RED HAT CONSULTING

redhat

Oracle Cartridge EWS 2.0 Datasource Test App Guide

PREPARED FOR - FRIT

Table of Contents

1. Synopsis

The purpose of this document is to present the steps required to create a OSE2 application using the Red Hat JBoss Enterprise Web Server (EWS) 2.0 and Oracle cartridges that will make a simple connection call to the newly provisioned Oracle tenant to confirm successful provisioning and fuctionality of the cartridges.

It is assumed that the JBoss EWS cartridge has been modified to support the Oracle datasource as outlined in the 'FRIT-ORACLE-Cartridge-EWS2-Datasource' document.

2. Create OSE Application

Using the Red Hat OpenShift Client Tools (RHC), create a application instance of EWS2 with the Oracle add-on cartridge, similar to the command below. Replace 'myapp' and 'mynamespace' with relevant values.

```
rhc app create -a myapp  -n mynamespace -t tomcat7 -t frb-oracle-12.0

Using jbossews-2.0 (Tomcat 7 (JBoss EWS 2.0)) for 'tomcat7'

Application Options
-------------------
Domain:     mynamespace
Cartridges: jbossews-2.0
Gear Size:  default
Scaling:    no

Creating application 'myapp' ... done

A instance has successfully be configured on the Oracle Database. Please make note of
these credentials:

 Script Result: SUCCESS@@oraclesrvr001@@1521@@tcdb_001
      Username: adminin5LXMY
      Password: 5XHV5JB2XteQ
     Tenant ID: tcdb_001

Waiting for your DNS name to be available ... done

Cloning into 'myapp'...
Warning: Permanently added 'myapp-mynamespace.example.com,54.0.0.193' (RSA) to the
list of known hosts.

Your application 'myapp' is now available.

  URL:        http://myapp-mynamespace.rhcloud.com/ (1)
  SSH to:     541a13a04382ec00000002f3@myapp-mynamespace.example.com
  Git remote: ssh://541a13a04382ec00000002f3@myapp-
mynamespace.example.com/~/git/myapp.git/

Run 'rhc show-app myapp' for more details about your app.
```

Once the application has been created, using git, clone the application to your local machine using the Git remote URL returned in the output from the RHC command.

```
git clone ssh://541a13a04382ec00000002f3@myapp-
mynamespace.example.com/~/git/myapp.git/
```

Change your working directory to the WebShift application '{git repo}' directory.

```
cd ./myapp
```

3. Modify EWS2 Configuration Files

Open the '{git repo}/.openshift/config/catalina.properties' and modify the 'common.loader to include '${cataline.home}/webapps,${catalina.home}/webapps/*.jar'

```
# List of comma-separated paths defining the contents of the "common"
# classloader. Prefixes should be used to define what is the repository type.
# Path may be relative to the CATALINA_HOME or CATALINA_BASE path or absolute.
# If left as blank,the JVM system loader will be used as Catalina's "common"
# loader.
# Examples:
#     "foo": Add this folder as a class repository
#     "foo/*.jar": Add all the JARs of the specified folder as class
#                  repositories
#     "foo/bar.jar": Add bar.jar as a class repository
common.loader=${cataline.home}/webapps,${catalina.home}/webapps/*.jar,${catalina.base}
/lib,${catalina.base}/lib/*.jar,${catalina.home}/lib,${catalina.home}/lib/*.jar
```

4. Create the Test Application

Create a directory to house the test application under the '{git repo}/webapps' directory

```
mkdir ./webapps/DSTester
```

Create a file named '{git repo}/webapps/DSTester/testOracleDS.jsp' in this newly created application directory. Place within it the following and save

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<sql:query var="rs" dataSource="jdbc/OracleDS">
SELECT SYSDATE FROM DUAL
</sql:query>

<html>
  <head>
    <title>DB Test</title>
  </head>
  <body>

  <h2>Results</h2>
     Successfully confirmed connection to "jdbc/OracleDS" via a "SELECT SYSDATA FROM
DUAL" query<br/>
     <c:forEach var="row" items="${rs.rows}">
          SYSDATE Returned from query: ${row.SYSDATE}<br/>
     </c:forEach>
  </body>
</html>
```

That JSP page makes use of JSTL's SQL and Core taglibs. You can get it from Apache Tomcat Taglibs - Standard Tag Library project (http://tomcat.apache.org/taglibs/standard/) — just make sure you get a 1.1.x or later release. Once you have them completed place them in to the '{git repo}/webapps/DSTester'' directory.

Create a 'WEB-INF' directory within this application directory

```
mkdir ./webapps/DSTester/WEB-INF
```

Create a 'web.xml' file within the newly created 'WEB-INF'directory with the following contents

```
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">
  <description>SQL Test App</description>
  <resource-ref>
      <description>DB Connection</description>
      <res-ref-name>jdbc/OracleDS</res-ref-name>
      <res-type>javax.sql.DataSource</res-type>
      <res-auth>Container</res-auth>
  </resource-ref>
</web-app>
```

Add all the newly created files to the git repo, commit, and push.

```
git add --all
git commit -am "commit message here"
git push
```

5. Use Test Application

Once the application has been git pushed, port-forward the application

```
rhc port-forward -a myapp -n mynamespace
```

Go to the following link, and you should see results, if there is a Apache error returned then either there there is a problem in the configuration.

```
http://127.0.0.1:8080/DSTester/testOracleDS.jsp
```

6. Reference Information

- OpenShift Oracle Cartridge
- How to configure datasource settings in EAP 6
- Apache Tomcat 7 JNDI Datasource HOW-TO