Creating deployment plans for enterprise applications

{scrollbar}

An enterprise application archive (EAR) can consist of many sub modules. The sub modules can be web modules (WAR), ejb modules (JAR), resource adapter modules (RAR) or application client modules (`jar`). INLINE

When an EAR consist of many sub modules, the deployment plans for all the sub modules can be provided in a single file named `geronimo-application.xml`.

This single file contains the deployment details of each of the sub modules of the EAR. Alternatively, each of the sub modules can package its corresponding deployment plan file within itself. However, the preferable way is to provide a single deployment plan through `geronimo-application.xml` for all the sub modules. This mechanism provides flexibility of allowing us to modify the deployment configuration for all modules through a single file. In this section, we explore EAR deployment plan and understand what it contains.

There are 3 places a deployment plan (partial) can be associated with an ear, and they are used in this order:

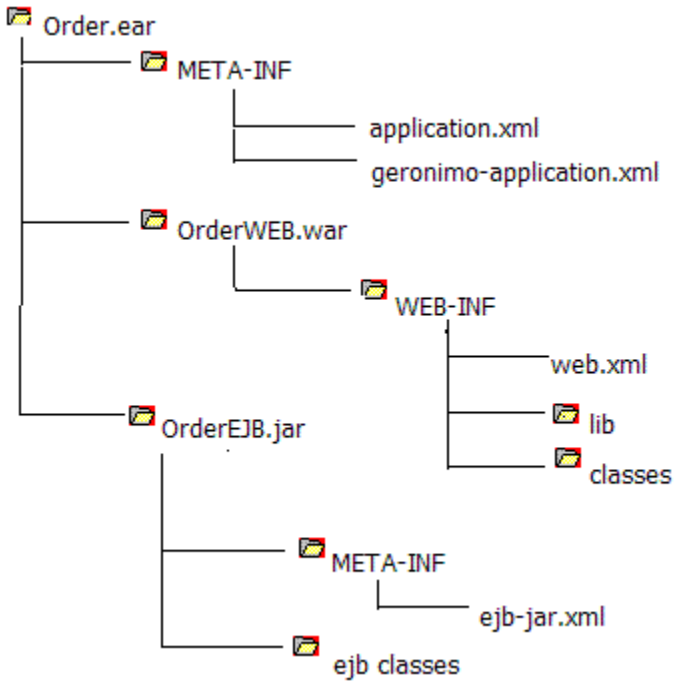
1. external plan to be specified at deployment time
2. ear level geronimo-application.xml
3. module level geronimo|openejb-*.xml

So, anything in an external plan takes precedence over bits in (2) or (3) configuring the same module. Anything in (2) takes precedence over a module level plan. If a module level plan is missing from (1) its looked for in (2),then (3); if missing from (1) and (2), then its looked for in the module (3).

There is no attempt to merge plans from these different locations: e.g. if there's something in (1) for a module, any other plans are ignored.

An enterprise application archive (EAR) should provide its deployment descriptor in the application.xml file. The `application.xml` lists all the sub modules in the EAR file along with the descriptions. In addition to the standard deployment descriptor, the EAR should also provide Geronimo specific deployment plan in geronimo-application.xml. Along with the description of each of the sub modules of the EAR file, this file also provides mappings for JEE resources that each of the sub modules refers in their deployment descriptor. The `geronimo-application.xml` is divided into several sections where in each section, the deployment plan for a sub module is provided. `geronimo-application.xml` is the highest level plan that provides deployment plan for all sub modules; hence it can contain XML elements from every other Geronimo XML schema used by Geronimo application deployer. The `geronimo-application.xml` is the super set of all other deployment plans.

For example, following is the structure of an EAR that has a web module and an ejb module.



The `Order.ear` file shown above contains two modules. One is `OrderWEB.war` file which is a web module and the other is `OrderEJB.jar` file which is an ejb module. The `META-INF` folder in `Order.ear` contains the application deployment descriptor (`application.xml`) and the Geronimo application deployment plan (`geronimo-application.xml`). The web application and the ejb application have packaged only their respective deployment descriptors. But the deployment plans for these modules are provided in the `geronimo-application.xml`.
The web application (`OrderWAR.war`) looks up stateless session bean in the `OrderEJB.jar` module to retrieve the order information. The `RetrieveOrderInfoBean` in `OrderEJB.jar` module uses JDBC connection to read the order information from a DB2 database.

The deployment descriptor of the OrderEJB.jar is as follows.

ejb-jar.xmlsolid <?xml version="1.0" encoding="UTF-8" ?> <ejb-jar xmlns="http://java.sun.com/xml/ns/javaee" version="3.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/ejb-jar_3_0.xsd"> <description>Stateless Session Bean Example</description> <display-name>Stateless Session Bean Example</display-name> <enterprise-beans> <session> <ejb-name>RetrieveOrderInfoBean</ejb-name> <business-local> examples.session.stateless_dd.RetrieveOrderInfo </business-local> <ejb-class> examples.session.stateless_dd.RetrieveOrderInfoBean </ejb-class> <session-type>Stateless</session-type> <transaction-type>Container</transaction-type> <resource-ref> <res-ref-name>jdbc/DB2DataSource</res-ref-name> <res-type>javax.sql.DataSource</res-type> <res-auth>Container</res-auth> <res-sharing-scope>Shareable</res-sharing-scope> </resource-ref> </session> </enterprise-beans> <interceptors> <interceptor> <interceptor-class> examples.session.stateless_dd.RetrieveOrderCallbacks </interceptor-class> <post-construct> <lifecycle-callback-method>construct</lifecycle-callback-method> </post-construct> <pre-destroy> <lifecycle-callback-method>destroy</lifecycle-callback-method> </pre-destroy> </interceptor> </interceptors> <assembly-descriptor> <interceptor-binding> <ejb-name>RetrieveOrderInfoBean</ejb-name> <interceptor-class> examples.session.stateless_dd.RetrieveOrderCallbacks </interceptor-class> </interceptor-binding> </assembly-descriptor> </ejb-jar>
The default namespace of the above XML document is `http://java.sun.com/xml/ns/javaee`. The XML elements that do not have a namespace prefix belong to the default namespace.

In the `RetrieveOrderInfoBean`, the following code is used to look up the Datasource object and obtain a database connection.

JAVAexamples.session.stateless_dd.RetrieveOrderInfoBeansolid ... ... Context initContext = new InitialContext(); Context envContext = (Context) initContext.lookup("java:comp/env"); DataSource ds = (DataSource)envContext.lookup("jdbc/DB2DataSource"); System.out.println("Got DataSource\n"); con = ds.getConnection(); System.out.println("Got Connection\n"); ... ...

The deployment descriptor of the `OrderWEB.war` is as follows.

web.xmlsolid <?xml version="1.0" encoding="UTF-8"?> <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="WebApp_ID" version="2.5"> <display-name>OrderWEB</display-name> <welcome-file-list> <welcome-file>index.html</welcome-file> <welcome-file>index.htm</welcome-file> <welcome-file>index.jsp</welcome-file> <welcome-file>default.html</welcome-file> <welcome-file>default.htm</welcome-file> <welcome-file>default.jsp</welcome-file> </welcome-file-list> <servlet> <description></description> <display-name>RetrieveOrder</display-name> <servlet-name>RetrieveOrder</servlet-name> <servlet-class> examples.web.servlet.RetrieveOrder </servlet-class> </servlet> <ejb-local-ref> <ejb-ref-name>ejb/RetrieveOrderInfo</ejb-ref-name> <ejb-ref-type>Session</ejb-ref-type> <local> examples.session.stateless_dd.RetrieveOrderInfo </local> <ejb-link>RetrieveOrderInfoBean</ejb-link> </ejb-local-ref> <servlet-mapping> <servlet-name>RetrieveOrder</servlet-name> <url-pattern>/RetrieveOrder</url-pattern> </servlet-mapping> </web-app>
The default namespace of the above XML document is `http://java.sun.com/xml/ns/javaee`. The XML elements that do not have a namespace prefix belong to the default namespace.

In the `RetrieveOrder` servlet, the following code is used to look up the ejb to retrieve the order details.

JAVAexamples.web.servlet.RetrieveOrdersolid ... ... Context ctx = new InitialContext(); System.out.println("Instantiating beans..."); retrieveOInfo = (RetrieveOrderInfo)ctx.lookup("java:comp/env/ejb/RetrieveOrderInfo"); String orderIdStr = request.getParameter("orderid"); int orderId = Integer.parseInt (orderIdStr); OrderInfo oInfo = retrieveOInfo.getOrderInfo(orderId); ... ...

The deployment descriptor of the Order.ear is as follows.

application.xmlsolid <?xml version="1.0" encoding="UTF-8"?> <application xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xmlns:application="http://java.sun.com/xml/ns/javaee/application_5.xsd" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/j2ee/application_5.xsd" version="5"> <description>EAR Example</description> <display-name>Order Sample</display-name> <module> <web> <web-uri>OrderWEB.war</web-uri> <context-root>/OrderDemo</context-root> </web> </module> <module> <ejb>OrderEJB.jar</ejb> </module> </application>
The default namespace of the above XML document is `http://java.sun.com/xml/ns/javaee`. The XML elements that do not have a namespace prefix belong to the default namespace.

The deployment plan of the `Order.ear` is as follows.

geronimo-application.xmlsolid <?xml version="1.0" encoding="UTF-8"?> <application xmlns="http://geronimo.apache.org/xml/ns/j2ee/application-2.0" xmlns:sys="http://geronimo.apache.org/xml/ns/deployment-1.2" application-name="Order"> <sys:environment> <sys:moduleId> <sys:groupId>Order</sys:groupId> <sys:artifactId>OrderEAR</sys:artifactId> <sys:version>5.0</sys:version> <sys:type>car</sys:type> </sys:moduleId> </sys:environment> <module> <web>OrderWEB.war</web> <web-app xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-2.0.1" > <sys:environment> <sys:moduleId> <sys:groupId>Order</sys:groupId> <sys:artifactId>OrderWEB</sys:artifactId> <sys:version>2.5</sys:version> <sys:type>war</sys:type> </sys:moduleId> </sys:environment> <context-root>/OrderDemo</context-root> </web-app> </module> <module> <ejb>OrderEJB.jar</ejb> <openejb-jar xmlns="http://openejb.apache.org/xml/ns/openejb-jar-2.2" xmlns:naming="http://geronimo.apache.org/xml/ns/naming-1.2"> <sys:environment> <sys:moduleId> <sys:groupId>Order</sys:groupId> <sys:artifactId>OrderEJB</sys:artifactId> <sys:version>3.0</sys:version> <sys:type>jar</sys:type> </sys:moduleId> <sys:dependencies> <sys:dependency> <sys:groupId>console.dbpool</sys:groupId> <sys:artifactId>OrderDS</sys:artifactId> <sys:version>1.0</sys:version> <sys:type>rar</sys:type> </sys:dependency> </sys:dependencies> </sys:environment> <enterprise-beans> <session> <ejb-name>RetrieveOrderInfoBean</ejb-name> <naming:resource-ref> <naming:ref-name>jdbc/DB2DataSource</naming:ref-name> <naming:resource-link>OrderDS</naming:resource-link> </naming:resource-ref> </session> </enterprise-beans> </openejb-jar> </module> </application>
The default namespace of the above XML document is `http://geronimo.apache.org/xml/ns/j2ee/application-2.0`. The XML elements that do not have a namespace prefix belong to the default namespace.

Observe how the JEE 5 resource names and ejb names in `ejb-jar.xml` and `web.xml` are mapped to actual resources deployed in the server through `geronimo-application.xml`.

As we can observe from the `geronimo-application.xml`, the deployment plans for web and ejb modules are wrapped in `<module> .... </module>` elements. The xml elements used to provide deployment plan for the web module are from the schema http://geronimo.apache.org/xml/ns/j2ee/web-2.0.1 which is the schema for `geronimo-web.xml`. Similarly, the xml elements used to provide ejb deployment plan are from the schema http://openejb.apache.org/xml/ns/openejb-jar-2.2 which is the schema for `openejb-jar.xml`. Hence, `geronimo-application.xml` borrows elements from all other schemas to provide deployment plan for its sub modules.

Also, observe that, in the `geronimo-application.xml`, along with moduleId configuration for the EAR itself, there is a moduleId configuration for each web and ejb modules. If the above EAR file deployed on the server and the configurations are listed, the following output would be displayed on the console.

solid C:\Geronimo-2.1\bin>deploy.bat --user system --password manager list-modules Using GERONIMO_BASE: C:\Geronimo-2.1 Using GERONIMO_HOME: C:\Geronimo-2.1 Using GERONIMO_TMPDIR: var\temp Using JRE_HOME: C:\sun\jre Found 92 modules + Order/OrderEAR/5.0/car `-> OrderWEB.war `-> OrderEJB.jar + console.dbpool/OrderDS/1.0/rar

The moduleId `Order/OrderEAR/5.0/car` is the configuration for the `Order.ear`. The ejb module declares a dependency on the `console.dbpool /OrderDS/1.0/rar` configuration in `<sys:dependencies>` section. This is the moduleId of the database pool that connects to the DB2 database where the order details are stored.