

# Running JPA Applications with Hibernate as a Third-Party Persistence Provider on SAP NetWeaver CE 7.1 EHP1



## Applies to:

SAP NetWeaver Composition Environment (CE) 7.1 including enhancement package 1, and all later releases.

For more information, visit the [Java homepage](#).

## Summary

This article describes the necessary steps to run a JPA application with Hibernate as a 3<sup>rd</sup> party persistence provider. The main goal is to keep the benefits that the persistence container of SAP NetWeaver AS Java provides, and to change the underlying persistence provider only.

The article does not cover usage of native Hibernate APIs.

**Authors:** Peter Matov, Valentina Ivanova, Yordan Yordanov

**Company:** SAP Labs Bulgaria

**Created on:** 4 December 2009

## Authors' Bio



**Peter Matov**

Peter is an Expert Developer who has been working for 10 years in the Java EE world: JDBC, JTA, JCA, EJB, JPA, JMX.



**Valentina Ivanova**

Valentina is a Quality Assurance expert who is specialized in the overall AS Java area.



**Yordan Yordanov**

Yordan is an Information Developer working in the Java EE development and AS Java areas.

## Table of Contents

1. Introduction .....	3
2. Prerequisites .....	3
3. Setting up the JPA Application to run with Hibernate .....	3
3.1 Import/Open the existing JPA application in the SAP NetWeaver Developer Studio .....	3
3.2 Prepare the JPA Application to Use Hibernate as Persistence Provider .....	4
3.2.1 Make the Hibernate implementation “visible” to the application .....	4
3.2.2 Set a reference from your application to the Hibernate library. ....	4
3.2.3 Change the persistence provider in the persistence.xml of your JPA, EJB, or Web project .....	4
3.2.4 Configure the JTA DataSource in the persistence.xml .....	4
3.2.5 Specify the necessary properties .....	5
3.2.5.1 Database dialect .....	5
3.2.5.2 TransactionManager lookup class .....	5
3.2.6 Configure the DataSource to be used by your application .....	6
4. Deploy and Run the Application .....	7
5. Restrictions and Side Effects Imposed by the SAP’s JPA Implementation and Hibernate as the Persistence Provider .....	8
5.1 Application scope .....	8
5.2 Complex packaging .....	8
5.3 Byte code modifications .....	8
6. Related Content .....	9
Copyright .....	10

## 1. Introduction

This article describes the necessary steps to run a JPA application with Hibernate as a 3<sup>rd</sup> party persistence provider on a SAP NetWeaver system. The main goal is to keep the benefits that the persistence container of SAP NetWeaver AS Java provides, and to change the underlying persistence provider only.

The article does not cover usage of native Hibernate APIs.

## 2. Prerequisites

You need the following:

- SAP NetWeaver CE 7.1 including enhancement package 1 (here: SP2)
- SAP NetWeaver Developer Studio 7.1 (here: SP4)
- Library files of the desired distribution of Hibernate EntityManager (you can download it from <http://www.hibernate.org>, or use this one: <http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/60aa9a0a-e7be-2c10-6bab-bb083a505eee>) (here: 3.4.0 GA)

**Note:** In the general case, you can remove the `junit.jar`, `jta.jar`, and `ejb3-persistence.jar` from the library files of the Hibernate distribution.

- SAP Project Management and Employee Services application (also referred to as *example application*), running on the SAP JPA provider (you can download it from <http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/f06deb7e-e7be-2c10-74ba-8e7a7b77e152>)

**Note:** The referred `DefaultProvider~EDM.jar` is an enhanced version of the application which is installed with SAP NetWeaver CE 7.1 EHP1. These enhancements overcome some restrictions imposed by the Hibernate specifics, so that changing the persistence provider is facilitated.

For more information about how to setup and run the application, see

[http://help.sap.com/saphelp\\_nwce711/helpdata/en/64/fcdf4103c24d469e332c335eea3a2a/frameset.htm](http://help.sap.com/saphelp_nwce711/helpdata/en/64/fcdf4103c24d469e332c335eea3a2a/frameset.htm)

- Installed database (here: MS SQL Server 2005). Depending on the database vendor, `data-sources.xml` and `persistence.xml` should be configured accordingly.

## 3. Setting up the JPA Application to run with Hibernate

This section describes the common steps that you have to perform in order to run your JPA application with Hibernate. The code examples, however, are particular for the Project Management and Employee Services application.

### 3.1 Import/Open the existing JPA application in the SAP NetWeaver Developer Studio

1. In the Developer Studio, choose *File* → *Import*.
2. In the *Import* dialog, select *General* → *Existing Projects into Workspace*, then choose *Next*.
3. Select the *Select archive file* radio button, then choose the associated *Browse* button.
4. In the file system browser, locate and select the JPA application to import (for example, `DefaultProvider~EDM.jar`), then choose *Open*. The *Import* dialog shows all available projects within the selected application in the *Projects* area.
5. Select all projects, then choose *Finish*.

## 3.2 Prepare the JPA Application to Use Hibernate as Persistence Provider

### 3.2.1 Make the Hibernate implementation “visible” to the application

To do this, you need to add all Hibernate resources as “heavy resources” to the application.

**Note:** In the example application, the JAR files in the referred `Hibernate_Lib.zip` are used.

For more information, see:

<https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/60642a88-95fe-2b10-d387-a245d48fc257>

[http://help.sap.com/saphelp\\_nwce711/helpdata/en/44/f773f6e38e2462e10000000a1553f7/frameset.htm](http://help.sap.com/saphelp_nwce711/helpdata/en/44/f773f6e38e2462e10000000a1553f7/frameset.htm)

### 3.2.2 Set a reference from your application to the Hibernate library.

Open the `application-j2ee-engine.xml` deployment descriptor of your application and specify the reference:

```
<?xml version="1.0" encoding="UTF-8"?>
<application-j2ee-engine
  xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
  xsi:noNamespaceSchemaLocation="application-j2ee-engine.xsd">
  <reference reference-type="hard">
    <reference-target target-type="application"
      provider-name="com.sap">
      hibernate~library
    </reference-target>
  </reference>
  <fail-over-enable mode="on_request" scope="instance_local"
    xsi:type="fail-over-enableType_enable"/>
</application-j2ee-engine>
```

### 3.2.3 Change the persistence provider in the `persistence.xml` of your JPA, EJB, or Web project

```
<provider>org.hibernate.ejb.HibernatePersistence</provider>
```

**Note:** In the example application, the project that contains the entities is an EJB project, which is called `xapps-ra.edm.prjgmt.jee.ejb`.

### 3.2.4 Configure the JTA DataSource in the `persistence.xml`

```
<jta-data-source>MY_DATA_SOURCE</jta-data-source>
```

**Note:** For the example application, the DataSource name is `EDM_DS_ALIAS`.

### 3.2.5 Specify the necessary properties

#### 3.2.5.1 Database dialect

**Note:** The properties used in the code samples below might be a subject of change over time.

##### 3.2.5.1.1 MS SQL Server

If you are using MS SQL Server, add the following property in the `persistence.xml`:

```
<property name="hibernate.dialect" value="org.hibernate.dialect.SQLServerDialect" />
```

##### 3.2.5.1.2 SAP Max DB

In case you want to specify an SAP Max DB dialect, use the following code:

```
<property name="hibernate.dialect" value="org.hibernate.dialect.SAPDBDialect" />
```

#### 3.2.5.2 TransactionManager lookup class

```
<property name="hibernate.transaction.manager_lookup_class"
value="org.hibernate.transaction.SAPWebASTransactionManagerLookup" />
```

The `SAPWebASTransactionManagerLookup` class must be added in the heavy class loader classpath. To achieve this, you can add the class in some of the Hibernate JAR files or you can add it in a separate JAR file that will later be added in the heavy class loader resources.

**Note:** For the example application, the `SAPWebASTransactionManagerLookup` class is added under `hibernate-core.jar!/org/hibernate/transaction/SAPWebASTransactionManagerLookup.class`.

**Note:** For the example application, the `persistence.xml` file has the following content:

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

  <persistence-unit name="ems/EJB3_EDM_DEMO_PU_version2">
    <provider>org.hibernate.ejb.HibernatePersistence</provider>
    <jta-data-source>EDM_DS_ALIAS</jta-data-source>
    <properties>
      <property name="hibernate.dialect" value="org.hibernate.dialect.SQLServerDialect" />
      <property name="hibernate.transaction.manager_lookup_class"
        value="org.hibernate.transaction.SAPWebASTransactionManagerLookup" />
    </properties>
  </persistence-unit>
</persistence>
```

For a detailed description of the elements in the `persistence.xml`, refer to the Java Persistence API specification at Sun Microsystems' Web site <http://java.sun.com>.

For a detailed description of the Hibernate-specific persistence properties, refer to the Hibernate documentation.

### 3.2.6 Configure the DataSource to be used by your application

The default DataSource of SAP NetWeaver AS Java uses READ\_UNCOMMITTED isolation level, as well as Open SQL for portability purposes. However, the SQL generated by the Hibernate runtime is not Open SQL compatible, so you must define a custom DataSource to be used by your application.

In general, DataSources can be created automatically during application deployment, if you add and configure the data-sources.xml file under the META-INF folder of your EAR project.

**Note:** For the example application, the EDM\_DS\_ALIAS DataSource is created during deployment. The data-sources.xml is added to the EAR project (hibernate-xapps~ra.edm.prjmgmt.jee.app) and configured as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE data-sources SYSTEM "data-sources.dtd" >
<data-sources>
  <data-source>
    <data-source-name>EDM_DS_ALIAS</data-source-name>
    <driver-name>SYSTEM_DRIVER</driver-name>
    <isolation-level>TRANSACTION_READ_COMMITTED</isolation-level>
    <sql-engine>native_sql</sql-engine>

    <!-- Please change the driver settings, user and password with the appropriate values. -->
    <jdbc-1.x>
      <!-- Settings for MS SQL. If not in use should be commented. -->

      <driver-class-name>
        com.microsoft.sqlserver.jdbc.SQLServerDriver
      </driver-class-name>

      <url>
        jdbc:sqlserver://localhost:1433;databasename=LKG
      </url>

      <!-- Settings for SAP DB. Should uncomment them and comment the one for MS SQL. -->
      <!--
        <driver-class-name>
          com.sap.dbtech.jdbc.DriverSapDB
        </driver-class-name>
        <url>jdbc:sapdb://localhost/LKG</url>
        -->

        <user-name>SAPLKGDB</user-name>
        <password>abcd1234</password>
      </jdbc-1.x>
    </data-source>
  </data-sources>
```

**Note:** You must verify that the values of the elements <data-source-name>, <url>, <driver-class-name>, <user-name>, and <password> are accorded with the particular application environment.

For a detailed description of the elements in data-sources.xml, see [http://help.sap.com/saphelp\\_nwce711/helpdata/en/64/0bee3da7138e5be10000000a114084/content.htm](http://help.sap.com/saphelp_nwce711/helpdata/en/64/0bee3da7138e5be10000000a114084/content.htm)

**Note:** The original example application uses the system DataSource via a DataSource alias. To avoid naming conflicts, the data-source-aliases.xml was removed from the META-INF folder of the application project, and replaced with data-sources.xml.

The URL below refers to the example application configured to run with Hibernate as persistence provider.

<http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/30c7092c-e7be-2c10-5db4-905776d14f5b>

**Note:** If you decide to use the ready `HibernateProvider~EDM.jar` and import it in the Developer Studio, you must also re-import the external libraries of the Hibernate distribution you are using (`Hibernate_Lib.zip`).

## 4. Deploy and Run the Application

For detailed information on how to deploy and run the application, see

[http://help.sap.com/saphelp\\_nwce711/helpdata/en/ae/f0b1ee32a040d98b111915b23fb66e/frameset.htm](http://help.sap.com/saphelp_nwce711/helpdata/en/ae/f0b1ee32a040d98b111915b23fb66e/frameset.htm).

As an alternative to step 3 (*Deploy and Run the Application*), you can also do this by opening the context menu of the `index.jsp` file within the Web project, choosing *Run As* → *Run On Server*, then select the desired server and choose *Finish*.

**Note:** If, during deployment or execution of the application, some Hibernate classes cannot be found, it is possible that you have not added all the necessary binaries, or the heavy resources have not been added correctly. In the first case you have to ensure that all required JAR files have been added. In the second case you must check if the heavy class loader has been initialized properly. To do this, use the following procedure:

1. Open the Telnet Administrator console.
  - a. Open the SAP Management Console, then select the *AS Java Process Table* of your Application Server.
  - b. In the context menu of the ICM process, choose *AS Java Telnet*.
  - c. Login to the Application Server by providing valid Administrator username and password.
2. Execute the `llr <ApplicationName>` command, where `<ApplicationName>` is the name of the application that contains the Hibernate libraries.

```
> llr <ApplicationName>
Loader name:
  [ApplicationName]
CSN component:
  []
DC name:
  [ApplicationName]
Direct parent loaders:
```

```
  [ApplicationName-library-loader]
```

The `[ApplicationName-library-loader]` line shows that the heavy class loader has been created and properly set as the first parent class loader of the application class loader.

For the example application particularly, the output is as follows:

```
> llr com.sap/hibernate~library
Loader name:
  [com.sap/hibernate~library]
CSN component:
  []
DC name:
  [com.sap/hibernate~library]
Direct parent loaders:
```

```
  [com.sap/hibernate~library-library-loader]
```

If no `[ApplicationName-library-loader]` has been created and set as a first parent class loader of the application class loader, then carefully read and apply step 6 (*Define Heavy Resource's Specific Libraries*) in the following article: <https://www.sdn.sap.com/irj/scn/go/portal/prtroot/docs/library/uuid/60642a88-95fe-2b10-d387-a245d48fc257>

If the problem persists, you can open a support ticket at SAP's problem reporting system.

## 5. Restrictions and Side Effects Imposed by the SAP's JPA Implementation and Hibernate as the Persistence Provider

### 5.1 Application scope

- Entity mapping xml files could be found on the whole class path. However, according to the specification, the explicitly specified files should be searched only within the application, and the default `orm.xml` should be searched only within the persistence unit root or the JAR files referenced from the `persistence.xml`.
- The explicitly specified classes are searched in the application classpath. However, according to the specification, they should be searched only within the application.

### 5.2 Complex packaging

- References to root JAR files that do not contain `persistence.xml` cannot be created.

### 5.3 Byte code modifications

- The `ClassTransformer` interface is not supported.



## 6. Related Content

[Running Spring-Based Applications on SAP NetWeaver Composition Environment 7.1 and MS SQL Server 2005](#)

[Running Seam-based Applications on SAP NetWeaver Composition Environment 7.1](#)

For more information, visit the [Java homepage](#).

## Copyright

© Copyright 2009 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.