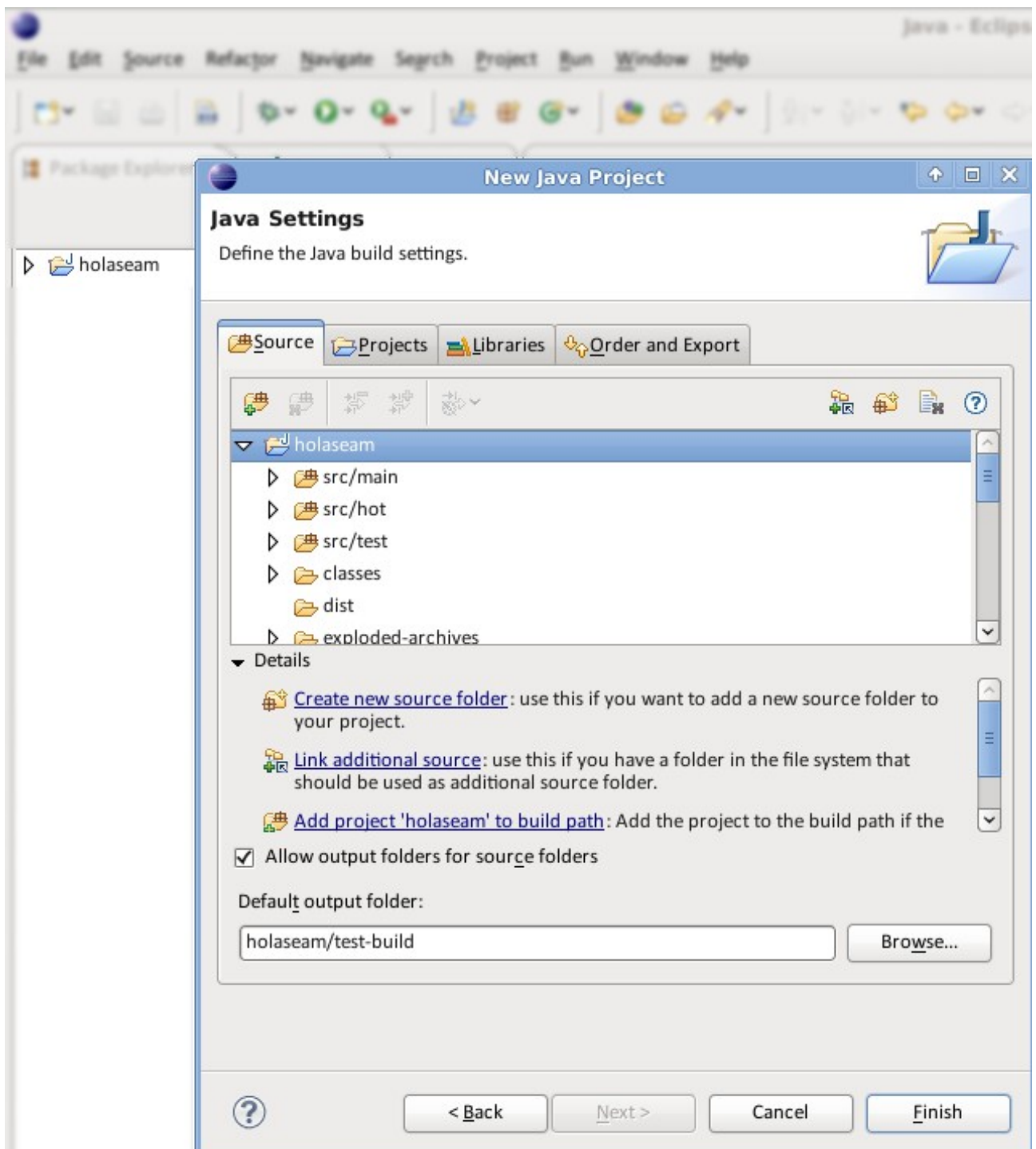
Abrimos eclipse y nos aseguramos que el workspace sea el mismo directorio en el que creamos el proyecto desde Seam. Vamos a la creación de un nuevo proyecto Java:



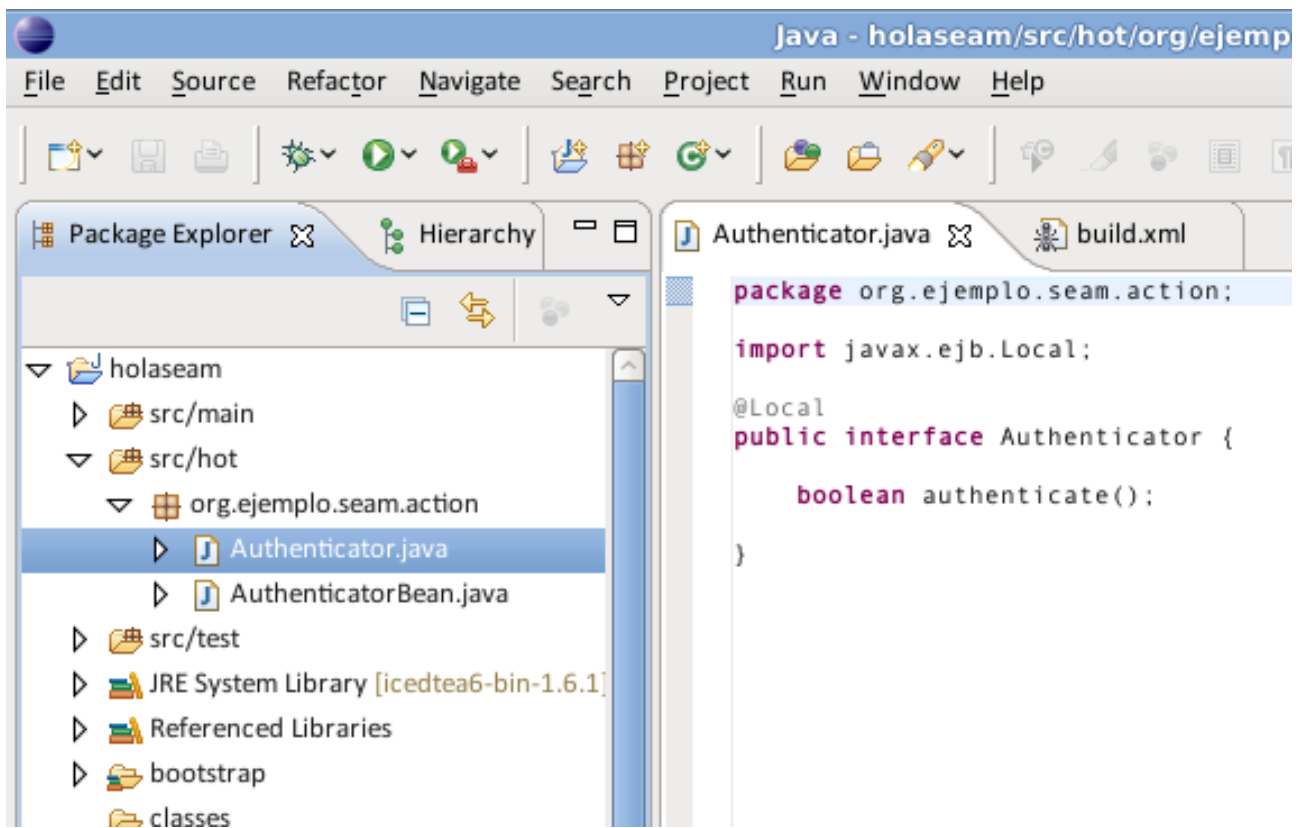
Le damos el mismo nombre que a nuestra aplicación Seam (en este caso *holaseam*):



Y puesto que ya existe un proyecto creado en el workspace muchas de las opciones ya aparecen asignadas. Hacemos clic en Next y vemos que ya hay una estructura de directorios creada con la base del proyecto Seam.



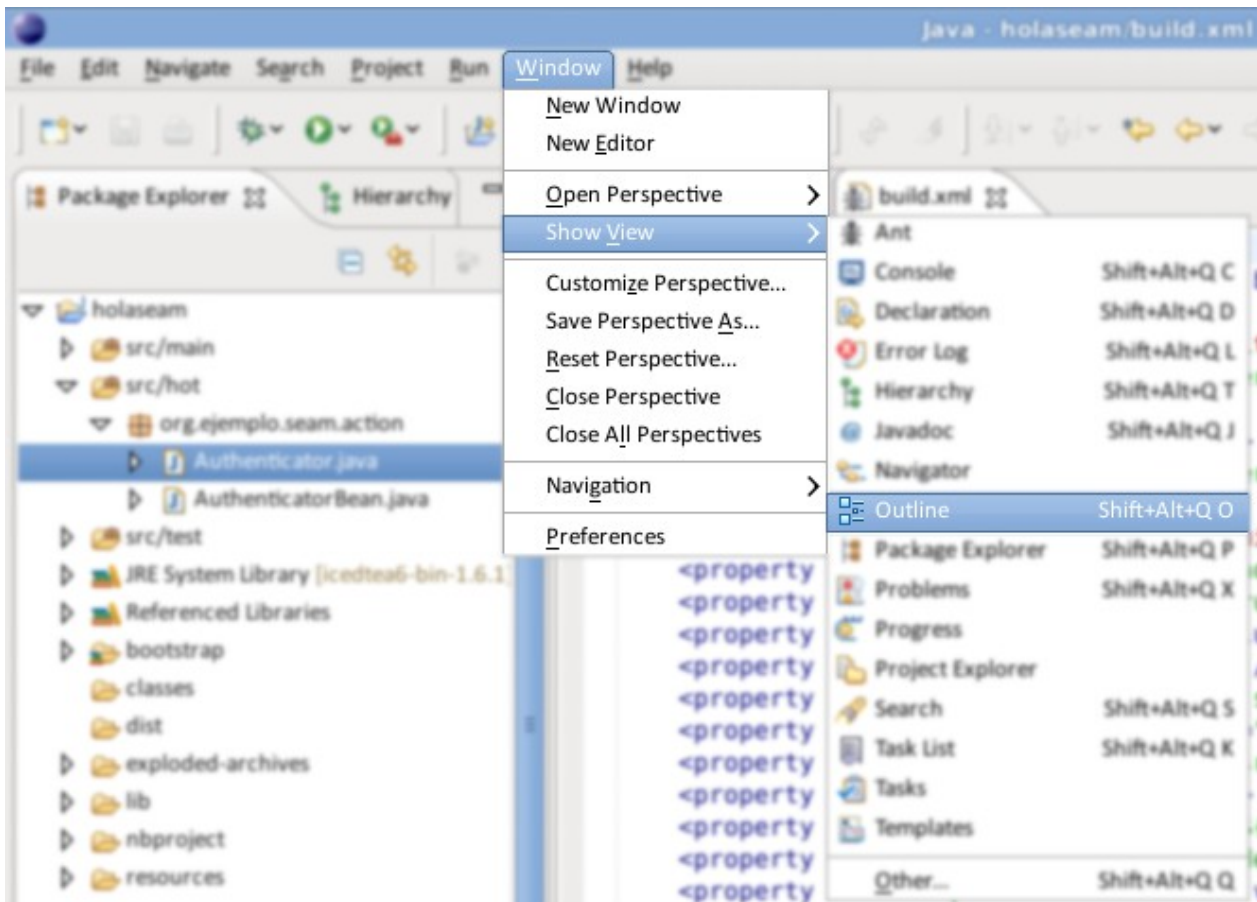
Hacemos clic en Finish, y ya podemos comenzar a editar y modificar el proyecto.



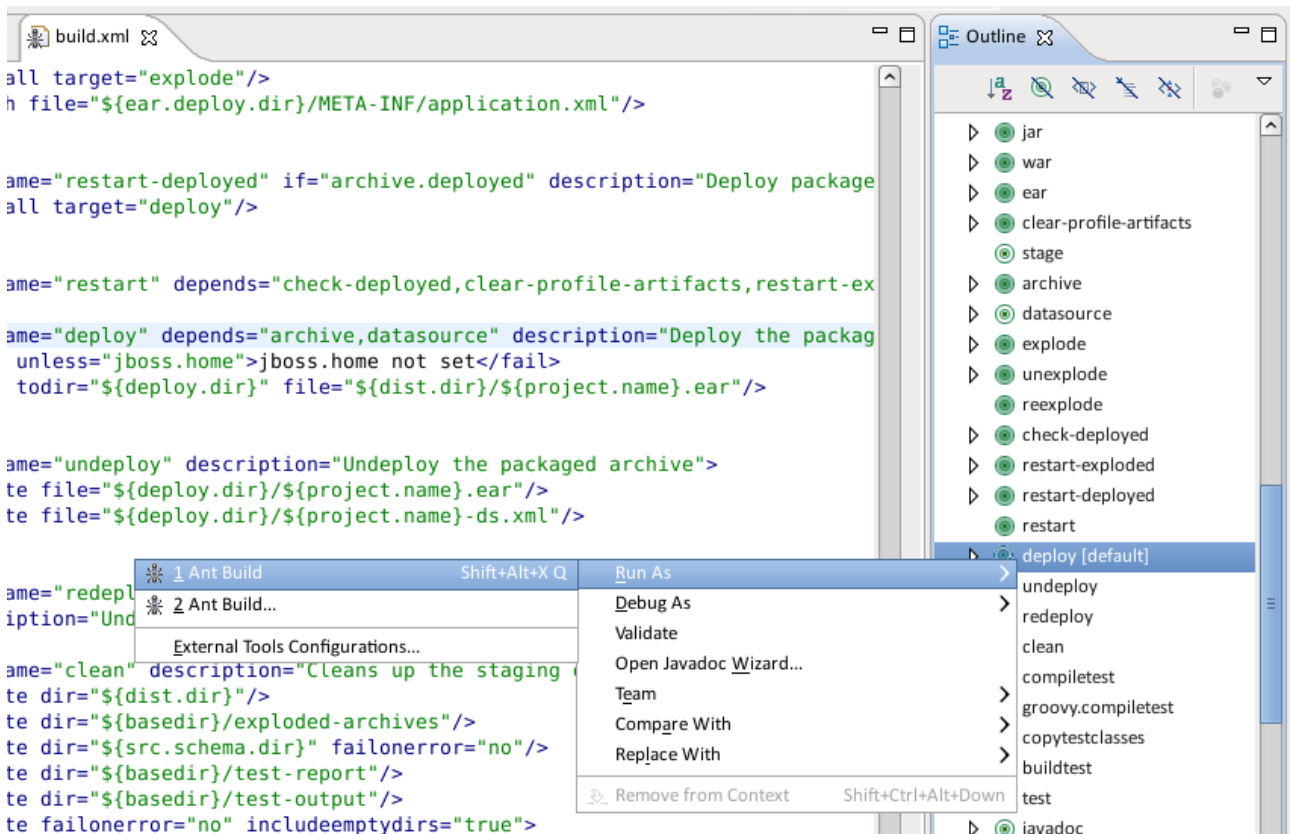
Despliegue del proyecto

Primero debemos iniciar el servicio de JBoss. Para ello ejecutamos el archivo run.sh (run.bat en Windows) del directorio bin de Jboss. Este dejará el servicio web corriendo por defecto en el puerto 8080.

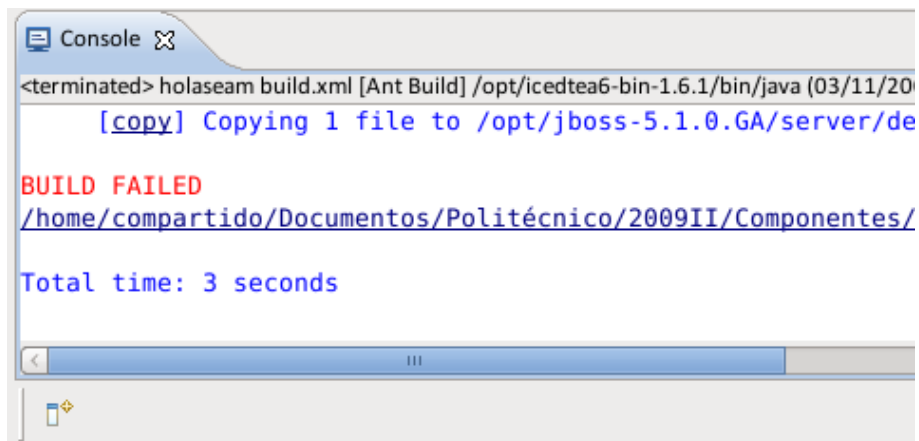
Para hacer el despliegue sobre JBoss, usamos un archivo Ant que se generó junto con el proyecto. Dicho archivo nos permite ejecutar tests, hacer (re)despliegues, generar el JavaDoc, entre otras. Así que habríamos el archivo build.xml y nos aseguramos que la vista Outline esté abierta:



Dentro de la vista Outline podemos ver los componentes del archivo build.xml y podemos ejecutarlos con Ant. En este caso, buscamos el componente `deploy`, hacemos clic derecho, `Run As`, y `Ant Build`.



Algunas veces se generan errores al hacer el despliegue:



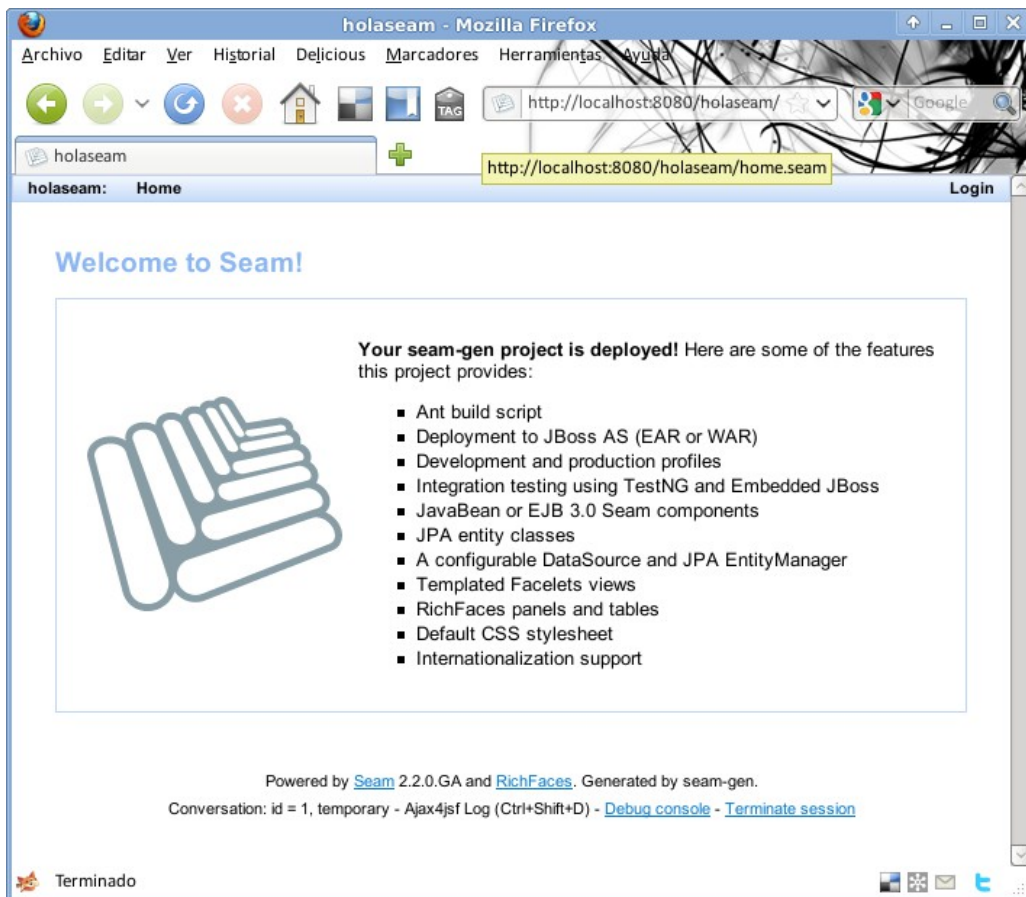
Si es tu caso, deberás eliminar los archivos y directorios que tengan el nombre del proyecto del directorio `deploy` en JBoss. En mi caso los elimino con este comando:

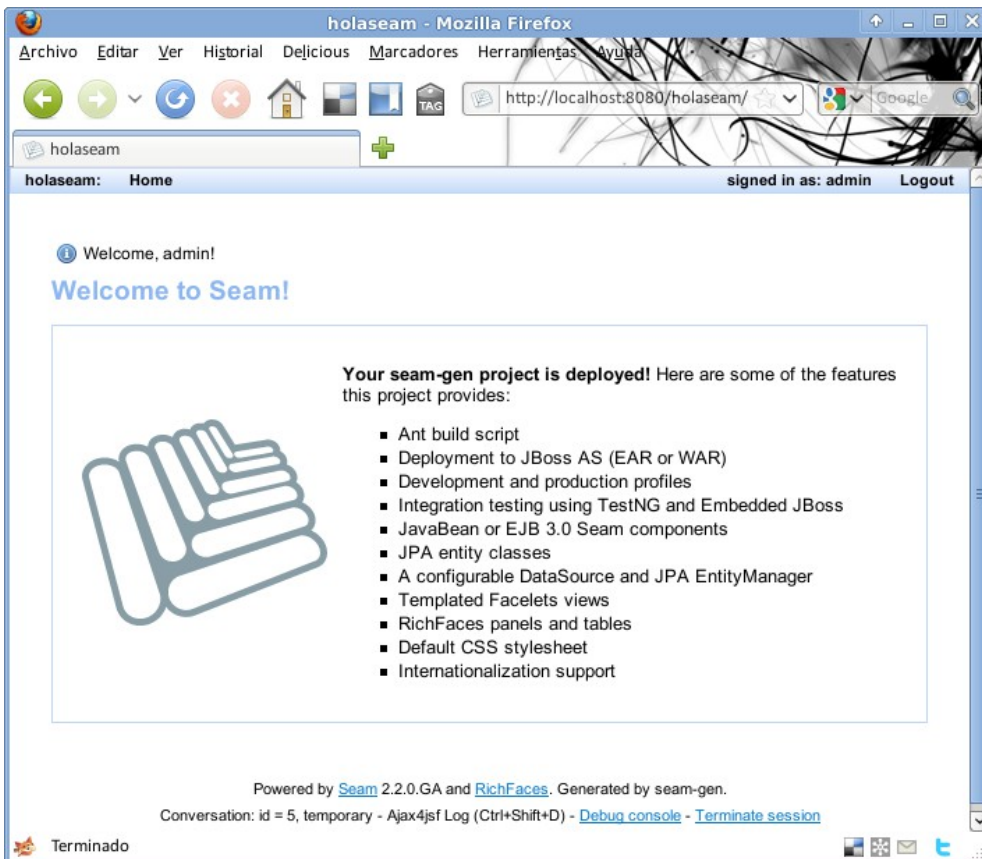
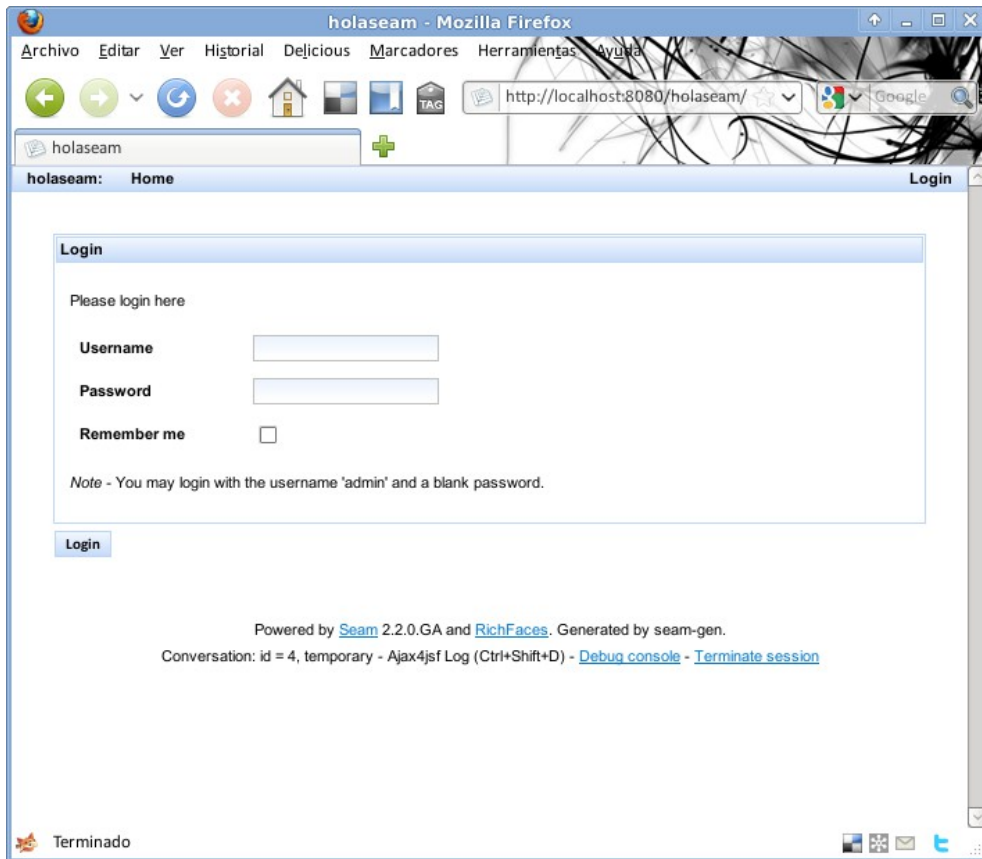
```
rm -rf /opt/jboss-5.1.0.GA/server/default/deploy/holaseam*
```

Si todo sale bien, cuando hagamos de nuevo *deploy* veremos algo como esto:

```
Console
<terminated> holaseam build.xml [Ant Build] /opt/icedtea6-bin-1.6.1/bin/java (03/11/2009 17:07:52)
Buildfile: /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspace/holaseam/build.xml
init:
clear-profile-artifacts:
  [delete] Deleting: /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspac
  [delete] Deleting: /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspac
  [delete] Deleting: /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspac
groovy.compile:
groovy.copy:
compile:
copyclasses:
jar:
  [copy] Copying 1 file to /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspac
  [copy] Copying 1 file to /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspac
war:
  [copy] Copying 1 file to /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspac
ear:
stage:
archive:
  [jar] Building jar: /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspac
  [jar] Building jar: /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspac
  [jar] Building jar: /home/compartido/Documentos/Politécnico/2009II/Componentes/seam-framework/workspac
datasource:
  [copy] Copying 1 file to /opt/jboss-5.1.0.GA/server/default/deploy
deploy:
  [copy] Copying 1 file to /opt/jboss-5.1.0.GA/server/default/deploy
BUILD SUCCESSFUL
Total time: 3 seconds
```

Es hora de probar la aplicación. Abrimos el navegador en la dirección: <http://localhost:8080/holaseam/>





Cristian Camilo Castiblanco
cristian@elhacker.net
<http://casidiablo.net>
Licenciado bajo GFDL

