Java™ Persistence Architecture for WebSphere Application Server

William (Al) Gilchrist, Jr.
wgilchri@us.ibm.com
Software Engineer
WAS Level 2 Support

Jerry C. Poon
poonje@us.ibm.com
Software Engineer
WAS Level 2 Support

Heath Thomann
htomann@us.ibm.com
Software Engineer
WAS Level 3 Support
JPA

Date 06/10/2014

Agenda

- Introduction to Java Persistence Architecture (JPA™)
- Apache JPA
- JPA for WebSphere Application Server
- Versions of JPA for WebSphere Application Server
- JPQL™
- Persistence.xml
- Troubleshooting JPA.
- Tracing JPA for WebSphere Application Server

Introduction: Java Persistence API

- JPA is not a new technology. It collected the best ideas from existing persistence technologies and result is standardized specification. It provides an Object relational mapping approach, meaning that it can converting data between incompatible programming languages. JPA allows to define how to map Java objects to relational database tables. Object relational mapping approach with JPA is entirely metadata-driven and can be done either adding annotations to the code or using externally defined XML. The primary to use JPA is to programmatically build code to access data in a database.

7.6 9)

Apache OpenJPA

- JPA for WebSphere Application Server is built on the Apache OpenJPA open source project.
  - Apache OpenJPA
    - Apache OpenJPA is a the Sun Microsystems JPA specification. WebSphere Application Server employs extensions to provide additional features and utilities. Since the JPA for WebSphere Application Server is built from OpenJPA, all OpenJPA functionality, extensions and configurations are unaffected. Users running OpenJPA applications do not need to make any changes to use their applications in WebSphere Application Server.
    - The properties for OpenJPA can be defined in one of two ways. You can either specify the property in the persistence.xml file or by using a Java virtual machine (JVM™) command line argument on either client or server.

Example:

- Specify the OpenJPA property in the persistence.xml file.

```
<properties>
    <property name="openjpa.jdbc.SchemaFactory" value="native(ForeignKeys=true)"
/</properties>
```

- Specify the OpenJPA property using a JVM command line argument on the client or server.

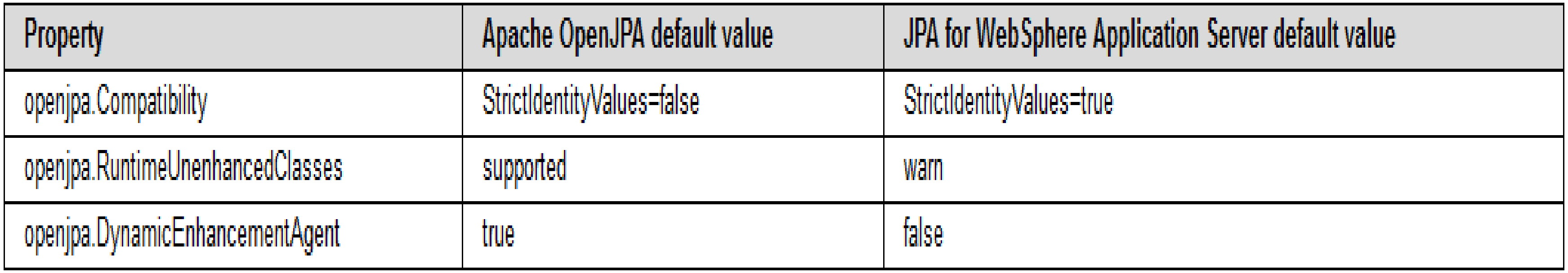
```
-Dopenjpa.jdbc.SchemaFactory="native(ForeignKeys=true)"
```

JPA for WebSphere Application Server

- JPA for WebSphere Application Server contains a set of tools for application development and deployment adding functionality:
  - Enhanced tracing support
  - Version ID generation
  - ObjectGrid cache plug-in support
  - WebSphere product-specific commands and scripts
  - Translated message files
  - Static SQL™ support using the DB2® pureQuery feature
  - Access intent support

Configuration Differences between WebSphere 7.0.0.x and Apache Open JPA 1.2.x

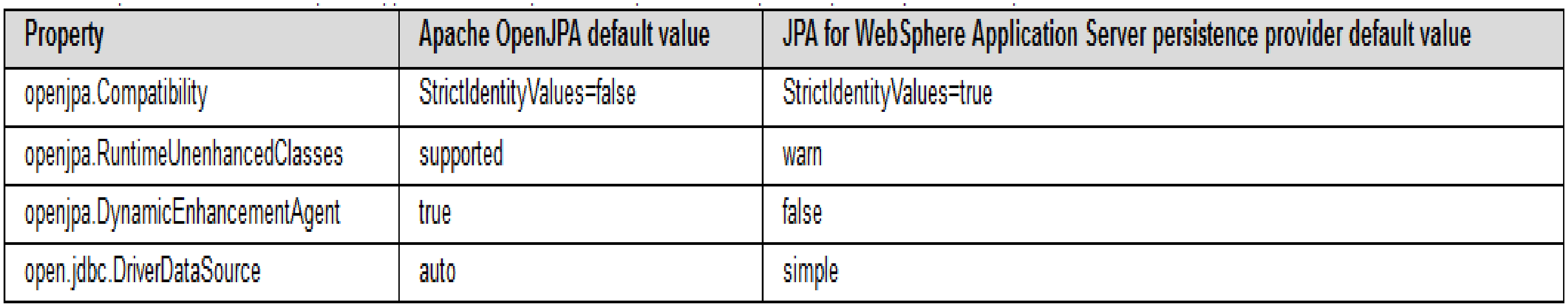
- The default values for the JPA for WebSphere Application Server provider configuration properties are different from the Apache OpenJPA provider:



Property	Apache OpenJPA default value	JPA for WebSphere Application Server default value
openjpa.Compatibility	StrictIdentityValues=false	StrictIdentityValues=true
openjpa.RuntimeUnenhancedClasses	supported	warn
openjpa.DynamicEnhancementAgent	true	false

Configuration Differences between WebSphere 8.0.0.x/ 8.5.x and Apache Open JPA 2.2.x

- The default values for the JPA for WebSphere Application Server persistence provider configuration properties are different from the Apache OpenJPA provider:



Property	Apache OpenJPA default value	JPA for WebSphere Application Server persistence provider default value
openjpa.Compatibility	StrictIdentityValues=false	StrictIdentityValues=true
openjpa.RuntimeUnenhancedClasses	supported	warn
openjpa.DynamicEnhancementAgent	true	false
open.jdbc.DriverDataSource	auto	simple

Versions of JPA used with WebSphere Application Server

- JPA Spec 1.0 implementation is/was shipped with EJB™ 3.0 Feature Pack for v6.1 and v7.0 via OpenJPA.

- - EJB 3.0 Feature pack for v6.1 ships OpenJPA version 1.0.x
- - v7.0 ships OpenJPA version 1.2.x

- JPA Spec 2.0 implementation is/was shipped with JPA 2.0 Feature pack , v8.0, v8.5.0, and v8.5.5.

- - JPA 2.0 Feature pack ships OpenJPA version 2.0.x
- - v8.0 ships OpenJPA version 2.1.x
- - v8.5.0 ships OpenJPA version 2.2.1.x
- - v8.5.5 ships OpenJPA version 2.2.x

- The Feature Pack for EJB 3.0 is an optionally installable product extension for IBM WebSphere Application Server Version 6.1 that implements the EJB Version 3.0 and Java Persistence API (JPA) 1.0 specification.

Functionality JPA provides

Java Persistence Architecture is a simplified programming model used for object relational mapping and data persistence.

Data persistence ensures that applications that interact with a databases have the ability to perform necessary data up-keep for the data contained in the database.

JPA simplifies this process by using Java representations of your database tables, called entities, and providing a set of APIs to persist and query data in the database.

JPA with Respect to WebSphere Application Server

- WebSphere Application server provides two ways to persist information to databases.

- Enterprise Java Beans Entity Beans (EJB). Less common way to persist today but were the normal way to persist prior to WAS version 6.1 Feature Pack and the JPA 1.0 specification. Was the only IBM provided way prior to JPA so many older programs still have this persistence method.

- Java Persistence architecture (JPA) which is most commonly used since version 6.1 EJB 3.0 Feature pack of the application server.

- **NOTE!** The application server will also allow you to use a third party persistence architecture. However, IBM does not provide support when a third party persistence provider is used. It is recommended that a user use the persistence provider shipped with WAS.

Java Persistence Query Language

- Java Persistence Query Language (JPQL) as per the tutorial found at the following URL: http://docs.oracle.com/javaee/6/tutorial/doc/bnbtg.html defines queries for entities and their persistent state.
- JPQL is very much like Structured Query Language (SQL) used for maintaining data within a database.
- JPQL is used to interact with the target database tables and produce the information needed based on the design of the query or to maintain the actual data in the database.

Persistence.xml

- JPA requires the use of an XML file called the "persistence.xml" that describes how to access the data.
- The XML file must be created in the META-INF directory.
- The JPA Specification (section 8.2 in the 2.0 spec version) defines the locations where the META-INF directory containing the persistence.xml must be located.

Persistence.xml (continued)

- To read a full description of the persistence.xml for JPA, please see the Apache OpenJPAV2.0 user's guide: Chapter 6. Look at the following URL for the manuals you can get: http://openjpa.apache.org/documentation.html
- In addition to the persistence.xml file, you can also create an orm.xml mapping file which contains mapping information for the classes listed in it. The information in this file can override annotations listed in an enitity class, or define information about a class in place of annotations.

Accessing a database with JPA

1) Create a JPA project in your IDE
2) Add a JDBC™ driver to your project for your DB or use a data source created in your IDE
3) Add a JPA file from the Apache site to the project
4) Modify the persistence.xml to have the user name and password for the DB in it.
5) Set up a small application with one entity.
6) Create a Java Bean to exercise the entity.
7) Run the driving application as a Java bean or a Java application.

Troubleshooting methods for JPA used by IBM Support.

When thinking of troubleshooting JPA in WAS, you have to break the effort down in to its parts and perform work to eliminate the identified parts. The part we will address in this presentation is only the JPA as the JPA does not have WAS enhanced tracing embedded as does the other WAS modules because JPA is open source and not entirely under IBM control.   JTS™,J2C™...etc that are not open source and that have to do with database access do have the WAS enhanced tracing built in and the advice from here is to use the enhanced tracing built in for troubleshooting those parts.

Troubleshooting methods for JPA used by IBM Support (continued)

For the JPA part of the application or issue, it is best to use an application that is not part of WAS first to see if the JPA works without any of the application server pieces.

Most of the time you can take the JPA entities and use case to replicate the failure outside of the J2EE™ environment (i.e. in a JSE™ unit test). This takes the 'noise' of an application server out of the issue and makes debugging far faster and easier. There are times when an issue is specific to the J2EE environment but more often than not a JSE environment suffices.

Producing a simple pared down test case that recreates the reported issue drastically speeds up problem resolution. If it is not possible to produce a test to recreate the issue, at least providing trace, your entities and mapping files such as persistence.xml does help in recreating and debugging the issue.

Troubleshooting methods for JPA (continued)

Now, speaking of the JPA part as is described in the previous slide, it is good to set up a Java SE program that uses the JDBC driver files to connect to the database and so forth to completely eliminate WAS as a part of the problem. This would include any data sources....etc. With all of the application server parts or “noise” eliminated, one can then focus directly on JPA and its abilities...etc and a determination can be made if the JPA part of the application is working properly outside of the application server.

Troubleshooting methods for JPA (continued)

- With a JPA project made to reproduce the problem in either eclipse, Rational® Application Developer or your favorite IDE that supports JPA, place in the entities that are part of your application that have the issue you are trying to troubleshoot.

- Also include some Java code or a Java bean that will exercise the entities based on your use case. In order to see if they will work with the database of your choice to produce the data you want to produce.

Troubleshooting methods for JPA (continued)

Once you have the above working, you should then run the application as a Java program or as a Java Bean in the IDE, or a Junit™ Test. This should produce some output that tells you if you are able to work with your database and produce the output you want.

Also note that the log you get while running as a Java app will expose more information at this time as the tracing for WAS is not as extensive in the Open JPA part as it is in other modules in the application server.

Troubleshooting methods for JPA (continued)

If you do reproduce the error you are trying to troubleshoot while running the app spoken of previously, I do suggest you attempt to see if you can find a JIRA™ that has been created for the particular error you see or error scenario. Look for the JIRA at the Apache OPENJPA site which is:

http://openjpa.apache.org/

To look-up in the JIRA area, move to the following:

https://issues.apache.org/jira/browse/OPENJPA#selectedTab=com.atlassian.jira.jira-projects-plugin:summary-panel

Troubleshooting methods for JPA (continued)

- If you do find a JIRA, please include the JIRA number in a PMR to IBM Support and Support will investigate based on the JIRA as the JIRA does have attachments...etc from the person who created the JIRA.

- If you do not find a JIRA that fits the problem you have, please describe you problem in depth in the PMR text to include the version of the application server on which you have had the problem and output from the wsjpaversion command.

- In all cases, please be prepared to send in the following information with your JPA PMR.
  - JPA entities
  - Simple application that will exercise the entities and show your issue.
  - Persistence.xml for the simple application.
  - If instructed, please be prepared to send in specified traces that are specified by support.

Troubleshooting methods for JPA (continued)

- The traces that will be needed for JPA PMRs will be as follows unless otherwise specified:

- *=info:JPA=all:openjpa=all

Websites of interest

The main site for Apache JPA is located at:
http://openjpa.apache.org/

You can find a tutorial for JPA that is located in the following location:

http://openjpa.apache.org/begin-using-openjpa---the-basics.html

JPQL:
http://docs.oracle.com/javaee/6/tutorial/doc/bnbtg.html

Summary

- Introduction to Java Persistence Architecture
- Versions of JPA for WebSphere Application Server
- JPQL
- Persistence.xml
- Use JPA to access a database.
- Troubleshooting JPA.

Additional WebSphere Product Resources

- Learn about upcoming WebSphere Support Technical Exchange webcasts, and access previously recorded presentations at: http://www.ibm.com/software/websphere/support/supp_tech.html

- Discover the latest trends in WebSphere Technology and implementation, participate in technically-focused briefings, webcasts and podcasts at: http://www.ibm.com/developerworks/websphere/community/

- Join the Global WebSphere Community: http://www.websphereusergroup.org

- Access key product show-me demos and tutorials by visiting IBM Education Assistant: http://www.ibm.com/software/info/education/assistant

- View a webcast replay with step-by-step instructions for using the Service Request (SR) tool for submitting problems electronically: http://www.ibm.com/software/websphere/support/d2w.html

- Sign up to receive weekly technical My Notifications emails: http://www.ibm.com/software/support/einfo.html

Connect with us!

1. **Get notified on upcoming webcasts**
   Send an e-mail to wsehelp@us.ibm.com with subject line “wste subscribe” to get a list of mailing lists and to subscribe
2. **Tell us what you want to learn**
   Send us suggestions for future topics or improvements about our webcasts to wsehelp@us.ibm.com

Questions and Answers