

# Migrating/Porting Petstore 2.0 to SAP NetWeaver Composition Environment



## Applies to

**SAP NetWeaver Composition Environment 7.1 EhP1.** You will need the runtime (CE 7.10 Application Server Java) and the design time (the Eclipse-based SAP NetWeaver Developer Studio) in this document often abbreviated with the “IDE”.

**Petstore** – the AJAX-enabled Java EE 5 Reference Application as provided by Sun. The version used here is Petstore 2.0.

**GlassFish** – Sun’s Java Application Server, which is the reference implementation of the Java EE 5 runtime specification. The version used here is GlassFish V2.1. This document will most likely also apply for the GlassFish version 3.0.

**Note:** The migration described in this document has also been tested with and applies to NetWeaver Composition Environment 7.20.

## Summary

This document is a detailed step-by-step guide on migration of Petstore 2.0 to SAP NetWeaver Composition Environment. There is a [paper](#) published on SDN discussing general issues of migrating Java Enterprise Applications to SAP NetWeaver Java.

**Author:** Goran Stoiljkovski

**Company:** SAP AG

**Created on:** 19 March 2010

## Author Bio



Goran Stoiljkovski is a solution architect whose main focus is on architecture and design of distributed computed systems and solutions. Goran is a member of the global Co-Innovation Lab (COIL) team as part of the SAP Global Ecosystem and Partner Group.

## Table of Contents

Introduction .....	4
Setting up the Environment.....	5
Step 1 – Download and Install GlassFish .....	5
Step 2 – Download Petstore .....	6
Step 3 – Install Petstore.....	7
Step 4 – Build and Deploy Petstore on GlassFish .....	7
Step 5 – Install SAP NetWeaver Composition Environment .....	7
Initial Migration of the Petstore Application .....	7
Step 1 – Deploy a Derby JDBC Driver to NetWeaver .....	7
Login to the NetWeaver Administrator .....	8
Navigate to “Application Resources” .....	8
Deploy New JDBC Driver.....	8
Set a Logical Name for the Driver .....	9
Add New Driver File .....	9
Step 2 – Deploy the Petstore JDBC Data Source .....	9
Create a Custom JDBC Data Source.....	10
Custom Data Source Settings.....	10
Initial Connections of the Data Source.....	11
Successfully Configured Custom Data Source .....	11
Step 3 – Maintain the SAP Server in the IDE.....	11
Open the IDE .....	12
Add a default server instance.....	12
Result – the SAP system is recognized .....	12
Add the server view to the IDE.....	13
Result – the default server instance view .....	13
Step 4 – Create the IDE Projects Structure .....	13
Create an empty Dynamic Web Project .....	14
Create the structure of the web project .....	14
Adjust Build Path of the Web Project Sources .....	14
Create an empty Enterprise Project .....	15
Include the Web Project in the Build Configuration .....	15
Step 5 – Import Petstore’s Source and Binaries into the IDE.....	15
Import Petstore’s web resources into your web project.....	16
Copy the dependent libraries .....	16
Copy the JPA specific persistence.xml .....	16
Copy the deployment descriptors.....	16
Import Java sources into the IDE .....	17
Fix HTML validation errors .....	17
Clean the web project petstore .....	17
Step 6 – Deploy Petstore .....	17

Step 7 – Test the Deployment and Fix the Issues.....	19
The Search function issue.....	20
The Search function issue resolution .....	20
The Tags issue .....	21
The Tags issue resolution .....	23
The Seller function issue.....	23
The Seller function issue resolution .....	23
Copyright.....	25

## Introduction

The **SAP NetWeaver Composition Environment** is an SAP **product** suitable for design, assembly, deployment and productive operation of complex and enterprise ready composite applications. The core of this product is a Java EE 5 compliant and certified application server, even though this server does not appear on SAP's price list as a stand-alone product.

**Note:** In this guide we will often refer to this application server as the "SAP Java Engine".

**Note:** SAP NetWeaver Composition Environment is a highly configurable platform. It introduces a concept of the so-called "Zero-Admin-Templates". A Zero-Admin-Template is a preconfigured profile meeting the requirements for a dedicated usage of the platform. For example, one can use the Composition Environment for

- development of plain Java EE applications,
- as a development platform for business processes
- or even as a runtime platform for scalable and highly available composite applications in productive usage.

All these usage types require different configurations of the platform, e.g. the OS specific executables. For example, the heap size configuration of the Java Virtual Machine is somewhat different for a system used in development or in production.

The Zero-Admin-Template concept also allows for extensions, so you can define your own templates and thus your own usage type of the platform. One can switch among the various usage types by simply activating the corresponding template e.g. profile on the platform already installed, since the preconfigured templates all come with the same binary set of the product shipment.

A template called "javaee\_development" is the most simple usage type. It allows for development and deployment of "standard" Java EE applications. This usage type defines a runtime server comparable to GlassFish.

**Note:** This guide describes the migration of the Petstore 2.0 Java EE 5 reference application to the SAP Java Engine – the Java EE 5 compliant and certified application server.

The major steps performed here are

- Downloading and installing GlassFish
- Downloading and installing SAP NetWeaver Composition Environment 7.1 EhP1.
- Downloading and installing/deploying Petstore 2.0 to GlassFish.
- Running Petstore and checking the functionalities of the application
- Initial migration of the Petstore application to the SAP Java Engine
- Minor adjustment of the application's source code to fully run on SAP Java Engine.

**Note:** The steps are performed on a developer's desktop machine with JDK 1.5. You will not need more than 3 GB of RAM. The memory footprint on the local hard drive depends on the requirements of the software installed but should not exceed the total of 12 GB. The major part of this footprint goes to the system database of the SAP Java Engine (over 7 GB), which is built in the architecture of the platform and thus not surprising.

**Note:** You will be able to run both – GlassFish as well as the SAP Java Engine along with SAP's Developer Studio – on the very same machine at the same time. There is no remote installation or runtime access required. This way you will be able to easily check and compare the functionality of the Petstore application running on both platforms – GlassFish and SAP Java Engine.

## Setting up the Environment

This guide assumes that the Petstore 2.0 reference application runs on Glassfish v2 with an embedded Derby database as Petstore's data persistence storage. For more details on installing Glassfish and deploying Petstore please refer to the corresponding documentation. You can start by downloading the Petstore 2.0 reference application from Sun Developer Network.

**Please note that some of the Sun specific links referred to in this document may change over time.**

### Step 1 – Download and Install GlassFish

In this case we downloaded GlassFish from <http://java.sun.com>. Under the area “Popular Downloads” you have the Link “Java EE 5 SDK”. We have selected the package “GlassFish Java EE + JDK” for Windows environments as indicated in the screenshot to the right hand side (last column).

GlassFish and SDK Update 7 Downloads (Last Updated: September 17, 2009)

Platform:

Choose a Language:

Components	GlassFish + MySQL	GlassFish Java EE	GlassFish Java EE + JDK
<b>Licenses</b>	<a href="#">View License</a>	<a href="#">View License</a>	<a href="#">View License</a>
	<a href="#">Download</a>	<a href="#">Download</a>	<a href="#">Download</a>
GlassFish*	•	•	•
MySQL	•		
Samples		•	•
BluePrints		•	•
Tutorial		•	•
Documentation		•	•
Java SE (JDK)*			•
Open ESB			
Access Manager			
Portlet Container*			
WSRP*			
Portal Pack*			
NetBeans IDE*			
	<a href="#">Download</a>	<a href="#">Download</a>	<a href="#">Download</a>
	Free, 140MB	Free, 100MB	Free, 170MB

**Note:** In this case the GlassFish installation directory is C:\Sun\SDK. Note that this is the property you have to maintain in Petstore's `build.properties` in order to successfully build and deploy Petstore to Glassfish. This property is also needed for deployment of the tables in the embedded Derby database – so make sure you maintain it correctly.

Select Installation Directory

Installation Directory:

C:\Sun\SDK [Browse...](#)

**Note:** The admin password is set in this step. This will generate an admin-password file, which corresponds to the default settings of Petstore's build script – if you have chosen the GlassFish installation directory as described in the previous step.

**Admin Configuration**

Admin User Name:

Password (min 8 chars.):

Re-enter Password:

☒ Don't Prompt for Admin User Name and Password

*The admin user name and password will be stored in .asadminpass file in user's home directory and will not have to be provided when performing admin functions.*

☐ Prompt for Admin User Name and Password

*The admin user name and password must be provided when performing all admin functions.*

Admin Port:

HTTP Port:

HTTPS Port:

Here is a summary of the components to be installed.

**Ready to Install**

The following items for the product Java Platform, Enterprise Edition 5 SDK will be installed:

Product: Java Platform, Enterprise Edition 5 SDK

Location: C:\Sun\SDK

Space Required: 265.12 MB

---

Java 2 SDK, Standard Edition 6.0

Sun Java System Message Queue 4.1

Sun GlassFish Enterprise Server v2.1

Sample Applications

Java BluePrints

Your First Cup: An Introduction to the Java EE Platform

## Step 2 – Download Petstore


Download Petstore 2.0. Start at <http://java.sun.com/developer/releases/petstore/> ...

### Download the Java Pet Store

The Java Petstore 2.0 Early Access runs on the [Java EE 5 SDK](#). Petstore Demo 1.3.2 are also available.





... And navigate to the early access download area.

petstore				
Java Pet Store Reference Application, Version 2.0 Early Access				
Filter this list			<input type="text"/> <input type="button" value="Filter"/>	
Name	Status	Modified by	Size	Reservations
 <a href="#">javapetstore-2.0-ea5-installer.jar</a>	Draft	<a href="#">basler</a> on Thursday, August 9, 2007 at 10:28:10 PM	7.46 mB	

### Step 3 – Install Petstore

Install Petstore by unpacking the archive `javapetstore-2.0-ea5-installer.jar` at a destination of your choice.











Name	Size	Type
 javapetstore-2.0-ea5		File Folder
 javapetstore-2.0-ea5-installer.jar	7.639 KB	Executable Jar File

### Step 4 – Build and Deploy Petstore on GlassFish

Open the `index.html` in the Petstore distribution and follow the instructions to set up Petstore (build and deploy).

**Note:** it is recommended that you maintain the admin password file in your GlassFish instance as described in the instruction.

**Note:** Make sure you have started Glassfish and the Derby database before you run the ant scripts for setting up Petstore (build and deploy).

Name	Size	Type
 bp-project		File Folder
 lib		File Folder
 nbproject		File Folder
 setup		File Folder
 src		File Folder
 web		File Folder
 3RD-PARTY-LICENSE.txt	22 KB	Text Document
 bp-project.xml	1 KB	XML Document
 build.xml	4 KB	XML Document
 index.html	14 KB	HTML Document

### Step 5 – Install SAP NetWeaver Composition Environment

Obtain the SAP NetWeaver Composition Environment 7.1 EhP1, e.g. by downloading a trial version from SDN (<http://www.sdn.sap.com/irj/scn/nw-ce-downloads>).

You would have to install the runtime server (SAP NetWeaver CE Java) and the design time (SAP NetWeaver Developer Studio).

**Note:** A detailed installation guide is included in the download package as a PDF document.

### Initial Migration of the Petstore Application

This guide primarily describes the migration of the Petstore application. The Petstore application uses a persistence data store – the embedded Derby database that came with the GlassFish installation. Here is what you need to do in order to deploy the Derby JDBC driver to SAP NetWeaver.

#### Step 1 – Deploy a Derby JDBC Driver to NetWeaver

## Login to the NetWeaver Administrator

You need to be an administrator to be able to perform the actions described in this section. The default administrator user ID is “administrator” (case-insensitive). You have specified the password for this user during the installation of NetWeaver.



## Navigate to “Application Resources”

Tab Configuration Management → Infrastructure. This is the area where you can maintain your resource, as for example your custom DB data sources (instances of `javax.sql.DataSource`) and related JDBC drivers. This is the place where you manage other application resources, e.g. JCA and JMS resources as well as your own resource adapters.

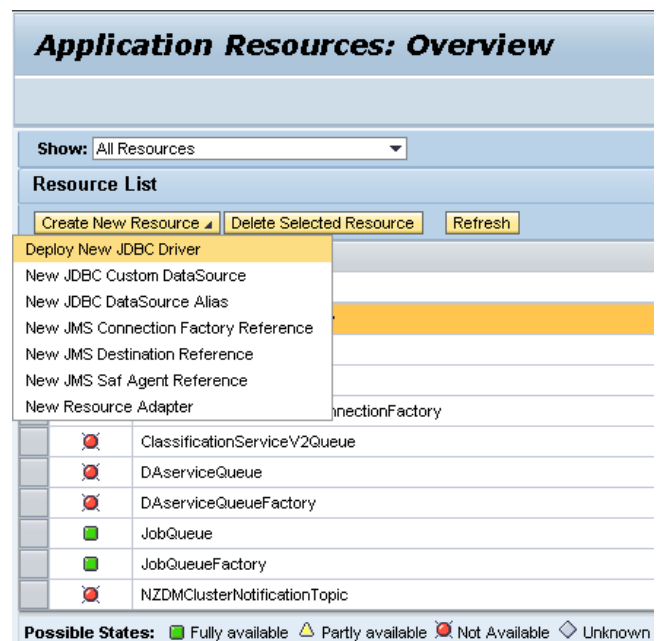


## Deploy New JDBC Driver

Deploy a Derby JDBC driver to NetWeaver. This means that you have to

- Give this driver a logical name and
- Upload a the Derby specific JAR file

See the next two steps for details.





### Set a Logical Name for the Driver

We named the driver as "Derby".

**Application Resources: Overview**

Show: All Resources

**New JDBC Driver Creation**

Save Cancel

Settings

JDBC Driver Name: \* Derby

Add New Driver File Remove Selected Driver File

File Name

No files to display

### Add New Driver File

The correct driver is called `derbyclient.jar` and can be found under `C:\Sun\SDK\javadb\lib` of your GlassFish installation directory.

Name	Size	Type
derby.jar	2,391 KB	Executable Jar File
derby.war	2 KB	WinRAR archive
derbyclient.jar	502 KB	Executable Jar File
derbyLocale_de_DE.jar	96 KB	Executable Jar File
derbyLocale_es.jar	91 KB	Executable Jar File
derbyLocale_fr.jar	97 KB	Executable Jar File
derbyLocale_it.jar	90 KB	Executable Jar File
derbyLocale_ja_JP.jar	111 KB	Executable Jar File
derbyLocale_ko_KR.jar	104 KB	Executable Jar File
derbyLocale_pt_BR.jar	86 KB	Executable Jar File
derbyLocale_zh_CN.jar	95 KB	Executable Jar File
derbyLocale_zh_TW.jar	97 KB	Executable Jar File
derbynet.jar	225 KB	Executable Jar File
derbyrun.jar	6 KB	Executable Jar File
derbytools.jar	153 KB	Executable Jar File

## Step 2 – Deploy the Petstore JDBC Data Source

Now you have to deploy a JDBC Data Source used by the Petstore application. These are the steps to do so and the necessary settings:

## Create a Custom JDBC Data Source

Navigate to Application Services and select “New JDBC Custom DataSource”.

### Application Resources: Overview

Show: All Resources

#### Resource List

Create New Resource

Delete Selected Resource

Refresh

Deploy New JDBC Driver

New JDBC Custom DataSource

New JDBC DataSource Alias

New JMS Connection Factory Reference

New JMS Destination Reference

New JMS Saf Agent Reference

New Resource Adapter

ConnectionFactory

ClassificationServiceV2Queue





DAserviceQueue

DAserviceQueueFactory

Derby

JobQueue

JobQueueFactory

Possible States:  Fully available  Partly available  Not Available  Unknown

## Custom Data Source Settings

Here are the settings and some detailed explanations on their meaning. Some of them are standard Java JDBC API attributes.

**“Application Name”:** Every data source is being deployed and thus managed as an application. This is the name of this application. This is NOT the JNDI name of the data source. You can set it to **“PetstoreDB”**. This is only a SAP-relevant name, so you can freely choose it.

**“DataSource Name”:** This is the JNDI name of the data source – as you would find it in the `web.xml` or `persistence.xml` of the application. It is highly recommended to keep the name as defined in the Petstore application: **“jdbc/PetstoreDB”**.

**“Isolation Level”:** This is a support level of SAP’s database abstraction. For more details on the concept consult online documentation <http://help.sap.com>. Set this level to **“Vendor SQL”**.

**“Driver name”:** Here you can select the **“Derby”** driver you have previously deployed.

**“Driver Class Name”:** In this case `org.apache.derby.jdbc.ClientDriver`

**“JDBC URL”:** JDBC URL of the petstore database: `jdbc:derby://<host_name>:1527/petstore`

**“User ID”:** APP

**“Password”:** APP

### Application Resources: Overview

Show: JDBC Custom DataSources

#### New JDBC Custom DataSource Creation

Save Cancel

Settings

Connection Pooling

Additional Properties

JDBC DataSource Aliases

Application Name: PetstoreDB

DataSource Name: \* jdbc:PetstoreDB

Driver Name: \* Derby

SQL Engine: \* Vendor SQL

Isolation Level: \* Default

JDBC Version: \* 1x (without XA support)

Driver Class Name: \* org.apache.derby.jdbc.ClientDriver

Database URL: \* jdbc:derby:localhost:1527/petstore;create=false

User Name: \* APP

Password: \* ...

Description:

### Initial Connections of the Data Source

It is recommended that you set the initial connections for this data source to something different than “0”. This is so to speak a design time check for the correctness of your configuration. This will prevent you from getting a runtime exception in the application caused by connection refusal.

For migration purposes it is sufficient to set this value to “1”. You will find these settings under the sub tab “Connection Pooling” of the data source.

Save the settings by pressing the “Save” button.

**Application Resources: Overview**

Show: JDBC Custom DataSources

**New JDBC Custom DataSource Creation**

Save Cancel

Settings Connection Pooling Additional Properties JDBC DataSource Aliases

Initial Connections: \* 1

Maximum Connections: \* 25

Maximum Time to Wait for Connection: \* 120

☐ Expiration

### Successfully Configured Custom Data Source

Now you can see the operational data source.

**Application Resources: Overview**

Show: JDBC Custom DataSources

**Resource List**

Create New Resource Delete Selected Resource Refresh

State	Resource Name
	jdbc:PetstoreDB

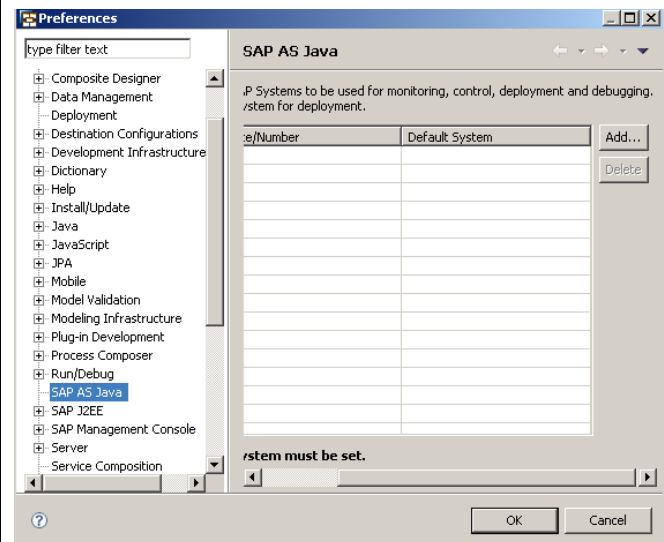
### Step 3 – Maintain the SAP Server in the IDE

In order to deploy the migrated application more efficiently, we suggest that you maintain the server instance in the SAP NetWeaver Developer Studio (IDE). This will allow for deployment of the application straight from the IDE.

## Open the IDE

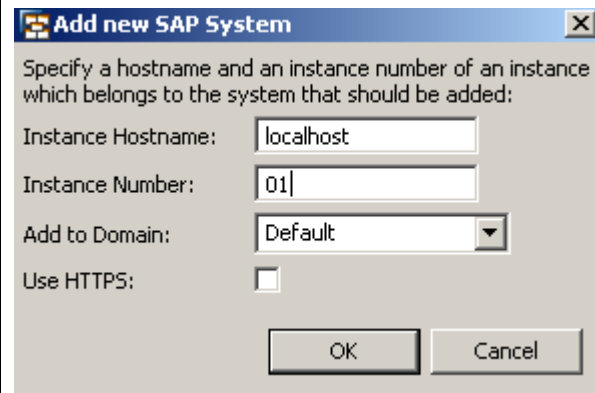
Open the IDE and go to Window → Preferences. Select “SAP AS Java” from the list of resources and add a default system. “Default system” means that this will be the system that the IDE will be deploying to by default.

**Note:** in a migration scenario you usually have only one system – the default one.



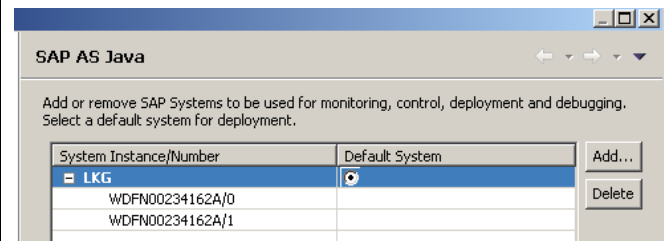
## Add a default server instance

Press the “Add” button and specify the settings for the instance host, the instance number and the domain as indicated in the screen shot on the right hand side.



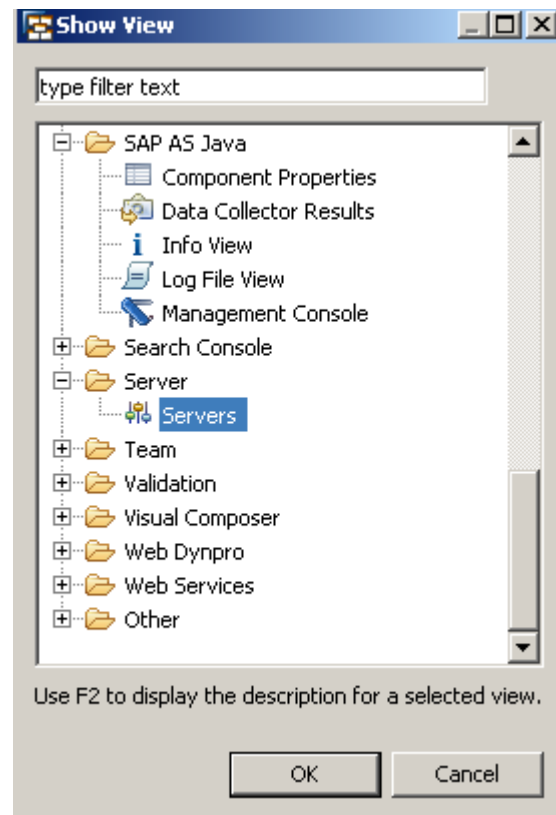
## Result – the SAP system is recognized

The IDE automatically recognized the SID and the DNS mapping of the SAP server running on the specific host – as defined maintained in the previous step.



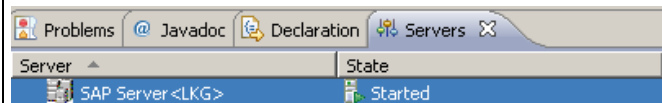
### Add the server view to the IDE

Go to Window → Open View → Other → Server → Servers and press “OK”.



### Result – the default server instance view

This opens the server view (Eclipse view) of your default server instance. In this particular case one observes that the default server instance is already started.



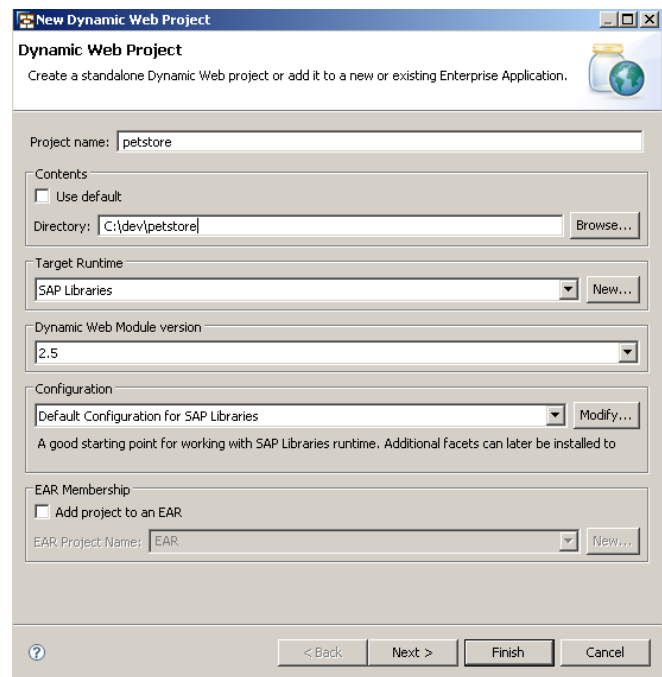
### Step 4 – Create the IDE Projects Structure

This will enable you to assemble the deployable archives and the code changes. Open the IDE if not already opened and switch to the Java EE perspective.

## Create an empty Dynamic Web Project

Go to File → New → Dynamic Web Project. Make sure the web module's version is 2.5. In this case we are not saving the project in the default workspace but rather under C:\dev\petstore. Changing the (Eclipse IDE) proposed default value has no implication on the migration.

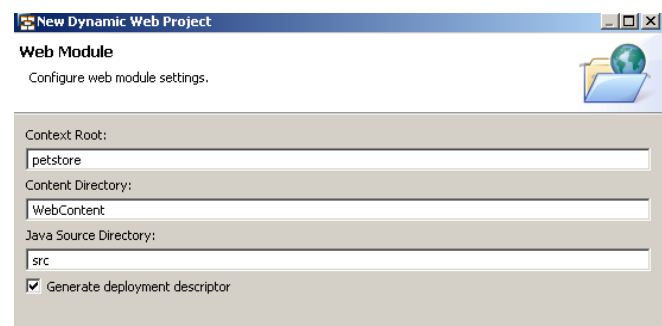
Click on “Next” to go to the next step.



## Create the structure of the web project

In this step you will create the structure of the web project as it will be managed by the IDE.

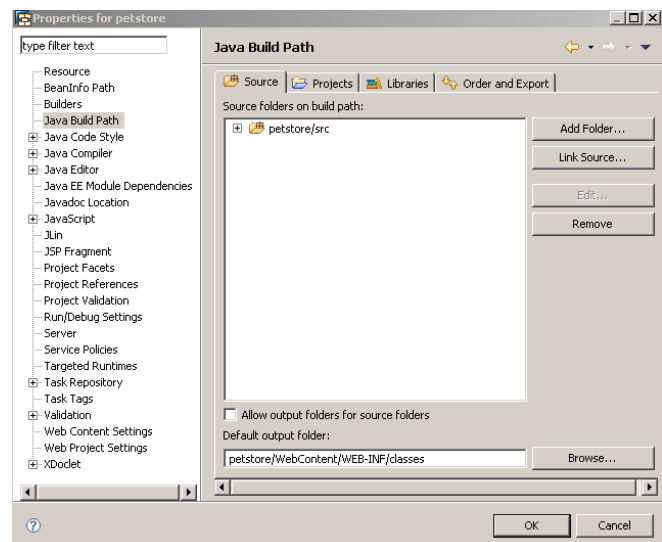
**Important:** Here you also set the context root of the migrated application. Set it to “petstore” as this is the context root of the original application. Press “Finish” to conclude the creation of the Eclipse project.



## Adjust Build Path of the Web Project Sources

In the preferences of the Dynamic Web Project (context menu of the web project “petstore”) go to “Java Build Path” and set the default output folder for the binaries to petstore/WebContent/WEB-INF/classes rather than the default value WEB-INF/classes.

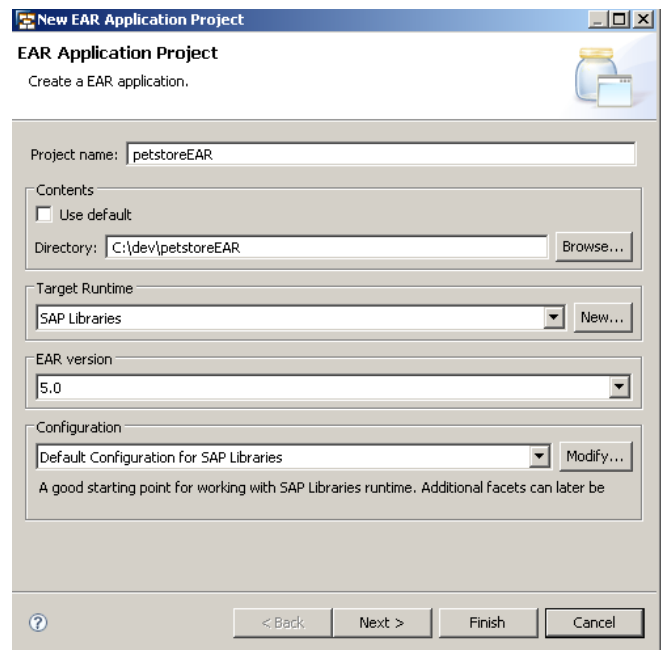
**This way the subsequent source code changes will be directly maintained in the Web Project archive (the WAR file).**



## Create an empty Enterprise Project

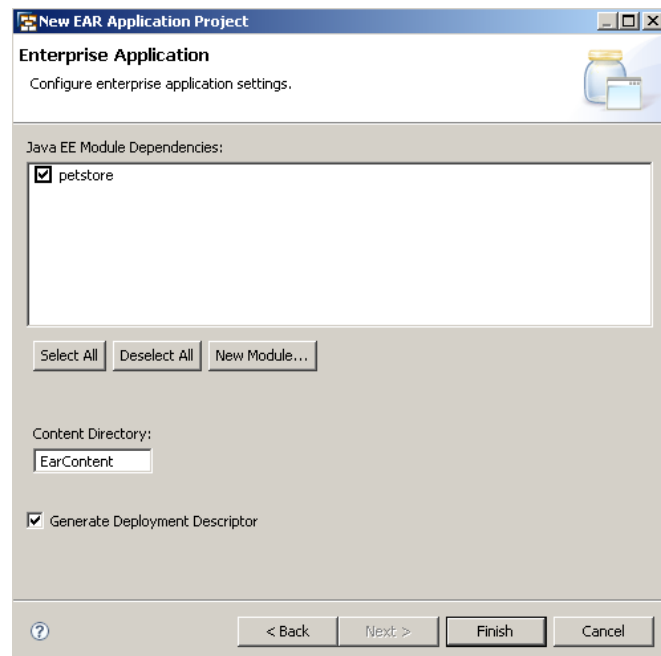
Go to File → New → Enterprise Application Project.

Again, as in case of the web project, we are storing the content of the project under `C:\dev\petstoreEAR` and not under the path proposed by Eclipse (the IDE).



## Include the Web Project in the Build Configuration

This will create a build dependency to the Web Project. This means that every time the Enterprise Archive (EAR) is built, the Web Archive is built prior to that action.



## Step 5 – Import Petstore’s Source and Binaries into the IDE

During regular development you usually create (web) artifacts by using the IDE wizards. Since we are dealing with a migration here, the development artifacts are already created. They just have to be imported in the IDE. We do so by performing copy-and-paste actions on file system level and subsequently refreshing the project content in the IDE.

**Note:** The Petstore’s original web project root is `C:\javapetstore-2.0-ea5\build\web`. Every operation performed and described in the following section refers to it.

## Import Petstore's web resources into your web project

Copy the web content `C:\javapetstore-2.0-ea5\build\web` (all but the `WEB-INF` folder) to `C:\dev\petstore\WebContent`.

Do not copy the `WEB-INF` folder now. There are adjustments to be done here, which are described in the next steps.

**Note:** as you may notice, we copy the Java sources in the web project. The Java sources also contain the business logic as well as the persistence classes (entity objects). **Petstore does not work with Entity EJBs but simple JPA persistence units.**

Name	Size	Type	Date Modified
Images		File Folder	31.07.2009 16:25
WEB-INF		File Folder	31.07.2009 16:25
accordion.css	2 KB	Cascading Style Sh...	24.09.2009 10:45
accordion.js	10 KB	JScript Script File	24.09.2009 10:45
banner.jsp	4 KB	JSP File	24.09.2009 10:45
bp_petstorelist.css	1 KB	Cascading Style Sh...	24.09.2009 10:45
bp_petstorelist.js	12 KB	JScript Script File	24.09.2009 10:45
bp_petstorelist_dom.js	6 KB	JScript Script File	24.09.2009 10:45
bp_petstorelist_one_div.js	8 KB	JScript Script File	24.09.2009 10:45
captchaerror.jsp	1 KB	JSP File	24.09.2009 10:45
catalog.js	13 KB	JScript Script File	24.09.2009 10:45
catalog.jsp	6 KB	JSP File	24.09.2009 10:45
catalog.jsp-old	6 KB	JSP-OLD File	24.09.2009 10:45
common.js	3 KB	JScript Script File	24.09.2009 10:45
download.jsp	1 KB	JSP File	24.09.2009 10:45
downloadAd.js	2 KB	JScript Script File	24.09.2009 10:45
downloadAd.json	1 KB	JSON File	24.09.2009 10:45
downloadAd.txt	1 KB	Text Document	24.09.2009 10:45
engine.js	11 KB	JScript Script File	24.09.2009 10:45
fileupload.jsp	13 KB	JSP File	24.09.2009 10:45

## Copy the dependent libraries

Copy the jars from `C:\javapetstore-2.0-ea5\build\web\WEB-INF\lib` to `C:\javapetstore-2.0-ea5\build\web\WEB-INF\lib` (all but the `petstoreIndexes.zip`).

**Note:** The `petstoreIndexes.zip` is a file based database search index. Petstore uses Apache's Lucene to create a denormalized search index of the database content and the search in the application is performed against this index and not against the database. Unfortunately this index is deployed and maintained by Petstore in a very rigid and inflexible way.

Name	Size
bp-ui-5.jar	1,514 KB
commons-fileupload-1.1.1.jar	32 KB
commons-io-1.2.jar	65 KB
commons-logging-1.1.jar	52 KB
jdom.jar	150 KB
lucene-core-2.0.0.jar	395 KB
petstoreIndexes.zip	26 KB
rome-0.8.jar	193 KB
rome-fetcher-0.7.jar	22 KB
shale-remoting.jar	39 KB

## Copy the JPA specific persistence.xml

Put the `C:\javapetstore-2.0-ea5\build\web\WEB-INF\persistence.xml` under `C:\dev\petstore\src\META-INF` of the web project. This is the configuration file for JPA. This will allow for assembling this configuration file in the enterprise deployment archive.

Name	Size	Type
classes		File Folder
lib		File Folder
faces-config.xml	4 KB	XML Document
persistence.xml	1 KB	XML Document
sun-web.xml	1 KB	XML Document
web.xml	5 KB	XML Document

## Copy the deployment descriptors

Copy the descriptors `web.xml` and `faces-config.xml` to `WEB-INF` of the SAP web project (from `C:\javapetstore-2.0-ea5\build\web\WEB-INF` to `C:\dev\petstore\WebContent\WEB-INF`).

**Note:** You may want to add a HTTP proxy in the `web.xml` if you want to get the RSS feeds displayed in the application and your SAP Engine is behind a firewall.

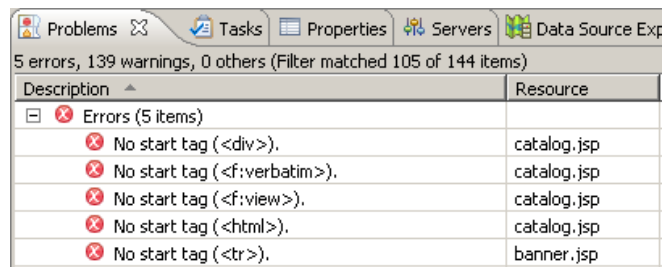
Name	
classes	
lib	
faces-config.xml	
persistence.xml	
sun-web.xml	
web.xml	



### Import Java sources into the IDE

- Copy the whole Java sources tree from  
C:\javapetstore-2.0-ea5\src\java to  
C:\dev\petstore\src.
- Refresh the IDE project.

You will get errors in the JSP files due to erroneous JSP files. This will prevent the Web Project from building, so you have to get rid of those errors first.

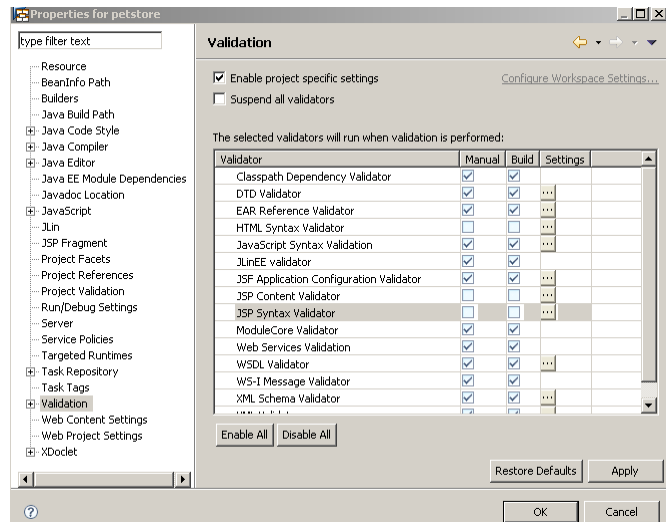


### Fix HTML validation errors

The easiest way to proceed is to turn off HTML and JSP syntax validation in your Web Project.

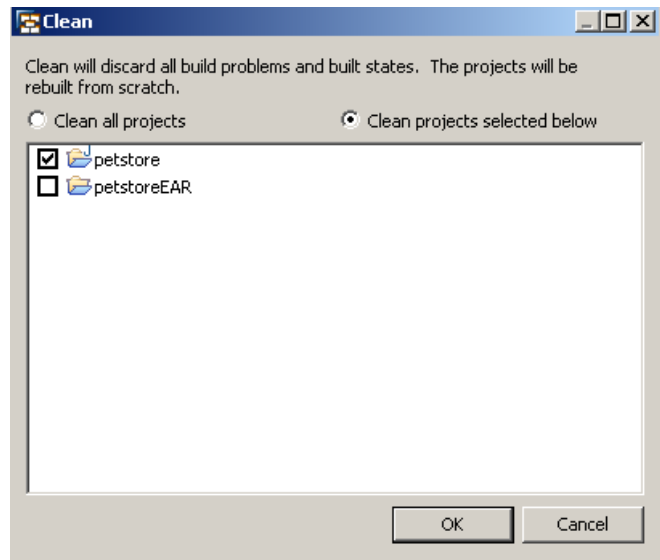
Go to the properties of the “petstore” project (context menu) and switch off:

- The HTML Syntax Validator
- The JSP Content Validator
- The JSP Syntax Validator



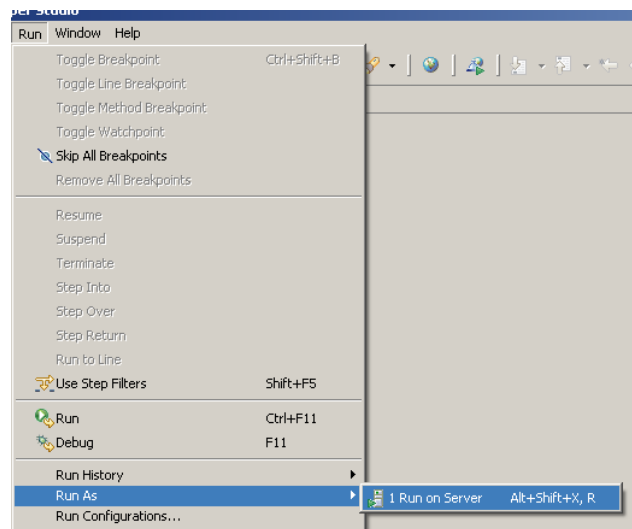
### Clean the web project petstore

After cleaning the project “petstore” the errors are gone. Go to Project → Clean and select the project “petstore”.

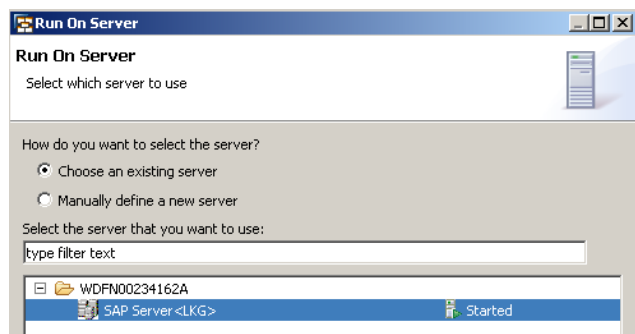


## Step 6 – Deploy Petstore

Mark the Enterprise Application Project “petstoreEAR” and from the main menu select Run → Run As → Run on Server.

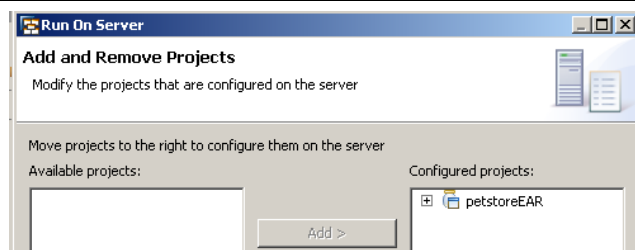


Then select your default server ...

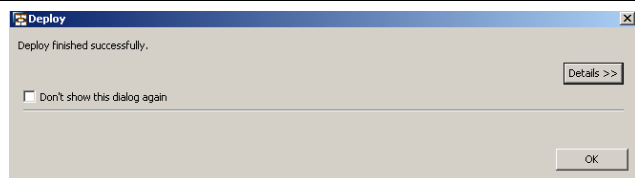


... And then the application you want to deploy – in this case “petstoreEAR”. Click “Finish” to start the deployment.

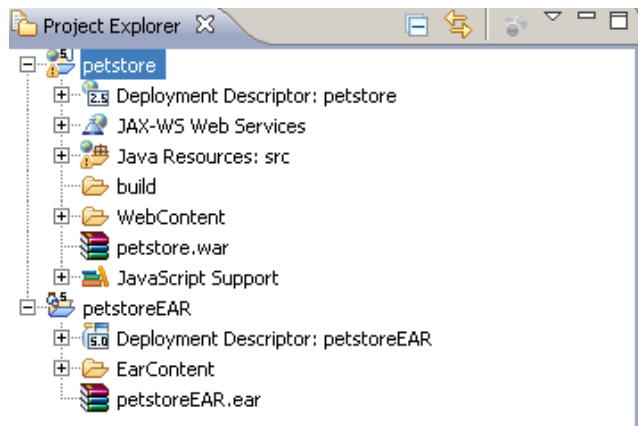
If this is your first deployment in a newly created IDE process – which means, if you have just started your IDE and have not performed any deployments yet – the IDE will ask you for your credentials, i.e. the credentials of the/an Engine’s administrator.



You will get a notification of the successful deployment of the application.



You will also see that the deployment archives (petstore.war and petstoreEAR.ear) for the projects have also been created under the defined folder structure.



## Step 7 – Test the Deployment and Fix the Issues

<http://localhost:50000/petstore/faces/index.jsp> is the entry URL of the migrated petstore application. One observes that the entry page is populated with data initially fetched from the database. But by browsing the application one soon notices that there are broken functionalities:

- The search function returns an empty data set, which results with a minimalistic page, even though there is no error message on the page itself.
- The tag.jsp – which is the tag cloud of the application – throws an error, when trying to follow a tag link.
- The upload function behind the Seller function throws an error.

In the next section we will take a closer look at these issues and their resolution.

**General Remark:** Petstore manages application artifacts dynamically but in a very inflexible way. Take a look at the utility class `com.sun.javaee.blueprints.petstore.util.PetstoreConstants.java`. Here is the relevant code I want to draw your attention to:

```
package com.sun.javaee.blueprints.petstore.util;

public class PetstoreConstants {

    public static final String PETSTORE_INDEX_DIRECTORY =
        System.getProperty("com.sun.aas.instanceRoot") + "/lib/petstore/searchindex";

    public static final String PETSTORE_IMAGE_DIRECTORY =
        System.getProperty("com.sun.aas.instanceRoot") + "/lib/petstore";

    ...
}
```

Here you see that there is obviously a JVM system property "com.sun.aas.instanceRoot", which the application looks up at runtime and thus dynamically creates the path to the Lucene index and to the image directory (see the note on Lucene in Step 5). These paths are then stored in the Java fields `PETSTORE_INDEX_DIRECTORY` and `PETSTORE_IMAGE_DIRECTORY` respectively.

**Note:** This is in our opinion a bad application design and works against the Java EE programming paradigms. The concept of Java based web applications foresees application folder structures like `WEB-INF` to handle such requirements. Using a JVM system property is a rather bad choice in this case.

### The Search function issue

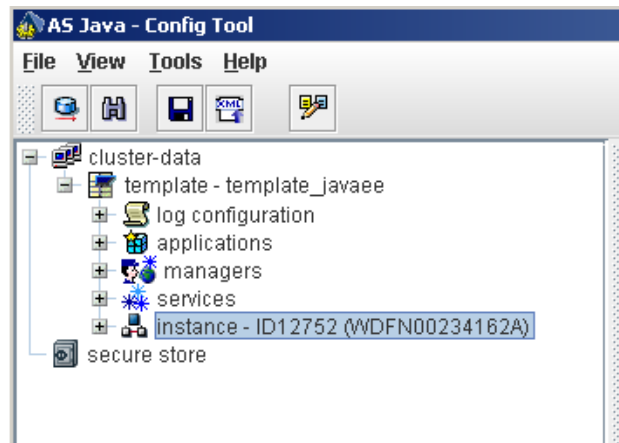
The function behind the Search Link of the upper navigation (<http://localhost:50000/petstore/faces/search.jsp>) returns an empty result set. This is due to the fact that this search is not performed against the data base but against (Lucene) files stored on the file system. The path to those files is obtained as a Java System Property, which is unknown to the NetWeaver server at this point in time.

### Search Page

### The Search function issue resolution

You would have to maintain the system property `com.sun.aas.instanceRoot` for the JVM.

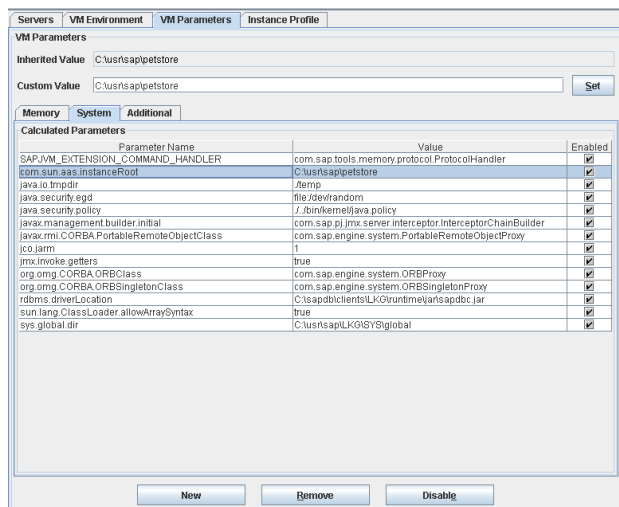
Open the NetWeaver Config Tool with the batch file `C:\usr\sap\KKG\J00\j2ee\configtool\configtool.bat`. The Config Tool will load the system properties of your instance from the underlying system database. Click on the instance ...



... and go to the tab "VM Parameters" and the sub tab "System" as indicated in the screen shot on the right hand side.

Create the new system property `com.sun.aas.instanceRoot` and choose a value for it. In our case, we set this value to `C:\usr\sap\petstore`. Enable this setting and save it.

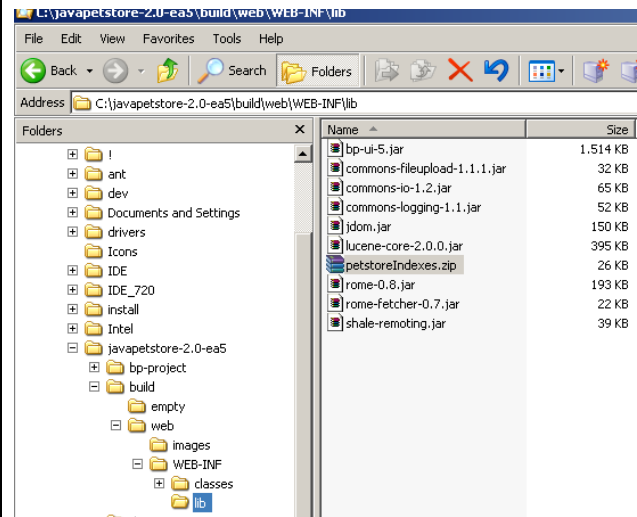
**Note that you will be required to restart your system in order for your setting to take effect. This is not surprising, since we are dealing with a JVM system property.**



Now you would have to manually create this path and add the search index files and the images folder to it.

- Create the folder structure  
C:\usr\sap\petstore
- The required (Lucene) search index is provided as zip file called `petstoreIndexes.zip` in the Petstore 2.0 distribution. Extract the file  
C:\javapetstore-2.0-ea5\build\web\WEB-INF\lib\petstoreIndexes.zip to  
C:\usr\sap\petstore\lib\petstore
- Create the folder C:\usr\sap\petstore\lib\petstore\images

**Note:** The images folder is also used by the “Seller” function to upload images as you want to define your own pets to be sold on the online shop.



Now the search works! The screen shot at the right hand side is the result of the search for “dogs”

**Search Page**

Search String:  Also Search Tags: ☒

Map	Name	Description	Tags	Price
<input type="checkbox"/>	<a href="#">Sad Puppy</a>	Help to put a smile on the face of this puppy. What? You did not know dogs could smile? Well, see what happens when you drop some steak on the floor!	<a href="#">fun</a> <input type="button" value="Add Tags"/>	\$99.00

Center Point Address:  For example: 4140 Network Circle, Santa Clara, CA, 95054  
 Area (in Miles):

### The Tags issue

The pets are tagged with attributes in the petstore application. So there is a many-to-many relationship between the tags and the pets. In the database, the pets are stored in the table “ITEM” and the tags in the table “TAG”.

The Tag Page (`Tag.jsp`) is a so-called tag cloud displaying all the tags available. A click on one of the tags resolves this relationship. You can see the tag cloud on the screen shot to the right hand side.

**Tag Page**

[awesome](#)  
  
[sunset](#)

[cool](#)  
  
[superior](#)

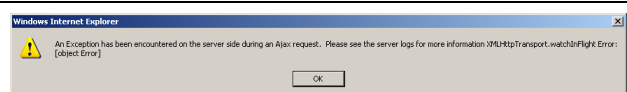
[excellent](#)  
  
[worthless](#)

[fun](#)  
  
[superior](#)

[inferior](#)  
  
[superior](#)

[interesting](#)  
  
[superior](#)

But the tag-item relationship is broken in the migrated application. So a click on one of the tags in <http://localhost:50000/petstore/faces/tag.jsp> throws an error.



Take a look in NetWeaver's main trace file by invoking the Log Viewer, e.g.:

<http://localhost:50000/nwa> → Problem Management  
→ Logs and Traces → Log Viewer → Show View → General → Default Trace.

There you will find something like this:

```
javax.persistence.PersistenceException: The
relationship >>items<< of entity
{com.sun.javaee.blueprints.petstore.model.Tag(tagID=3)} cannot be loaded because the
entity is detached
```

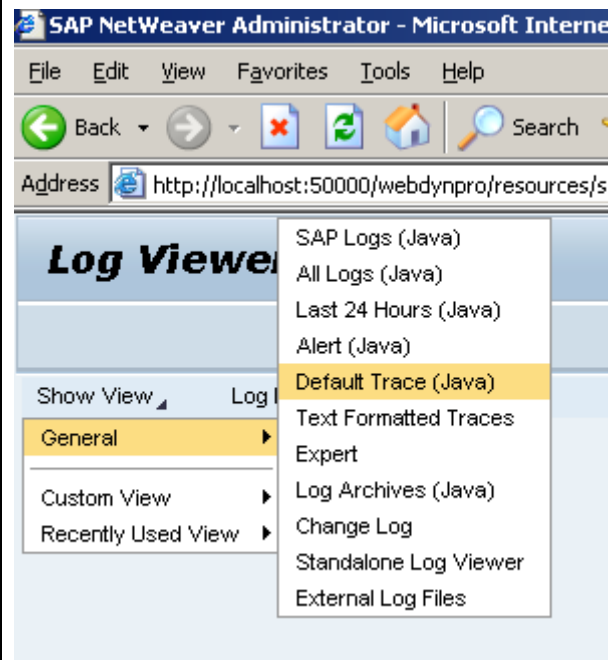
or

```
javax.persistence.PersistenceException: The
relationship >>tags<< of entity
{com.sun.javaee.blueprints.petstore.model.Item(itemID=10)} cannot be loaded because the
entity is detached
```

Here you see the root cause of the problem: the persistent objects **Item** and **Tag** (along with their relationships **tags** and **items** defined in the entities) are detached at runtime.

**The reason for this is the different implementation of the so-called JPA fetch strategy in GlassFish vs. the one in NetWeaver.**

GlassFish implements this fetch strategy for many-to-many entity relationships as "eager" opposed to NetWeaver's "lazy" implementation. By the way, the JPA spec foresees the "lazy" implementation.



### The Tags issue resolution

The resolution of the problem is adjusting the source code of the persistence entities (Entity EJBs) `Item.java` and `Tag.java`.

Add the property `fetch = FetchType.EAGER` to the `@Many-To-Many` annotation in both files.

**Save the changes, build the projects and redeploy the application and thus verify the fix**

In `Tag.java`:

```
...
@ManyToMany(fetch=FetchType.EAGER)
...
```

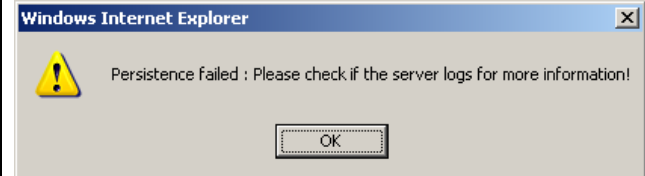
In `Item.java`:

```
...
@ManyToMany(mappedBy = "items",
fetch=FetchType.EAGER)
...
```

### The Seller function issue

The Petstore application is an online shop for pets. It allows for uploading data of a pet one wants to sell. This functionality is provided on the "Seller" page. Follow the instructions of the application in order to upload your pet's and seller's data. But even after providing all the necessary information, you will find that this functionality is broken.

The problem is again in the persistence layer of the application.



### The Seller function issue resolution

In this case the resolution is somewhat complex. On one hand we have to

- adjust the error prone application's source code and
- on the other hand adjust the JPA configuration (`persistence.xml`)

In the class

`com.sun.javaee.blueprints.petstore.model.CatalogFacade.java` there is the method `addItem(Item item)`. The source code changes required are highlighted in blue.

```
public String addItem(Item item) {
    EntityManager em =
    emf.createEntityManager();
    try {
        utx.begin();
        em.joinTransaction();
        Collection<Tag> mergedTags =
        new Vector<Tag>();
        for(Tag tag : item.getTags()) {
            tag.incrementRefCount();
            tag.getItems().add(item);
            Tag mergedTag = em.merge(tag);
            mergedTags.add(mergedTag);
        }
        item.setTags(mergedTags);
        em.persist(item);
        utx.commit();
        // index item
        if (bDebug)
            System.out.println("\n***Item id of
            new item is : " + item.getItemID());
            indexItem(new IndexDocument(item));
    } catch (Exception exe) {
        try {
            utx.rollback();
        } catch (Exception e) {}
        throw new RuntimeException("Error
        persisting item", exe);
    }
}
```

	<pre>     } finally {         em.close();     }     return item.getItemID(); } </pre>
<p>Adjust the <code>persistence.xml</code> (SAP specific JPA configuration). Add the property <code>"com.sap.jpa.execution.foreign-key-handling"</code> with the value <code>"topological"</code>.</p> <p>This is the file as it should look like (beyond the comments). The relevant parts are high lightened.</p>	<pre> &lt;?xml version="1.0" encoding="UTF-8"?&gt; &lt;persistence version="1.0" 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"&gt;     &lt;persistence-unit name="PetstorePu"&gt;         &lt;description&gt;Petstore Persistence Unit         &lt;/description&gt;         &lt;jta-data-source&gt;jdbc/PetstoreDB&lt;/jta- data-source&gt;         &lt;non-jta-data- source&gt;jdbc/PetstoreDB&lt;/non-jta-data- source&gt;         &lt;properties&gt;&lt;property name = "com.sap.jpa.execution.foreign-key- handling" value = "topological"/&gt;&lt;/properties&gt;     &lt;/persistence-unit&gt; &lt;/persistence&gt; </pre>
<p>Save the files, rebuild and redeploy the project.</p> <p>This is the result of uploading a pet with a <code>sunset.jpg</code> photo of it. The seller's name is "peter". This verifies the fix.</p>	 <p>The screenshot shows a web application interface with a light blue background. At the top, it says "peter, Thank you for submitting your pet sunset!". Below this, it says "Here's the uploaded photo of your pet" and displays a small image of a sunset. Further down, it asks "Would you like to :-" and shows three small thumbnail images of different pets (a dog, a cat, and a bird) with their respective details.</p>

This concludes the migration.



## Copyright

© Copyright 2010 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Excel, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, System i, System i5, System p, System p5, System x, System z, System z10, System z9, z10, z9, iSeries, pSeries, xSeries, zSeries, eServer, z/VM, z/OS, i5/OS, S/390, OS/390, OS/400, AS/400, S/390 Parallel Enterprise Server, PowerVM, Power Architecture, POWER6+, POWER6, POWER5+, POWER5, POWER, OpenPower, PowerPC, BatchPipes, BladeCenter, System Storage, GPFS, HACMP, RETAIN, DB2 Connect, RACF, Redbooks, OS/2, Parallel Sysplex, MVS/ESA, AIX, Intelligent Miner, WebSphere, Netfinity, Tivoli and Informix are trademarks or registered trademarks of IBM Corporation.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group.

Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.

JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

SAP, R/3, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP Business ByDesign, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and other countries.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius, and other Business Objects products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of Business Objects S.A. in the United States and in other countries. Business Objects is an SAP company.

All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.