Oracle Real-Time Scheduler Mobile Application Installation and Deployment Guide Release 2.2.0.3

Copyright © 2000, 2015 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>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Installing the Android Targets</td>
<td>A-5</td>
</tr>
<tr>
<td>Proxy Settings</td>
<td>A-6</td>
</tr>
<tr>
<td>Node js</td>
<td>A-6</td>
</tr>
<tr>
<td>Git</td>
<td>A-6</td>
</tr>
<tr>
<td>Android SDK</td>
<td>A-7</td>
</tr>
<tr>
<td>Mac OS X</td>
<td>A-7</td>
</tr>
</tbody>
</table>
This guide describes how to install the mobility module of Oracle Real-Time Scheduler. This preface contains these topics:

- **Audience**
- **Related Documents**
- **Conventions**
- **Acronyms and Definitions**

## Audience

This guide is intended for system administrators installing the mobility module of Oracle Real-Time Scheduler.

To complete this installation, you should have:

- Experience installing and configuring development environment for mobile applications, application servers and other software.
- Administrative privileges on the host where you are installing, building and deploying the software.

## Related Documents

The following is the complete set of documentation available with this release.

### Installation, Configuration, and Release Notes

- Oracle Real-Time Scheduler Release Notes
- Oracle Real-Time Scheduler Quick Install Guide
- Oracle Real-Time Scheduler Server Installation Guide
- Oracle Real-Time Scheduler Mobile Application Installation and Deployment Guide (HTML5-based)
- Oracle Real-Time Scheduler DBA Guide
- Oracle Real-Time Scheduler Configuration Guide

### User Guides

Implementation and Development
  • Oracle Real-Time Scheduler Mobile Application Implementation and Development Guide (HTML5-based)

Map Editor Installation and User Guides
  • Oracle Real-Time Scheduler Map Editor User’s Guide
  • Oracle Real-Time Scheduler Map Editor Installation Guide

Framework Guides
  • Oracle Utilities Application Framework v4.2.0.2 Business Process Guide
  • Oracle Utilities Application Framework v4.2.0.2 Administration Guide
  • Oracle Utilities Application Framework v4.2.0.2 Release Notes

Supplemental Documents
  • Oracle Real-Time Scheduler Server Administration Guide
  • Oracle Real-Time Scheduler Batch Server Administration Guide
  • Oracle Real-Time Scheduler Security Guide

Conventions
The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>boldface</td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td>italic</td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td>monospace</td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>

Acronyms and Definitions
The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ORS</td>
<td>Oracle Real-Time Scheduler</td>
</tr>
<tr>
<td>WLS</td>
<td>Weblogic Server</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Development Kit</td>
</tr>
<tr>
<td>NPM</td>
<td>Node Package Manager</td>
</tr>
<tr>
<td>CLI</td>
<td>Apache Cordova Command Line Interface</td>
</tr>
<tr>
<td>OUI</td>
<td>Oracle Universal Installer</td>
</tr>
<tr>
<td>iOS</td>
<td>Apple mobile operating system</td>
</tr>
<tr>
<td>Convention</td>
<td>Meaning</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Android</td>
<td>Google mobile operating system</td>
</tr>
</tbody>
</table>
This chapter includes:

- Installation Overview

**Installation Overview**

Follow these steps to install, build and deploy the Oracle Real-Time Scheduler mobile application:

1. Install and configure Oracle Real-Time Scheduler server application as described in *Oracle Real-Time Scheduler Server Application Installation Guide*.

2. Review the mobile application architecture as described in *Oracle Real-Time Scheduler Mobile Application Implementation and Development Guide*.

3. Understand the requirements and supported build and runtime platforms for installing, building and deploying the application as described in the section *Build Environments vs Runtimes*.

4. Review and prepare for the download and installation of required Oracle and third-party software as described in the section *Pre-requisite Software*.

   **Note**: Please review proxy settings setup as described in Appendix *Configurations*.

5. Plan your installation and prepare configuration properties for project and plug-ins as described in Appendix *Configurations*.

6. Determine the type of installation and follow the instructions in the chapter *Installing the Oracle Real-Time Scheduler Mobile Application* corresponding to that type of installation.

7. Setup project and build the mobile application. Follow the instructions in the chapter *Building the Mobile Application* corresponding to the selected runtime platform.

8. Deploy the mobile application. Follow the instructions in the chapter *Deploying the Mobile Application* corresponding to the selected runtime platform.
Chapter 2

Supported Platforms and Hardware Requirements

This chapter includes:

- Supported Environments and Platforms
- Hardware Requirements
- Pre-requisite Software
- Build Environments vs Runtimes
Supported Environments and Platforms

The mobile application, based on HTML5, is supported for iOS, Android as well as web-based deployment.

Hardware Requirements

The following are the hardware requirements for the mobile application:

**iOS**

iOS devices running iOS 8.1.x

**Android**

Android device running Android 4.4 or greater

**Application Server**

For information on the application servers supported by this release, refer to the section “Operating Systems and Application Servers” in the Oracle Real-Time Scheduler Server Installation Guide.

Pre-requisite Software

The following software must be installed and configured prior to installation of the mobile application of Oracle Real-Time Scheduler:

- Node.js (v0.10.29 or above) a platform built on Chrome’s JavaScript runtime for building fast, scalable network applications.
- Git (v1.9.4) a repository which offers all of the distributed revision control and source code management functionality in order to have the mobile standard plugins.
- Apache Ant (v1.9.4) or above for automating the software build processes.
- Ant contrib (v1.0b2)
- Java Development Kit (v1.6 or above) with Java Standard Edition for compiling the Ant tasks and builds.
- Cordova (v3.6.3-0.2.13) is a mobile development framework for enabling programmers to develop applications in HTML5, Javascript and CSS3 instead of relying on platform-specific APIs like those in iOS or Android. Cordova enables wrapping up of HTML, CSS and Javascript code depending upon the platform of the device.
- Weblogic 11g for deploying the HTML5 applications as the war file
- Android SDK for creating new applications for the Android operating system. The recommended level is Android SDK Level 19 (Android version of 4.4.2).
- XCode 6.1 or greater and Apple Developer ID for iOS 8.x devices for developing the applications using iOS SDK.
- jQuery (version 2.1.0) is a fast, small, and feature-rich JavaScript library. It simplifies HTML document traversal and manipulation, event handling, animation and Ajax, with an easy-to-use API that works across a multitude of browsers.
- jQuery Mobile (version 1.4.2) the jQuery mobile framework for designing a single highly-branded responsive web site or application that works on all popular smart phone, tablet, and desktop platforms.
- jSignature (version 2) - a JavaScript widget for simplifying the creation of a signature capture field in a browser window, allowing a user to draw a signature using mouse, pen or finger.
• knockout.js (version 3.1.0) a JavaScript library for creating rich, responsive display and editor user interfaces with a clean underlying data model.

• Oracle MapViewer HTML5 API - Javascript API for building highly-interactive map views that provide a rich user experience.

## Build Environments vs Runtimes

A build environment is a desktop or laptop on which the pre-requisite software (such as Apache Cordova Command Line Interface) and the mobile integrated development environments (Android SDK, iOS XCode and Weblogic Server) are installed.

Build environments allow for the customizing and packaging of mobile applications (such as .apk and .ipa files).

Examples of build environments are:

• Windows
• Mac OS X
• Linux Environments

Runtimes are mobile devices where the generated artifacts like .apk and .ipa files are deployed. Runtimes have underlying operating systems such as Android and iOS.

Examples of runtimes are:

• iOS devices running iOS 8.1.x
• Android devices running Android 4.4 or greater

<table>
<thead>
<tr>
<th>Build Environment\Runtime</th>
<th>iOS device</th>
<th>Android device</th>
<th>Online Client</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Mac OS X</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Linux</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Chapter 3
Installing Pre-Requisite Software

This chapter describes the steps to install the pre-requisite software for various environments such as:

- Windows Environment
- Mac OS X Environment
- Linux Environment

Windows Environment

The iOS runtime is not supported on Windows. Only Android runtime is supported on Windows.

The following procedure describes the steps to install the pre-requisite software on your Windows environment.

   a. To ensure that npm is working from the command line, execute the below command:
      ```
      npm -version
      ```

2. Download and install Git client (http://git-scm.com)
   a. Set the path variable as below, if Git is not in the path
      ```
      SET PATH=<Git Installed path>/bin;%PATH%
      ```

3. Verify whether the installation system is behind a corporate proxy. If yes, refer to the section Proxy Settings in this guide.

4. Install the Cordova module using the npm utility of Node.js. The Cordova module is automatically downloaded by the npm utility.
   ```
   npm install -g cordova@3.6.3-0.2.13
   ```
   **Note:** Ensure that the Cordova command executes successfully from any directory from command line after installation.

5. Download and install JDK 1.6 (http://www.oracle.com/technetwork/java/javase/downloads/index.html) or above
   a. Configure the variable as JAVA_HOME
   b. Add JAVA_HOME\bin to the path variable according to the environments
   c. Once JDK is installed, Apache Ant runs successfully

6. Download Apache Ant 1.9.x version (http://ant.apache.org/bindownload.cgi)
   a. Unzip the Ant at any preferred location.
b. Set the following environment variables.
   SET ANT_HOME=<unzipped path>

c. Add the bin directory to the path variable
   SET PATH=%ANT_HOME%\bin;%PATH%

d. Ensure that Ant command executes successfully by running:
   ant -version

7. Download ant-contrib.jar version (1.0b2 or later) from http://sourceforge.net/projects/ant-contrib/files/ant-contrib/ and copy the jar under the ANT_HOME\lib directory.

   a. Based on the platform, the appropriate zip file is automatically downloaded from the Content Delivery Network.
   b. Unzip the Android SDK and set the following environment variables in the path.
   c. Set the following environment variables.
      SET PATH= <Unzipped Loc>\tools;%PATH
      SET PATH= <Unzipped Loc>\platform-tools;%PATH%
   d. Once Android is downloaded, ensure that the compatible target of Android for the Cordova version is set, for example Cordova 3.5 supports Android 19.
   e. Refer to the section Installing the Android Targets and install the android targets.

9. Download and install Weblogic11g (http://www.oracle.com/technetwork/middleware/weblogic/downloads/wls-main-097127.html) with basic features in order to support the deployment of the web client application on the server.

Mac OS X Environment

The following procedure describes the steps to install the pre-requisite software on your Mac OS X environment.

1. Download and install the Node.js (http://nodejs.org) and ensure node and npm are working from the command line.

2. Git client (http://git-scm.com) is pre-configured in Mac OS X environment. Ensure that the version is compatible with that of the information provided in the Pre-requisite Software section.

3. Verify whether the installation system is behind a corporate proxy. If yes, refer to the section Proxy Settings in this guide.

4. Install the Cordova module using the npm utility of Node.js. The Cordova module is automatically downloaded by the npm utility.
   a. Ensure the “g” option is provided while installing the Cordova projects from anywhere in the system.
      sudo npm install -g cordova@3.6.3-0.2.13
   b. Provide the password of the user when prompted
   c. Ensure that the Cordova command runs from any directory in the command line

5. Download and install JDK 1.6 or above
   a. Accept the license agreement and download the file.
   b. From either the Downloads window of the browser, or from the file browser, double click the .dmg file to launch it.
   c. A Finder window appears containing an icon of an open box and the name of the .pkg file. Double click the package icon to launch the Install app.
d. The **Install** app displays the Introduction window. Click **Continue**.

e. Note that, in some cases, a **Destination Select** window appears. This is a bug, as there is only one option available. If you see this window, select **Install for all users of this computer** to enable the **Continue** button. Click **Continue**.

f. The **Installation Type** window appears. Click **Install**.

g. A window appears with the message, “Installer is trying to install new software. Type your password to allow this.” Enter the Administrator login and password and click **Install Software**.

h. The software is installed and a confirmation window appears. Click the **ReadMe** for more information about the installation.

i. After the software is installed, delete the dmg file if you wish to save disk space.

6. Download Apache Ant 1.9.x version.
   a. Change directory to Downloads folder
      
      ```bash
      cd ~/Downloads
      ```
   b. Extract the tar.gz file in the downloads folder.
   c. Ensure that the `/usr/local` directory already exists. By default this directory exists out-of-the-box in Mac OS X
      
      ```bash
      sudo mkdir -p /usr/local
      ```
   d. Copy the extracted ant folder into the below location
      
      ```bash
      sudo cp -rf apache-ant-1.9.4-bin /usr/local/apache-ant
      ```
   e. Add the new version of Ant to future terminal sessions
      
      ```bash
      echo 'export PATH=/usr/local/apache-ant/bin:$PATH' >> ~/.profile
      ```

7. Download ant-contrib.jar version (1.0b2 or more) and copy the jar under the `/usr/local/apache-ant/lib` set in step 4.

   a. Download the XCode file. Accept the License agreement.
   b. Open the **Finder** window and go to the **Downloads** folder.
   c. In the downloads window, double click the .dmg folder and the pop up launches.
   d. Drag the XCode icon to the Applications folder.
   e. Once Xcode is installed, click on the icon the dock folder and set it as fix to the Dock for convenience.

---

**Linux Environment**

The following procedure describes the steps to install the pre-requisite software on your Linux environment.

**Note:** The below commands assume a Bash shell. If you use shells other than Bash in your Linux environment, please use appropriate commands.

1. Create a directory where the Linux user has all the access permissions as well as the ownership of the folder created as follows.

   ```bash
   mkdir <MOBILEAPPS_DIR>
   For example mkdir /spl/MobileApps
   echo 'export PROGRAMS=/spl/MobileApps' >> ~/.bashrc
   source ~/.bashrc
   ```
2. Check if the Linux is installed with Python version greater than 2.6, else install by downloading the Python version 2.7 or above and follow these steps:
   a. Extract the python jar to the downloaded folder
      
      tar -xvf Python-2.7.5.tgz
   b. Copy the extracted folder to the $PROGRAMS folder
      
      cp -R Python-2.7.5 $PROGRAMS
   c. Change directory to the $PROGRAMS directory
      
      cd $PROGRAMS/Python-2.7.5
   d. Configure as pre-requisite step for installing
      
      ./configure --prefix=$PROGRAMS/python27
   e. Execute the make command
      
      make
   f. Execute the below command to install.
      
      make install
   g. Execute the below commands to ensure that the path entries are set for the user.
      
      echo 'export PATH=$PROGRAMS/python27/bin:$PATH' >> ~/.bashrc
      source ~/.bashrc

3. Install JDK
   a. Go to installs folder where the jdk installable is downloaded and install JDK.
   b. Copy jdk1.6.0_<xx> to $PROGRAMS
      
      cp -R jdk1.6.0_<xx> $PROGRAMS
   c. Execute the commands to ensure the path is set for the user
      
      echo 'export PATH=$PROGRAMS/jdk1.6.0_<xx>/bin:$PATH' >> ~/.bashrc
      source ~/.bashrc

4. Install Apache Ant 1.9.x or above.
   a. Extract the ant tar folder to the downloaded folder
      
      tar -xvf apache-ant-1.9.4-bin.tar.gz
   b. Copy the folder to $PROGRAMS folder
      
      cp -R apache-ant-1.9.4 $PROGRAMS
   c. Change directory to the $PROGRAMS directory
      
      cd $PROGRAMS
   d. Rename the shortened version for convenience
      
      mv apache-ant-1.9.4 apache-ant
   e. Execute these commands to ensure the path is set for the user.
      
      echo 'export ANT_HOME=$PROGRAMS/apache-ant' >> ~/.bashrc
      echo 'export PATH=$PROGRAMS/apache-ant/bin:$PATH' >> ~/.bashrc
      source ~/.bashrc

5. Download ant-contrib.jar version(1.0b2 or more) and copy the jar file under the $PROGRAMS/apache-ant/lib directory.

6. Install nodejs
   a. Extract the node-v0.10.33.tar.gz file.
      
      tar -xvf node-v0.10.33.tar.gz
   b. Change directory to the extracted folder.
      
      cd node-v0.10.33/
c. Configure with the prefix value in order to ensure the write permission as well as ownership of the folder you target to.

   
   ./configure --prefix=$PROGRAMS/nodejs

d. Execute the make command

   
   make

e. Execute the make install command

   
   make install

f. Ensure that the path is set for the user by executing the below commands

   
   echo 'export PATH=$PROGRAMS/nodejs/bin:"$PATH"' >> ~/.bashrc
   source ~/.bashrc

7. Install Cordova

   Before proceeding, verify if the installation system is behind a corporate proxy. If yes, refer to the section Proxy Settings in this guide.

   a. Run the command with the version of Cordova you wish to install

      
      npm install -g cordova@3.6.3-0.2.13


   a. The appropriate zip file for the Linux platform will be downloaded for e.g. adt-bundle-linux-x86_64-20131030.zip

   b. Execute the following commands:

      
      unzip adt-bundle-linux-x86_64-20131030.zip -d .
      mv adt-bundle-linux-x86_64-20131030 Android
      cp -R Android $PROGRAMS

   c. Execute the below commands to set the Android SDK path

      
      echo 'export PATH=$PROGRAMS/Android/sdk/tools:"$PATH"' >> ~/.bashrc
      echo 'export PATH=$PROGRAMS/Android/sdk/platform-tools:"$PATH"' >> ~/.bashrc
      source ~/.bashrc

   d. Please refer to the section Installing the Android Targets and install the android targets.

9. Download and install Weblogic 11g (http://www.oracle.com/technetwork/middleware/weblogic/downloads/wls-main-097127.html) with basic features in order to support the deployment of the web client application on the server.
Installing the Oracle Real-Time Scheduler Mobile Application

This chapter explains the steps to install the Oracle Real-Time Scheduler Mobile Application using the Oracle Universal Installer.

This chapter includes:

- Installing the Mobile Application

## Installing the Mobile Application

To install the mobile application, perform these steps:

1. Download the Oracle Real-Time Scheduler v2.2.0.3 Mobile Application Multiplatform (HTML5-based).zip from the Oracle Software Delivery Cloud.
2. Unzip the file in a temporary directory `<TEMPDIR>`.
3. Upon extracting the zip file, a jar file called Mobile_Application_Installer.jar is created in the `<TEMPDIR>` directory.
4. Set the JAVA_HOME and PATH variables as shown in the examples below:
   
   **Note:** The minimum recommended version of JDK is 1.7.0_65 in order to launch the Oracle Universal Installer.

   **Windows:**
   ```
   set JAVA_HOME=<JAVA_INSTALL_LOCATION>\jdk1.7.0_65
   set PATH=%JAVA_HOME%\bin;%PATH%;.
   ```

   **Linux:**
   ```
   export JAVA_HOME=<JAVA_INSTALL_LOCATION>/jdk1.7.0_65
   export PATH=${JAVA_HOME}/bin:$PATH
   ```

   **Mac OS X:**
   ```
   export JAVA_HOME=<JAVA_INSTALL_LOCATION>/jdk1.7.0_65
   export PATH=${JAVA_HOME}/bin:$PATH
   ```

5. Execute the following command:
   ```
   java -jar <TEMPDIR>/Mobile_Application_Installer.jar -logLevel finest
   ```

   The **Next Generation Oracle Universal Installer** is launched.
6. Click **Next** on the **Welcome** page of the Next Generation Oracle Universal Installer.

7. On the **Installation Inventory Setup** page, specify a directory where the installation metadata files are to be stored in the **Inventory Directory** field and click **Next**.

   **Note:** The **Installation Inventory Setup** page only appears for a first-time installation on your machine through **Oracle Universal Installer**.

8. On the **Installation Location** page, browse and point to a local folder where you wish to install the mobile application (this is known as the `<ORACLE_HOME>`) and click **Next**.

   ![Installation Location Page]

   The **Installation Progress page** launches and displays the progress of the installation.

9. Click **Next** on the Installation Progress page, once the installation is shown as 100%.

   ![Installation Progress Page]
The **Installation Complete** page launches.

10. Click **Finish** on the **Installation Complete** page.

    This completes the installation of the mobile application.

**Post-installation Steps**

For Max OS X and Linux environments, an additional post-installation step is required to set permissions to build the mobile application.

**Note:** The `<PRODUCT_HOME>` is defined as:

```
<PRODUCT_HOME> = <ORACLE_HOME>/MobileLibrary
```

**Mac OS X Environment**

Run the following commands to set the permissions:

```
cd <PRODUCT_HOME>
sudo chmod -R 755 *
```

**Linux Environment**

Run the following command to set the permissions.

```
cd <PRODUCT_HOME>
chmod -R 755 *
```
Chapter 5
Building the Mobile Application

This chapter describes the following steps for building the mobile application:

- Downloading Javascript and CSS Libraries to the Installed Location
- Pre-build Configuration Updates
- Creating Apache Cordova Project
- Preparing SDKs for Building the Mobile Application (for iOS)
- Building the Mobile Application on Apache Cordova Project
Downloading Javascript and CSS Libraries to the Installed Location

The following section describes the steps to be followed to download various Javascript and CSS libraries used for building the application, to the installed location.

Under the `<PRODUCT_HOME>/source/www` location, create the ‘libs’ folder:

1. jQuery Libraries
   a. Create folder `<PRODUCT_HOME>/source/www/libs/jquery`
   b. Download jQuery js file version 2.1.0 (https://code.jquery.com/jquery/) 
   c. Copy the `jquery-2.1.0.min.js` file into `<PRODUCT_HOME>/source/www/libs/jquery`

2. jQuery Mobile Libraries
   a. Create folder `<PRODUCT_HOME>/source/www/libs/jquery/mobile`
   b. Download jQuery mobile library version 1.4.2 (http://jquerymobile.com/download/all/)
   c. Copy the js and css files and the images folder from the jQuery mobile downloaded location into `<PRODUCT_HOME>/source/www/libs/jquery/mobile`

3. jSignature Libraries
   a. Create folder `<PRODUCT_HOME>/source/www/libs/jSignature`
   b. Download and copy jSignature (http://willowsystems.github.io/jSignature)
   c. Copy the `jSignature.min.js` file from the downloaded location to `PRODUCT_HOME>/source/www/libs/jSignature`

4. Knockout libraries
   a. Create folder `<PRODUCT_HOME>/source/www/libs/knockout`
   b. Download knockout version 3.1.0 (http://knockoutjs.com/downloads/) and rename the js file to “knockout.js”
   c. Download the knockout js mapping plugin (http://knockoutjs.com/documentation/plugins-mapping.html) and rename the js file to “knockout.mapping.js”.
   d. Copy the knockout and knockout mapping js files from the downloaded location to `<PRODUCT_HOME>/source/www/libs/knockout`

5. Oracle Maps
   a. Create folder `<PRODUCT_HOME>/source/www/libs/oraclemaps/v2`
   b. Download the Standalone Javascript V2 (HTML5) API library for Oracle Fusion Middleware MapViewer version 11g ps6 (11.1.1.7.3) (http://www.oracle.com/technetwork/middleware/mapviewer/downloads/index.html)
   c. Copy the contents of the downloaded zip files v2 folder to `<PRODUCT_HOME>/source/www/libs/oraclemaps/v2`
   d. You can delete the apidoc and tutorials folders to reduce the size of the generated native application
Pre-build Configuration Updates

Theme/styles extension placeholder files

Placeholder files need to be created for theme/styles extension. These files can be used to add custom content for customizing themes/styles for the application.

a. Create `<PRODUCT_HOME>/source/www/cm/themes`

b. Create the following files without any content and add them to the above mentioned directory:
   - `cm.jquery.theme.min.css`
   - `jquery.mobile.icons.min.css`
   - `cm.styles.css`
Creating Apache Cordova Project

The Apache Cordova project is used to create native applications for different mobile operating systems. The same Apache Cordova project can be used to create native applications for different mobile operating systems.

1. Configure the properties mentioned in the Appendix Configurations.

   There are two types of installation
   - Silent Installation - If all the properties in Appendix Configurations are provided, then the installation proceeds in silent mode.
   - Interactive Installation - If any of the properties are not provided in the properties file, then you are allowed to enter values in interactive mode.

2. Create the Cordova project at the location provided as input parameter or as input properties.

   Go to the location where the product has been extracted (<PRODUCT_HOME>) and run the following command:
   ```
   cd <PRODUCT_HOME>/bin
   ant -f InstallCreateProject.xml
   ```
   This command by default takes all the values from the product_config.properties file.

   If you wish to use a different properties file, run the following command:
   ```
   ant -f InstallCreateProject.xml -DproductProperties=<Different Location>/product_config_<xyz>.properties
   ```

   The creation of Apache Cordova Project does the following:
   - Executes the Cordova commands through ant utility to create the project at desired location as CORDOVA_HOME.
   - Creates the Apache Cordova project with the project name provided in the configuration file.
   - Adds the Android and iOS platforms for the Cordova project, based on the flag set in the configuration file.
   - Gets the plugins from Git repository and installs under the Cordova plugins directory.
   - Creates the custom plug-ins, if any, onto the Cordova plugins directory.
Preparing SDKs for Building the Mobile Application (for iOS)

This section describes the steps required to prepare SDKs for building the mobile application for iOS.

Refer to the iOS Developer Library -> App Distribution Guide for more information about application distribution.

Registration of Certificate

1. From the Finder screen select Go -> Applications -> Utilities -> Keychain Access.
2. Ensure that the login key chain is the default.
   The Keychain Access page opens.
4. Select the login keychain and from the Main Menu, select File -> Import Items.
   5. In the Import dialog, select the <Name of Certificate>.p12 file from the unpacked zip and click Open.
   6. When prompted, enter the password that you received from the certifying authority.
      The certificate is now registered on the machine.
7. Right click the newly-added certificate iPhone Distribution: <Your Company> (Ent1) and click on “Evaluate iPhone Distribution: <your Company> (Ent1)”

8. In the Certificate Assistant page that opens, select Code Signing and click Continue.

9. On the Certificate Assistant page, under Specify Certificates to be Viewed and Evaluated, check the option Include certificates from my keychains and click Done.

This profile can now be used for building the Oracle Real-Time Scheduler mobile application project.

**Code Signing of Project**

1. Launch the XCode application.
2. Click Open another project.
3. Browse to the directory where the XCode project (henceforth in this guide, the project name will be referred as ORS_Application) is created and select the XCode project name and click Open.
4. In the Project Navigator, click on the XCode project name.
5. Under Project, select XCode project name and click Build Settings.
6. Under Code Signing section in the Build Settings, ensure that all the entries under the “Code Signing Identity” are pointing to “iPhone Distribution:<Your Company>” entry.

This completes the code signing of the project.

**Adding an Account for the Project**

1. Select the ORS Application, select the General tab.
2. In the Identity section, select the Team option as Add an account.
3. In the dialog that pops up, enter the Apple ID and Password for the account and click Add.

**Adding the Device Orientation**

1. Select the ORS Application, select the General tab.
2. In the Deployment Info section, select the Device Orientation as:
   • Portrait
• Upside Down
• Landscape Left
• Landscape Right
Building the Mobile Application on Apache Cordova Project

This section describes how to build the mobile application using Apache Cordova.

Go to the location where the product has been extracted and run the following command:

```
cd <PRODUCT_HOME>/bin
ant -f InstallBuildProject.xml
```

OR

If you wish to build the project using a different properties file, run the following command:

```
ant -f InstallBuildProject.xml -DproductProperties=<DifferentLocation>/product_config_<xyz>.properties
```

This script performs the following:

- Copies code artifacts that are developed are copied from PRODUCT_HOME to CORDOVA_HOME.
- Compiles the internally developed and the Cordova provided plugins, with the Cordova build utility invoked by our ant wrapper scripts.
- Generates the xcodeproject and apk files through the Cordova build command, based on the flag set in the properties file.
- Creates the WAR File which is deployable on Weblogic server.
- If “Deploy on Weblogic Server” is opted, then the application is deployed on the Weblogic server mentioned.
Chapter 6

Deploying the Mobile Application

This chapter describes the steps for deploying the mobile application and includes:

- Android Deployment
- iOS Deployment
- Web Application

Android Deployment

Android application (APK) file is generated after you follow the steps in Building the Mobile Application on Apache Cordova Project. The application (APK) file is created in the Cordova project under the folder <PRODUCT_HOME>/output or any other output folder as specified in the properties file. The file created is ORS_Application-release-signed_<timestamp>.apk and can be installed on Android devices.

1. To install the APK file on Android device, make sure the option to install apps from unknown sources is checked in your device. This option is under Settings -> Security under Android ICS.
2. Copy the APK file to your device by connecting it to the PC using a USB cable.
3. Browse to the APK location using File Manager and open and install the APK.
   
   If File browser is not available on the device, you can install it from Google Play e.g. ES File Explorer File Manager.

iOS Deployment

The iOS application has to be built from the Xcode IDE. The section Building the Mobile Application on Apache Cordova Project creates the Xcode project which in turn can be used to create the iOS application (IPA) file.

1. Through Finder, navigate to the <Cordova project>/platforms/ios folder and double click the ORS_Application.xcodeproj. This opens the project in Xcode.
2. Xcode can be used to test the application in iOS Simulator or to build the archive (IPA) file which can be installed on iOS device using iTunes.
3. To test the application on Simulator in Xcode, select the iPad/iPhone simulator and click Run.
   
   This opens the application in the simulator. You may test the application.
4. Change the active scheme to iOS Device or to the iPhone name which is connected to Mac.
5. Select the ORS Application -> Build Settings -> Code Signing section, ensure you select iPhone Distribution for Enterprise option for both Debug and Release options.


7. Enter the password for the pop-ups like keychain login.

   This launches the Organizer which displays the current and past builds.

8. Select the latest build and click Export.


10. Select an appropriate development team to use for provisioning and click Choose.


   You are prompted to enter a name for the ipa file.

12. Enter the name as ORS_Application and browse to the directory where you wish to save this ipa file.

13. Click Export.

   This generates the ipa file.

14. Connect your iOS device to the machine and launch iTunes.

15. Under your device -> Apps tab, the ORS_Application is displayed and can be installed by clicking Install.

   This completes the installation of the application on the device.

---

**Web Application**

The web application should be installed into the same domain/host where the Oracle Real-Time Scheduler server application (REST services) is installed.

When the InstallBuildProject command is executed:

- The war file is created.
- The script reads the values for the user name and password (MOB_CLIENT_WL_USER and MOB_CLIENT_WL_PWD) from the product_config.properties file.
- For security purposes, the script creates the Weblogic User configuration files and removes these entries for user name and password from the product_config.properties file.
- If the “install.web.application” option is set as ‘y’, the bundled war file is deployed on the server with the properties provided.

The build script ensures that the web application is deployed on the Weblogic instance on the configured target server.

In order to verify whether the application is installed on the server, follow these steps:

1. Launch the Oracle Weblogic Server Administration Console.

2. Click the Deployments link.
3. Click the selected mobile client to view the details of the deployed application.

![Image of Oracle Real-Time Scheduler]

4. In the **Overview** tab, take a note of the context root.

5. Open a browser and enter the URL as “http://<hostname>:<portnumber>/<contextroot>” to the context root and enter it in your browser as your URL.

   This launches the Oracle Real-Time Scheduler web client application.
Appendix A

Configurations

This appendix serves as a reference for the various configurations for building the Oracle Real-Time Scheduler mobile application, including:

- Project Configurations
- Plugin Configurations
- Installing the Android Targets
- Proxy Settings

Project Configurations

The extracted folder or etc/config consists of the following configurable items:

- **internal_product_config.properties** - These properties are internal properties which are not intended to be changed. Oracle does not recommend you to change any of these entries.

- **product_config.properties** - The tables below describe the configurable properties in this file.

Common Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>cordova.project.directory</td>
<td>Creates the Apache Cordova Project</td>
<td>C::/ORS_Application/CordovaProjects/HD22100_Release</td>
</tr>
<tr>
<td>install.android.application</td>
<td>Option to create the android platform for the Apache Cordova Project</td>
<td>y for Yes n for No</td>
</tr>
<tr>
<td>install.ios.application</td>
<td>Option to create the iOS platform for the Apache Cordova Project. However, the iOS platform is supported only on Mac OS.</td>
<td>y for Yes n for No</td>
</tr>
<tr>
<td>install.web.application</td>
<td>Supports installation of the Online Client web war</td>
<td>y for Yes n for No</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------------</td>
</tr>
<tr>
<td>projectId</td>
<td>Used to create the projects on devices. It is the package name of the application as well.</td>
<td>com.oraclecorp.internal.mwm.orsapplication</td>
</tr>
<tr>
<td>projectName</td>
<td>Name of the project to be generated.</td>
<td>ORS_Application</td>
</tr>
<tr>
<td>REST_SERVICES_WEB_WLHOST</td>
<td>Host where the REST services reside. Highly recommended to set this to be the same server where client war is installed.</td>
<td>xxxx.oracle.com</td>
</tr>
<tr>
<td>REST_SERVICES_WEB_WLPORT</td>
<td>Port number where the REST services deployed listens.</td>
<td>9999</td>
</tr>
<tr>
<td>MAP_FEATURE_CONFIG_NAME</td>
<td>Map server configuration code as configured in Admin Menu -&gt; Feature Configuration. There should only be 1 Map server configuration defined in the system.</td>
<td>CM-MapConfig</td>
</tr>
<tr>
<td>MW_HOME</td>
<td>Middleware home where you wish to deploy the application. This property is valid when install. web.application is set as y</td>
<td>C:/Oracle/Middleware</td>
</tr>
<tr>
<td>WL_HOME</td>
<td>Weblogic Home</td>
<td>wlservice_10.3 for 11g</td>
</tr>
<tr>
<td>MOB_CLIENT_WAR_FILENAME</td>
<td>File name of the Mobile Client war file</td>
<td>${PRODUCT_HOME}/output/HDmobileClient.war</td>
</tr>
<tr>
<td>MOB_CLIENT_WEB_CONTEXT_ROOT</td>
<td>Context Root of the Mobile Application</td>
<td>/mobility</td>
</tr>
<tr>
<td>MOB_CLIENT_WAR_APP_NAME</td>
<td>Application Name</td>
<td>HDmobileClient</td>
</tr>
<tr>
<td>MOB_CLIENT_WEB_WLHOST</td>
<td>Host where Mobile Client is to be installed. <strong>Note:</strong> This must be installed into the same domain/host where the Oracle Real-Time Scheduler server application (REST services) is installed.</td>
<td>xxxx.oracle.com</td>
</tr>
</tbody>
</table>
Plugin Configurations

Plugins are used by the mobile applications in order to use native features e.g. camera, video recording etc. Apache Cordova comes with a default set of plugins available which are called core plugins. The list of core plugins is available in the Apache Cordova documentation.

Besides the core plugins, implementers can also develop their own plugins or use other available plugins.

The Apache Cordova plugins that are needed to be added for the mobile application can be configured using the `<Installed Location>/etc/config/plugins.xml` file. While creating the Cordova project, if the install attribute of the respective plugin element is set to “yes”, then the plugin is installed.

**Note:** For creating the Cordova project for iOS platforms, set the install=”no” for "${PRODUCT_HOME}/plugins/com.oraclecorp.internal.mwm.security.crypto" plugin

The following is a sample of the plugins.xml file.

```xml
<plugins xmlns="http://ouml.cordova.plugins.oracle.com">
    <plugin name="https://git-wip-us.apache.org/repos/asf/cordova-plugin-device.git#r0.2.12" install="yes"/>
    <plugin name="https://git-wip-us.apache.org/repos/asf/cordova-plugin-camera.git#r0.3.2" install="yes"/>
    <plugin name="https://git-wip-us.apache.org/repos/asf/cordova-plugin-file.git#r1.3.1" install="yes"/>
</plugins>
```
<plugin name="https://git-wip-us.apache.org/repos/asf/cordova-plugin-geolocation.git#r0.3.10" install="yes"/>
<plugin name="https://git-wip-us.apache.org/repos/asf/cordova-plugin-inappbrowser.git#r0.5.2" install="yes"/>
<plugin name="https://git-wip-us.apache.org/repos/asf/cordova-plugin-network-information.git#r0.2.11" install="yes"/>
<plugin name="https://github.com/wildabeast/BarcodeScanner" install="yes"/>
<plugin name="https://github.com/brodysoft/Cordova-SQLitePlugin" install="yes"/>
<plugin name="https://github.com/katzer/cordova-plugin-background-mode" install="yes"/>
<plugin name="${PRODUCT_HOME}/plugins/com.oraclecorp.internal.mwm.security.crypto" install="yes"/>

The following table describes these plugins.

<table>
<thead>
<tr>
<th>Plugin</th>
<th>Plugin name in XML</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>org.apache.cordova.device</td>
<td><a href="https://git-wip-us.apache.org/repos/asf/cordova-plugin-device.git#r0.2.12">https://git-wip-us.apache.org/repos/asf/cordova-plugin-device.git#r0.2.12</a></td>
<td>Defines a global device object, which describes the device's hardware and software.</td>
</tr>
<tr>
<td>org.apache.cordova.camera</td>
<td><a href="https://git-wip-us.apache.org/repos/asf/cordova-plugin-camera.git#r0.3.2">https://git-wip-us.apache.org/repos/asf/cordova-plugin-camera.git#r0.3.2</a></td>
<td>Provides an API for taking pictures and for choosing images from the system's image library. Takes a photo using the camera, or retrieves a photo from the device's image gallery. The image is passed to the success callback as a base64-encoded String, or as the URI for the image file.</td>
</tr>
<tr>
<td>org.apache.cordova.plugin.file</td>
<td><a href="https://git-wip-us.apache.org/repos/asf/cordova-plugin-file.git#r1.3.1">https://git-wip-us.apache.org/repos/asf/cordova-plugin-file.git#r1.3.1</a></td>
<td>Implements a File API allowing read/write access to files residing on the device.</td>
</tr>
<tr>
<td>org.apache.cordova.geolocation</td>
<td><a href="https://git-wip-us.apache.org/repos/asf/cordova-plugin-geolocation.git#r0.3.10">https://git-wip-us.apache.org/repos/asf/cordova-plugin-geolocation.git#r0.3.10</a></td>
<td>Provides information about the device's location, such as latitude and longitude. Common sources of location information include Global Positioning System (GPS) and location inferred from network signals such as IP address, RFID, WiFi and Bluetooth MAC addresses, and GSM/CDMA cell IDs.</td>
</tr>
<tr>
<td>org.apache.cordova.inappbrowser</td>
<td><a href="https://git-wip-us.apache.org/repos/asf/cordova-plugin-inappbrowser.git#r0.5.2">https://git-wip-us.apache.org/repos/asf/cordova-plugin-inappbrowser.git#r0.5.2</a></td>
<td>Provides a web browser view that displays when calling window.open.</td>
</tr>
<tr>
<td>Cordova-Plugin-network-information</td>
<td><a href="https://git-wip-us.apache.org/repos/asf/cordova-plugin-network-information.git#r0.2.11">https://git-wip-us.apache.org/repos/asf/cordova-plugin-network-information.git#r0.2.11</a></td>
<td>Provides information about the cellular and Wi-Fi connectivity devices, and whether the device must connect to the Internet.</td>
</tr>
</tbody>
</table>
1. Launch the Android SDK Manager using the following command:

   android

   The Android SDK Manager launches.

2. Ensure that the SDK Platform and ARM EABI v7a System Images are installed with the revision of 19 as depicted above.

3. If the system is behind the corporate proxy, please refer to section Proxy Settings.

4. Ensure that the compatible Android API is selected for that of the compatible version of the Cordova version that is installed. Here in this scenario for Cordova 3.6.x, the targeted version is Android 4.4 API 19.

5. Create the Android Virtual Device by navigating through the Tools -> Manage AVDs

<table>
<thead>
<tr>
<th>Plugin</th>
<th>Plugin name in XML</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BarcodeScanner</td>
<td><a href="https://github.com/wildabeast/BarcodeScanner">https://github.com/wildabeast/BarcodeScanner</a>&quot;</td>
<td>Cross-platform BarcodeScanner for Cordova</td>
</tr>
<tr>
<td>Cordova-SQLitePlugin</td>
<td><a href="https://github.com/brodysoft/Cordova-SQLitePlugin">https://github.com/brodysoft/Cordova-SQLitePlugin</a></td>
<td>A Cordova/PhoneGap plugin to open and use SQLite databases on Android/iOS/WP(8) with HTML5 Web SQL API</td>
</tr>
<tr>
<td>cordova-plugin-background-mode</td>
<td><a href="https://github.com/katzer/cordova-plugin-background-mode">https://github.com/katzer/cordova-plugin-background-mode</a></td>
<td>A bunch of background mode plugins for Cordova 3.x.x</td>
</tr>
</tbody>
</table>

Table 1: Plugin Descriptions
6. Configure the values as below.

7. Click **OK**.

The Android environment is ready for building the Cordova applications for Android platform.

**Proxy Settings**

This section describes the configurations that are needed in order to bypass the corporate proxy for the following applications:

**Node js**

Run the below commands from Command line interface to bypass the npm entries:

```
npm config set proxy http://<<proxy-host>>:<<proxy-port>>
npm config set https-proxy http://<<proxy-host>>:<<proxy-port>>
```

While running the commands, if you face any issues please create the directory

```
<Users-directory>\AppData\Roaming\npm
```

**Git**

For by passing the git entries, run the below commands from Command Line interface

```
git config --global http.proxy http://<<proxy-host>>:<<proxy-port>>
```

```
git config --global https.proxy http://<<proxy-host>>:<<proxy-port>>

In order to ensure that the below commands are successfully executed, go to the users folder.

Windows: - C:/Users/<user-name>
Linpus: - /usr/<user-name>

Ensure that the .npmrc and .gitconfig files are present. Ensure that the proxy host and proxy port details are defined for http and https protocols.

Android SDK

1. Launch the SDK Manager.
2. Select Tools → Options.
   The Settings page launches.
3. Enter values for the Http Proxy Server and Http Proxy Port fields.

Mac OS X

1. In order to set the proxy settings for the Mac OS X, go to the Apple Icon → System Preferences.
2. Click the Network Icon
3. Select the type of connection you have chosen and click Advanced.
4. Select the Proxies tab.
5. Enter the values for Web Proxy and Secure Web Proxy.