Support for Java2db in AS9.0

As of 11/21/2005

Currently we support java2db from within the application server environment. When the out of container case is supported this document would be updated to reflect it.

How to enable the java2db feature for a ejb 3.0 ear ?

To enable java2db for an ejb 3.0 ear define the following properties for a persistence unit descriptor in the persistence.xml.
"ddl-generation" ::: with possible values of "createtables"/"dropandcreate"/"none"
"create-ddl-jdbc-file-name" ::: (optional): <name of the create ddl file>
"drop-ddl-jdbc-file-name" ::: (optional): <name of the create ddl file>
"application-location" (For within a container in the code this value would be set to the app generated directory. This property would be used for the out of container case to define the location where one would want the jdbc ddl files to be stored.).

e.g. of a persistence.xml usage could be

```
<?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" >
   <persistence-unit name="em">
      <description>This is a test persistence.xml for java2db support</description>
      <properties>
         <property name="ddl-generation" value="dropandcreate"/>
         <property name="create-ddl-jdbc-file-name" value="create_ora.jdbc"/>
         <property name="drop-ddl-jdbc-file-name" value="drop_ora.jdbc"/>
      </properties>
   </persistence-unit>
</persistence>
```

How to enable java2db feature w/o changing the ejb3.0 ear ?

The easiest way to enable this feature w/o changing the application ear is by using the asadmin cli commands :
--createtables=true/false. Specified at deploy time to ensure that the tables are created in the database.
--dropandcreatetables=true/false. Specified at redeploy time to ensure that the old tables are dropped and new ones created at redeploy time.
--droptables=true/false. Specified at undeploy time to ensure that the tables are dropped from the database.

These are the same commands that work with our application server for cmp2.x beans. These options overwrite the setting of the properties that might be defined in the persistence.xml.

example :

1. Command to deploy an ejb3.0 application that does not contain the properties defined but still create the required tables in the database :
   $S1AS_HOME/bin/asadmin deploy --retrieve . --user admin --password adminadmin --host localhost --port 4848 --createtables=true --name onetomany --force=true / export/home/tools/ejb30_related/persistence-onetomanyApp.ear
2. Command to redeploy an ejb3.0 application that does not contain the properties and ensure that

the tables get recreated in the database :
$S1AS_HOME/bin/asadmin deploy --retrieve . --user admin --password adminadmin --host localhost --port 4848 --dropandcreatetables=true --name onetomany --force=true / export/home/tools/ejb30_related/persistence-onetomanyApp.ear

3. Command to undeploy the ejb3.0 application that does not contain the properties and ensure that the tables get dropped from the database :

$S1AS_HOME/bin/asadmin undeploy --user admin --password adminadmin --host localhost --port 4848 --droptables=true onetomany

How does the cli commands interact/affect the properties if they are defined in persistence.xml ?

The cli options (--createtables/--dropandcreatetables/--droptables) take precedence over the properties in the persistence.xml.

Lets take an example to further explain this :
The user has defined the following in their persistence.xml
ddl-generation : dropandcreate
and then when deploying the application has specified
--createtables=false.
We then create the jdbc ddl statements but not execute the ddl statements against the database. But if the --createtables option is not defined at the deploy time, we would create the jdbc ddl statements and also execute the ddls against the database.

What happens if the jdbc ddl file names are not specified in the persistence.xml ?

This situation is similar to the cmp2.x case and we end up create ddl files with some default names. We try to create the name of the file by prepending the application and/or module name followed by the persistence unit name and then a static string of the form "createDDL.jdbc" or "dropDDL.jdbc".

So you might see files with names like

default_em_dropDDL.jdbc and *default_em_createDDL.jdbc*
OR
realApp_war1_ejb_em1_dropDDL.jdbc and *realApp_war1_ejb_em1_createDDL.jdbc*
(this is case where the persistence.xml is defined in a war file)

Do all the asadmin command options for java2db defined for AS8.x work as is with AS9.0 ?

Not at this point of time. There are options like –uniquetablenames and –dbvendorname that can be defined for AS8.x and are currently not supported for AS9.0. These options might be supported in the future and this document would be updated to reflect the same.