Oracle Utilities Customer Self Service

Whitepaper:

Creating and Deploying the Sample Mobile Client Application

Release 2.2.0.0

E78239-01

November 2016



Oracle Utilities Customer Self Service Whitepaper: Creating and Deploying the Sample Mobile Client Application

Release 2.2.0.0

E78239-01

November 2016

Copyright © 2011, 2016, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Table of Contents

Overview	4
Additional Resources	5
Acronyms	5
Supported Environments	5
Platforms	5
Software Requirements	5
Eclipse	5
iOS	
Android	
Third Party Software	
Environment Setup	9
Setting Up Eclipse	9
Deploying the Web Application to a Server	
Setup for iOS Native App	
Setup for Android Native App	
Sample Project	
Architecture	
MobileWebApp	
Utility JavaScript Code	
index.js	
Anatomy of a CSS Mobile Page	
HTML	
Page Structure (Snippet)	
JavaScript	
Public REST Services	19

Chapter 1

Overview

This guide describes the steps necessary for the deployment of a sample Mobile Web Application for Oracle Utilities Customer Self Service (OUCSS). This guide also describes the steps required to generate native apps for various mobile platforms using the sample Mobile Web Application and Apache Cordova.

The sample Web Application has the following features:

- Alerts
- Account Summary
- Enroll
- Financial History
- Make Payment
- Outage Map
- Report Outage
- Register
- Service Charges to Date
- Usage Overview
- View Bill

The sample Mobile Web Application uses jQuery Mobile and calls REST services that are part of OUCSS 2.2.0.0 to fetch and update data from CSS application. The Web Application can be accessed from a Mobile/Desktop browser.

The sample Mobile Web Application can be modified as needed in accordance with the client environment.

The sample application MobileWebApp2.2.0.0.zip can be downloaded from <u>Oracle Utilities Customer Self Service</u> section on the Oracle Technology Network (OTN) web site (<u>http://docs.oracle.com/cd/E72219_01/documentation.html</u>) Refer to the OUCSS 2.2.0.0 Installation and Implementation Guides for instructions on installation and configuration of the REST services, OUCSS Shared libraries which the REST services reference, and the InboundServices application, which is required to access the REST Services.

There are several fixes to REST Services which are part of Patch 24501061 (available from the Oracle Software Delivery Cloud at <u>edelivery.oracle.com</u>), This patch must be applied for features in this sample application to work correctly.

The Enroll and Register features in this sample Mobile Web Application do *not* support the CSS-CCB User Linking feature which was introduced in the OUCSS 2.2 Portal application.

Additional Resources

Resource	Location
JQuery	http://jquery.com
JQuery Mobile	http://jquerymobile.com
jqPlot	http://www.jqplot.com
Apache Cordova	http://cordova.apache.org
Oracle Utilities Customer Self Service Installation Guide	Check Oracle Utilities Customer Self Service on the Oracle Technology Network (OTN) web site (<u>http://docs.oracle.com/cd/E78536_01/index.htm</u>) for the latest version of the documents.
Oracle Utilities Customer Self Service Implementation Guide	Check Oracle Utilities Customer Self Service on the Oracle Technology Network (OTN) web site (<u>http://docs.oracle.com/cd/E78536_01/index.htm</u>) for the latest version of the documents.

Acronyms

OUCSS - Oracle Utilities Customer Self Service iOS – Apple mobile operating system Android – Google mobile operating system

Supported Environments

Platforms

Apache Cordova supports several platforms including iOS, Android, Blackberry, Windows, Tizen. This whitepaper covers steps related to iOS and Android platforms.

Refer to the *Platform Guides section of Apache Cordova Documentation*(<u>http://cordova.apache.org</u>) - for the list of supported platforms and versions.

Software Requirements

The following are the software requirements for generating native apps for iOS and Android.

Eclipse

Download Eclipse from <u>http://www.eclipse.org/downloads/</u>. Download a package which enables creation of Java EE and web applications (e.g., Eclipse IDE for Java EE Developers).

iOS

- A computer running Mac OS.
- Xcode.
- iOS SDK. (See the "iOS Platform Guide" section of the *Apache Cordova Documentation* at <u>http://cordova.apache.org</u> for information regarding the supported versions for Xcode and iOS SDK.).
- An Apple Developer ID.

Android

Android SDK and its tools (see http://developer.android.com/sdk/) for supported versions.

Third Party Software

The sample mobile application depends on the following third party software. Please download it from the suggested websites and place it in the folders specified.

CSS files should be placed under www/css folder



JavaScript files should be placed in the www/js folder.

- D B META-INF D D WEB-INF a 😓 www CSS b 🗁 custom b 🗁 images 🔺 🗁 js Index.js jqplot.canvasAxisLabelRenderer.min.js \triangleright jqplot.canvasAxisTickRenderer.min.js jqplot.canvasTextRenderer.min.js jqplot.categoryAxisRenderer.min.js jqplot.cursor.min.js jqplot.dateAxisRenderer.min.js \triangleright jqplot.highlighter.min.js jqplot.json2.min.js \triangleright jqplot.pointLabels.min.js jquery-1.9.1.js jquery.jqplot.min.js jquery.mobile-1.4.5.min.js jquery.ui.map.js b > > mods b 🗁 res 🔊 Login.html
 - Iogin.js
 - 📄 springboard.html

The following files are referenced in the sample project:

- JQuery (<u>http://www.jquery.com</u>)
 - jquery-1.9.1.js
- JQuery Mobile (<u>http://jquerymobile.com/</u>)
 - jquery.mobile.structure-1.4.5.css
 - jquery.mobile.structure-1.4.5.min.css
 - jquery.mobile.theme-1.4.5.css
 - jquery.mobile.theme-1.4.5.min.css
 - jquery.mobile-1.4.5.css
 - jquery.mobile-1.4.5.min.css
 - jquery.mobile-1.4.5.min.js
- jqPlot (http://www.jqplot.com/) version 1.0.8r
 - jqplot.canvasAxisLabelRenderer.min.js
 - jqplot.canvasAxisTickRenderer.min.js
 - jqplot.canvasTextRenderer.min.js
 - jqplot.categoryAxisRenderer.min.js
 - jqplot.cursor.min.js

- jqplot.dateAxisRenderer.min.js
- jqplot.highlighter.min.js
- jqplot.json2.min.js
- jqplot.pointLabels.min.js
- jquery.jqplot.min.js
- jquery.jqplot.min.css
- Google Maps (<u>http://code.google.com/p/jquery-ui-map/)</u>
 - jquery.ui.map.js
- Apache Cordova
- Android OS and Android SDK
- iOS and Xcode

Chapter 2

Environment Setup

This section describes the settings and requirements for a successful installation of the Web Application and for generating native mobile apps.

Setting Up Eclipse

- 1 Download Eclipse.
- 2 Extract the contents of MobileWebApp2.2.0.0.zip into a local folder.
- 3 Make sure all Third Party software is downloaded and copied to the respective folders as described in <u>Third Party</u> <u>Software</u>.
- 4 Create a workspace in Eclipse.
- 5 Import the MobileWebApp project into the Workspace: File Menu > Import > General > Existing Projects into Workspace.



- 6 Select the location of the expanded MobileWebApp directory, then click Finish to create a project.
- 7 Expand the WebContent > www > custom folder and provide the URL for secured and public REST services in config.js as follows:



- 8 Save the files.
- **9** OUCSSRegisterService is a secured REST service. To access it without authentication, the REST service must be made public; for the procedure, see Chapter 4, <u>Public REST Services</u>).

Deploying the Web Application to a Server

To deploy the Web Application to a server and access it from a desktop/mobile browser:

- 1 Right click on the MobileWebApp project, then choose Export > WAR File.
- **2** Specify the location of the war file.

Note: The Web Application should be deployed in the same server as OUCSS REST services to avoid Cross-Origin Resource Sharing (CORS) issues.

- 3 Copy the war file to the OUCSS server where REST services are deployed. The war file should be copied to the Admin server Upload folder. e.g., xxx/user_projects/domains/xxx_domain/servers/AdminServer/upload.
- **4** From the WebLogic Administrator console choose Deployments > Install.

Customize this table

Dep	reployments							
Ins	tall Update Delete						Showing 1 to 53	of 53 Previous Next
	Name 🗞	State	Health	Туре	Targets	Scope	Domain Partitions	Deployment Order
	adf.oracle.businesseditor(1.0,12.2.1.0.0)	Active		Library	DefaultServer	Global		100
	efadf.oracle.domain(1.0,12.2.1.0.0)	Active		Library	DefaultServer	Global		100
	adf.oracle.domain.webapp(1.0,12.2.1.0.0)	Active		Library	DefaultServer	Global		100
	Scoherence-transaction-rar	Active	🖋 ок	Resource Adapter	DefaultServer	Global		100
	Com.oracle.ugbu.ss.commercial.lib(2.2,2.2.0.0.0)	Active		Library	DefaultServer	Global		100
	Com.oracle.ugbu.ss.lib(2.2,2.2.0.0.0)	Active		Library	DefaultServer	Global		100
	Com.oracle.ugbu.ss.residential.lib(2.2,2.2.0.0.0)	Active		Library	DefaultServer	Global		100
	Com.oracle.ugbu.ss.rest.lib(2.2,2.2.0.0.0)	Active		Library	DefaultServer	Global		100
	DMS Application (12.2.1.0.0)	Active	🖋 ок	Web Application	DefaultServer	Global		5
	extend.oucss.portal(2.2,2.2.0.0.0)	Active		Library	DefaultServer	Global		100
	▲jax-rs(2.0,2.21.1.0)	Active		Library	DefaultServer	Global		100
	isf(2.0,1.0.0.0_2-2-8)	Active		Library	DefaultServer	Global		100
	istl(1.2,1.2.0.1)	Active		Library	DefaultServer	Global		100
	🗄 🐻 MobileWebApp	Active	🖋 ок	Web Application	DefaultServer	Global		100
	odl.clickhistory(1.0,12.2.1)	Active		Library	DefaultServer	Global		100
	odl.clickhistory.webapp(1.0,12.2.1)	Active		Library	DefaultServer	Global		100
	et ohw-rcf(5,12.2.1.0.0)	Active		Library	DefaultServer	Global		100
	no hw-uix(5,12.2.1.0.0)	Active		Library	DefaultServer	Global		100
	🖻 🐻 opss-rest	Active	🖋 ок	Web Application	DefaultServer	Global		150

- **5** Select MobileWebApp.war, then click Next.
- 6 Select Install this deployment as an application, then click Next.
- 7 Select the server on which the OUCSS REST services are deployed as Target.
- 8 Click Finish.

The MobileWebApp should be deployed successfully.

9 Access the Web App from a desktop/mobile browser as follows: http://server:port/MobileWebApp/www/Login.html

nttp.//server.port/MobilewebApp/www/Login.ntml

Setup for iOS Native App

- 1 Ensure you have the following installed and running:
 - A computer running Mac OS.
 - Xcode. After installing Xcode you must run it at least once and complete the Apple licensing and setup dialogs.
 - iOS SDK. (See the "iOS Platform Guide" section of the *Apache Cordova Documentation* at <u>http://cordova.apache.org</u> for information regarding the supported versions for Xcode and iOS SDK.)
- **2** Create an Apple developer ID.
- **3** Install Cordova by downloading Node.js. (For detailed instructions, see the "Command Line Interface" section of the *Apache Cordova Documentation* at <u>http://cordova.apache.org</u>.)
- 4 Create a Cordova project as follows. xxx below depends on the Apple developer ID configuration. cordova create CSSMobileWebApp com.xxx. MobileWebApp CSSMobileWebApp cd CSSMobileWebApp
- 5 Copy the www folder from the sample MobileWebApp project to CSSMobileWebApp.

6 Add platform support for iOS as follows:

```
cordova platform add ios cordova build
```

7 Open the project in Xcode. Double-click to open CSSMobileWebApp\platforms\ios\CSSMobileWebApp.xcodeproj.

For detailed instructions on generating a native app for iOS, see the "iOS Platform Guide" section of the *Apache Cordova Documentation* at <u>http://cordova.apache.org</u>.

Setup for Android Native App

- 1 Ensure you have the following:
 - A computer running Windows, Linux, or Mac OS.
 - Android SDK with Platform 2.3 or later, and the corresponding tools, installed on the OS.
- 2 Download Eclipse from http://www.eclipse.org/downloads/. Download a package download which enables creation of Java EE and web applications e.g. Eclipse IDE for Java EE Developers.
- 3 Download and install the Android SDK from http://developer.android.com/sdk/index.html.
- **4** Install Eclipse and then install the Android development tools (ADT) plug-in as given in the instructions at http://developer.android.com/sdk/installing/installing-adt.html.
- **5** For the Apache Cordova, Eclipse and Android SDK setup follow instructions in Android Platform Guide section of Apache Cordova Documentation(http://cordova.apache.org).
- 6 Create a Cordova project as follows.

cordova create CSSMobileWebApp com.xxx. MobileWebApp CSSMobileWebApp
cd CSSMobileWebApp

- 7 Copy the www folder from the sample MobileWebApp project to CSSMobileWebApp.
- 8 Change the content in config.xml to point to Login.html.



9 Add platform support for Android using the following commands:

cordova platform add android cordova build

10 Open the project in Eclipse and deploy it to Emulator or an Android device.

Chapter 3

Sample Project

The sample Web Application described in this section (available for download from the Oracle Utilities Customer Self Service documentation section on the Oracle Technology Network at http://www.oracle.com/technetwork/apps-tech/utilities/documentation/index.html) invokes OUCSS REST services to fetch/update data. The sample project can be modified using the Eclipse IDE.

Note: See <u>Third Party Software</u> for information on the software required for the development and building of the Sample mobile app.

۵

Architecture

MobileWebApp

D > > META-INF
WEB-INF
a 🍃 www
👂 🗁 CSS
👂 🗁 custom
👂 🗁 images
> 🗁 js
a 📂 mods
> 🗁 acctList
> 🔝 billing
> 🔝 enroll
> 🐎 outage
> 🔝 pymt
> 📂 register
b > b servicecharges
> 🗁 usage
👂 🗁 res
🔏 Login.html
Iogin.js
📄 springboard.html

- Login.html is the first page launched. This page is used to authenticate and to store credentials for future logins.
- springboard.html has the menu structure.
- js/index.js has the following base utility functions:
 - The js folder has all the required javascript files for jQuery, jQuery Mobile, cordova, other plugins like jQPlot, etc.
 - The mods folder contains html and javascript files for specific pages.
 - The css folder contains css files for jQuery Mobile and OUCSS.

Utility JavaScript Code

index.js

Base JavaScript with all the utility functions that are loaded when the CSS App starts. The data is retained for the lifetime of the application.

- OUCSS. Utilities: Has the utility functions to log to the console.
- OUCSS.Storage: Functions to store data to Local and Session storage per the HTML5 specification. This specification only supports key-value pairs. Local storage is persistent and will be retained until the user deletes the cache. Session storage is per session. Supported by all browsers and Mobile OSs. (Not used. Web storage is used instead.)
- setTheme: To change the theme for the current page.
- OUCSS.CacheMgr: Store Lookups and labels per session. This is used to get the data for Labels and Lookups.
- OUCSS.AJAX: Used to make AJAX calls to REST services.
- OUCSS.DB: Functions using the Web SQL feature to store Labels and Lookups.
 - Local persistent database implemented by browser and OS.
 - Supported by Safari, Chrome, iOS, Android.
 - This Util function creates the database if it does not exist, calls REST services to read Labels and Lookups, stores them in Web SQL, and populates OUCSS.CacheMgr.
 - In future invocations of the app, if data already exists, this function will read data from the database and populate OUCSS.CacheMgr.
- OUCSS.CSSApp: Core functions to handle Page context, page rendering, and form submit.
 - get/set PageContext.
 - get/set GlobalContext.
 - pageBeforeChange: Called when moving from one page to another. Used to call a custom page instead of the base page. Also passes page parameters.
 - renderPage: Render the page based on the "oucss-" tags on the elements. Supports table, div, select, label, input.
 - pageBeforeShow:
 - o Displays Labels and Lookups based on "oucss-label" and "oucss-lookup" tags.
 - Invoke the REST service to load the data and call renderPage. If the page has the same function implemented as OUCSS.PageEvents.pageId.pageBeforeShow, then call that function instead.
 - If a custom JS function exists in the form of OUCSS.PageEvents.pageId.cmext.pageBeforeShow, then call that function.
 - pageSubmitForm:
 - If a custom JS function exists in the form of OUCSS.PageEvents. **pageId**.cmext.pageSubmitForm, then call that function.
 - Submit the form to the REST service based on the "oucss-service" tag on the page. If the page has implemented the same function, then call the page-specific function instead.

Anatomy of a CSS Mobile Page

HTML

• Every jQuery Mobile page should have divs with data-roles of page, header, content, and footer.

- Custom "oucss-" tags can be used to link the UI elements with the data:
 - "oucss-service" tag on the page element to indicate the name of the REST service for getting the page data.
 - "oucss-path" tag on any element represents the relative path of the data element in the REST service's output.
 - "oucss-label" tag represents the label name.
 - "oucss-lookup" tag represents the lookup name.

Page Structure (Snippet)

```
<div oucss-form="paymentForm" data-role="page" id="makePaymentPage" data-mini="true" class="pageCLs">
```

```
<script src="payments.js"></script>
<script src="payments_cmExt.js"></script>
<div data-role="header" data-position="fixed" class="cssheader">
   <div class="ui-btn-left" data-role="controlgroup">
       >
               <a data-icon="grid" data-iconpos="notext"
                   data-role="button" href=".././springboard.html" data-rel="back">Menu</a>
               Menu
           </div>
   <h1>
       <span class="headerLargeFont">Pay Now</span>
   </h1>
</div>
<div data-role="content" id="payment-content" class="card">
   <div>
       <form name="paymentForm" id="paymentForm" method="post"
           data-ajax="false">
           <div class="center-wrapper">
               <span class="textLargeFont"><b id="acctInfoPay">Account
                      : </b></span>
           </div>
           <div data-role="fieldcontain" id="paymentInfoDiv"</pre>
               class="ui-hide-label">
               <label for="paymentType" class="select"
                   oucss-label="AUTOPAY PAYMENT TYPE LBL" data-mini="true">Pvmt
                   Type</label> <select data-theme="d" name="md paymentType"
                   oucss-path="mainData.paymentType" id="paymentType"
                   oucss-lookup="PAYMENT TYPES" data-mini="true" class="dropDownTxt">
               </select> <label for="paymentAmount" oucss-label="AMOUNT_LBL">Amt</label> <input
                   type="text" oucss-path="mainData.paymentAmount"
                   name="md paymentAmount" id="paymentAmount" value=""
                   data-mini="true" placeholder="Amount ($)" />
           </div>
```

JavaScript

- Page-specific code to bind actions for buttons, etc.
- OUCSS.PageEvents.pageId: Functions can be written for specific Page Events for non-standard code.
 - **pageBeforeShow**: Called instead of the base function in index.js. Can be used to have page specific code to call the REST service and render the page.
 - **pageShow**: Called after jQuery enhances the page. Can be used to have page-specific code to call the REST service to get data and to change the page layout.
 - **pageSubmitForm**: Called instead of base function in index.js. Can be used to have page-specific code to submit the page.

Chapter 4

Public REST Services

All OUCSS REST services are secured by an OWSM policy. If a REST service must be accessed without authentication (e.g., OUCSSRegisterService in the sample app) then that REST service can be made public as described in the following procedure.

1 Open JDeveloper and create a new application by selecting **New** and choosing the **Fusion Web Application**(**ADF**) template as shown in the following image:

his list is filtered according to the currer	t project's <u>selected technologies</u> .	
00 Search)	
Categories:	Items:	Show All Description
∃General	Generic Application	
Connections	Application from EAR File	
·····Deployment Descriptors ·····Deployment Profiles	C Application Template	
	Fusion Web Application (ADF) Creates a databound ADF web ap project for the view and controller Flows), and another project for the) pplication. The application consists of one r components (ADF Faces and ADF Task ne data model (ADF Business Components).
Unit Tests	🔁 Java Desktop Application	
Business Tier ADF Business Components	🔁 Java Desktop Application (ADF)	
Data Controls	🔁 Java EE Web Application	
Web Services	🔁 SOA Application	
·····SOA Tier	B WebCenter Portal Framework Apr	plication

2 Provide an appropriate name and directory for the application in the wizard. Follow the wizard, retaining all default values to completion.

lame your project				5
Application Name Project 1 Name	Project Name: Model Directory:	P	PublicRestServices\Model	Bro <u>w</u> se
Project 1 Java Settings Project 2 Name Project 2 Java Settings	Project Technologies <u>Available:</u> <u>ADF Desktop Integrat</u> ADF Faces ADF Library Web Appl ADF Mobile Browser ADF Page Flow ADF Swing Ant Database (Offline) EJB <u>Technology Description</u>	ication Support	Associated Libraries gelected: ADF Business Components Java	
	Application Developm governs interaction b	ent Framework (Oracle ADF etween the rest of the app	 ADF Business Componentication and the data store 	ents din the V

Name your project			HOIDTOTOTOTOTOTOTOTO	5
Application Name Project 1 Name	Project Name: ViewControlle Directory:	PublicRest	tServices\ViewController	Bro <u>w</u> se
Project 1 Java Settings	Project Technologies Ge	enerated Components	Associated Libraries	
Project 2 Name	<u>Available</u> :	<u>S</u>	elected:	
	ADF Desktop Integration ADF Library Web Application ADF Mobile Browser ADF Swing Ant Database (Offline) EJB JavaBeans	on Support	ADF Page Flow ATML lava ISF ISP and Servlets AML	
	ADF Faces adds very high personalization and skinnir support, client-side valida	quality components, a ng capabilities. ADF Fac tion, partial rendering o	dialog framework, as well ces features include: file up of a page (AJAX-style), da	as pload ta tables;

3 Verify that the application in JDeveloper has the following structure:



4 To begin the process of creating the weblogic.xml descriptor, right-click on the WEB-INF folder and select New. In the next window, select Deployment Descriptors > Weblogic Deployment Descriptor.

> New Gallery				
All Technologies Current Project Technol This list is filtered according to the current p Search Categories: Applications Connections	Items:	Show All Descriptions		
	Crade Deployment Descriptor taglib.tld (JSP Tag Library Descriptor) WebLogic Deployment Descriptor Launches the Create WebLogic Deployr select the specific WebLogic descriptor To enable this option, you must select a Application Navigator.	r ment Descriptor wizard, in which you type and version you wish to create. a project or a file within a project in the		
Web Tier Applet HTML JSF JSP Servlets MI Items V				
Help		OK Cancel		

5 Select **weblogic.xml**, as shown in the following image.

🐣 Create WebLogic Deplo	yment Descriptor - Step 1 of 4	x
Select Descriptor	0101010101010101010101010105510	
Select Descriptor	Select the deployment descriptor you wish to create: persistence-configuration.xml plan.xml weblogic-appclient.xml weblogic-application.xml weblogic-cmp-rdbms-jar.xml weblogic-ejb-jar.xml weblogic-jdbc.xml weblogic-jdbc.xml weblogic-index.xml weblogic-regord agnostics.xml weblogic.xml Description: Creates the Oracle WebLogic platform-specific deployment descriptor weblogic.xml, populated with default deploy-time settings. To enable this option, you must select a project or a file within a project in the Application Navigator. You may have only one weblogic.xml deployment descriptor in a project.	on
Help	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish Ca	incel

6 Follow the wizard, retaining all defaults, then press Finish to create the weblogic.xml descriptor.

Oreate WebLogic Deploy	yment Descriptor - Step 3 of 3	
Summary	010101010101010101010101010	C
Select Descriptor. Select Version Summary	You have completed creating your new deployment descriptor. When you click Finish, the wizard will create the following deployment descriptor: weblogic.xml 12.1.3	
Help	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish Cancel	

- 7 Select the Libraries tab in the weblogic.xml and add the following entries to refer to the OUCSS shared libraries:
 - com.oracle.ugbu.ss.lib
 - extend.oucss.portal
 - com.oracle.ugbu.ss.rest.lib
 - jax-rs 2.0

					?
Application Coherence	Libraries			4	×
Container	Library Name	Specification Version	Implementation Version	Match Version Exactly	•
Directories	com.oracle.ugbu.ss.lib			false	
JSP	extend.oucss.portal			false	
Libraries	com.oracle.ugbu.ss.rest.lib			false	
References	jax-rs	2.0		false	1.
Message					
Security					
Servlets					
Session					
Work Priority					
Asynchronus					

8 Create a RESTful service invoking OUCSSRegisterService with these sample Java files:



```
GenericApplication.java
```

```
package view;
import java.util.HashSet;
import java.util.Set;
import javax.ws.rs.ApplicationPath;
import javax.ws.rs.core.Application;
@ApplicationPath("public")
public class GenericApplication
  extends Application
ł
 public Set<Class<?>> getClasses()
    Set<Class<?>> classes = new HashSet<Class<?>>();
    // Register feature classes.
    classes.add(org.glassfish.jersey.jackson.JacksonFeature.class);
    // Register OUCSS Register Service Resource.
    classes.add(OUCSSPublicRegisterService.class);
    // Register provider classes.
    classes.add(OUCSSJacksonJsonProvider.class);
   return classes;
  }
3
```

OUCSSPublicRegisterService.java

OUCSSJacksonJsonProvider.java

```
package view;
import javax.ws.rs.ext.ContextResolver;
import com.fasterxml.jackson.annotation.JsonInclude.Include;
import com.fasterxml.jackson.databind.MapperFeature;
import com.fasterxml.jackson.databind.ObjectMapper;
public class OUCSSJacksonJsonProvider
  implements ContextResolver<ObjectMapper>
 private static final ObjectMapper MAPPER = new ObjectMapper();
  static
   MAPPER.setSerializationInclusion(Include.NON_EMPTY);
   MAPPER.disable(MapperFeature.USE_GETTERS_AS_SETTERS);
  }
 public OUCSSJacksonJsonProvider()
  ł
    super();
  }
 @Override
 public ObjectMapper getContext(Class<?> type)
  ł
    return MAPPER;
  }
3
```

9 Right-click the ViewController project, then choose Project properties. Set the Web Context Root of the application to an appropriate name (e.g, OUCSSRestPublic) as used in the sample application. This context will be used to access the public REST service.

Project Properties - C:\spl\SS\2	.1.0.2\portlet_applications\Services\PublicRestService:	s\ViewController\ViewControll
(Search	Java EE Application	
Project Source Paths M	 Use <u>C</u>ustom Settings Use Project Settings 	Customi <u>z</u> e Settings
ADF Model ADF Task Flow ADF View ADF View Compiler Dependencies Deployment EJB Module Extension Javadoc Java EE Application JSP Tag Libraries JSP Visual Editor Libraries and Classpath Resource Bundle Run/Debug/Profile Technology Scope	• Use Project Settings The following properties are used when running this proin the integrated WLS server. Java EE Web Application Name: PublicRestServices-ViewController-webapp Java EE Web Context Root: OUCSSRestPublic Integrated WLS Command Line: \${jwn} \${java.options}]	pject as a Java EE module or application
Help		OK Cancel

10 Deploy the Application to an EAR file from JDeveloper. Deploy this EAR file to the WebLogic server using the server console through Deployments -> Console, as described in Deploying the Web Application to a Server.

Deploy PublicRestServices_application1	
Deployment Action	0101010101010101010101010
Deployment Action	Select a deployment action from the list below. Deploy to Application Server Deploy to EAR Creates an Enterprise Archive (EAR) file for the platform defined in the Profile.
Help	< <u>Back</u> <u>Next</u> <u>Finish</u> Cancel

11 OUCSSRegisterService can be accessed using the URL

http://host:port/OUCSSRestPublic/public/OUCSSRegisterService.