Creating deployment plans for Web applications

Creating deployment plans for Java Persistence API | Creating deployment plans for applications | Naming (JNDI)

In the `geronimo-web.xml` file, application deployer maps the security roles, ejb names, database resources, JMS resources, etc. declared in `web.xml` to corresponding entities deployed in the server. In addition to that, if there are any web container specific configurations, such as Tomcat or Jetty specific, depending on the application needs, all these settings are configured as well here. If the web application depends on any third party libraries or other services running in the server, all these dependencies are declared in the plan. Some web applications require class loading requirements different from the default class loading behavior. The `geronimo-web.xml` allows application deployer to configure this as well. There are many more configurations that could be done through `geronimo-web.xml` depending on the needs of web application. The following sections briefly explain how `geronimo-web.xml` can be used to configure the web container and web applications.

The `geronimo-web.xml` uses XML elements from http://geronimo.apache.org/xml/ns/j2ee/web-2.0.1 namespace and one or more namespaces mentioned in Configuring resources in the application scope section earlier in the document.

- Sample plan for a Web application
- Container specific configuration in a web application
  - The container-config element

Sample plan for a Web application

For example, the following `web.xml` and `geronimo-web.xml` are the deployment descriptor and Geronimo deployment plan respectively, of a web application that connects to a datasource deployed on DB2 and retrieves data from a table.

web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
         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/web-app_2_5.xsd" version="2.5">

    <resource-ref>
        <res-ref-name>jdbc/DataSource</res-ref-name>
        <res-type>javax.sql.DataSource</res-type>
        <res-auth>Container</res-auth>
        <res-sharing-scope>Shareable</res-sharing-scope>
    </resource-ref>

    <welcome-file-list>
        <welcome-file>jsp/EMPdemo.jsp</welcome-file>
    </welcome-file-list>
</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.

With Servlet 2.5 specification, many of the declarations done through `web.xml` can also be done through corresponding annotations in the servlets and JSPs. When both annotations and web.xml are provided, the declarations in web.xml takes precedence over annotations.

The web module connects to back end datasource using its JNDI name `jdbc/DataSource` as declared in the `web.xml`.

In `geronimo-web.xml`, a <dependency> element is added to make the the resource archive (RAR) visible to your application, and then a mapping is provided in <resource-ref>.

geronimo-web.xml

```
<web-app xmlns="http://geronimo.apache.org/xml/ns/j2ee/web-2.0.1"
          xmlns:naming="http://geronimo.apache.org/xml/ns/naming-1.2"
          xmlns:sec="http://geronimo.apache.org/xml/ns/security-2.0"
          xmlns:sys="http://geronimo.apache.org/xml/ns/deployment-1.2">

    <sys:environment>
        <sys:moduleId>
            <sys:groupId>samples</sys:groupId>
            <sys:artifactId>EmployeeDemo</sys:artifactId>
            <sys:version>2.5</sys:version>
            <sys:type>war</sys:type>
        </sys:moduleId>

        <sys:dependencies>
            <sys:dependency>
                <sys:groupId>samples</sys:groupId>
                <sys:artifactId>EmployeeDatasource</sys:artifactId>
                <sys:version>2.5</sys:version>
                <sys:type>rar</sys:type>
            </sys:dependency>
        </sys:dependencies>
    </sys:environment>

    <context-root>/EmployeeDemo</context-root>
        <naming:resource-ref>
        <naming:ref-name>jdbc/DataSource</naming:ref-name>
        <naming:resource-link>jdbc/EmployeeDatasource</naming:resource-link>
    </naming:resource-ref>
</web-app>
```


The default namespace of the above XML document is `http://geronimo.apache.org/xml/ns/j2ee/web-2.0.1`. The XML elements that do not have a namespace prefix belong to the default namespace.

Observe the various XML tags and corresponding namespaces used in the deployment plan for various purposes.

`<sys:environment> .. </sys:environment>` : These elements provide the moduleid configuration and the dependencies. The moduleId elements provide the configuration name for the web module. So, when the web module is deployed, it is given the configuration name `samples/EmployeeDemo/2.5/war`. The dependencies elements provide the configurations and third party libraries on which the web module is dependent on. These configurations and libraries will be available to the web module via a classloader hierarchy. In this case, the web module is dependent on `samples/EmployeeDatasource/2.5/rar` which is the configuration of the deployed Datasource that connects to a back end DB2 database. The Datasource deploys a database connection pool (`javax.sql.Datasource`) with name `jdbc/EmployeeDatasource`.

`<sys:context-root> .. </sys:context-root>` : The XML elements used to provide the web context root of the web applications.

`<naming:resource-ref> .. </naming:resource-ref>` : These elements help us to configure the resource references. In this case, the datasource reference `jdbc/DataSource` is mapped to `jdbc/EmployeeDatasource`.

In the `EMPdemo.jsp`, the following java code snippet is used to obtain a connection from the datasource. It reference the resource from the `java:comp/env` context entries.

EMPdemo.jsp

```
....
....
Context initContext = new InitialContext();
Context envContext  = (Context)initContext.lookup("java:comp/env");
DataSource ds = (DataSource)envContext.lookup("jdbc/DataSource");
Connection con = ds.getConnection();
....
....
```

The above descriptor and the plan files are the simple illustrations that explain how web modules are developed and assembled for Apache Geronimo. Similarly, many other configurations can be performed in the `geronimo-web.xml`.

All the XML schema files are located at `<geronimo_home>/schema` directory. Please go through the `.xsd` files to have a feel of XML tags that can be used in `geronimo-web.xml` for configuring web applications.

Container specific configuration in a web application

In some cases, you will need to provide container specific settings for your Web application. The WASCE server only includes the Tomcat Web container so only Tomcat specific settings are shown here.

The container-config element

The **<container-config>** element is used to hold container specific Web application settings using the following format. None of the Tomcat configuration elements are required, but if specified, they must follow the order in the example below.

```
<container-config>
    <tomcat xmlns="http://geronimo.apache.org/xml/ns/web/tomcat/config-1.0">
        <!-- Begin Tomcat configuration elements -->
        <host>host</host>
        <cross-context/>
        <disable-cookies/>
        <valve-chain>valve</valve-chain>
        <listener-chain>listener</listener>
        <tomcat-realm>realm</tomcat-realm>
        <manager>manager</manager>
        <cluster>cluster</cluster>
        <!-- End Tomcat configuration elements -->
    </tomcat>
</container-config>
```

where

- *host* is the name of a host defined in the *name* parameter of the *initParams* attribute of a virtual host **<gbean>** where the Web application will be deployed. See the Configuring virtual host reference for more information on configuring a virtual host.
- **<cross-context/>** indicates that this Web application will dispatch requests to other applications and this function should be enabled. Do not specify this tag if you want to disable cross context dispatch requests.
- **<disable-cookies/>** indicates that this Web application will not use cookies and this function should be disabled. Do not specify this tag if you want to enable cookies.
- *valve* is the name of a <gbean> that defines the first valve in a valve chain. See the valve chain reference for more information on defining valve chains.
- *listener* is the name of a <gbean> that defines the first listener in an lifecycle event listener chain. See the lifecycle listener reference for more information on defining lifecycle listener chains.
- *realm* is the name of a Tomcat realm **<gbean>** to be used for authentication and authorization for this Web application. If unspecified, the server will use the realm defined in the Tomcat configuration.
- *manager* is the name of a Tomcat manager **<gbean>** to be used to manage the Web application resources.
- *cluster* is the name of a Tomcat cluster **<gbean>** where the Web application will be deployed. See the Clustering and farming reference for more information on defining a Tomcat cluster.