



**Hardware and Software**  
Engineered to Work Together

# Siebel Open UI - Advanced JavaScript API

Student Guide

D84417GC20

Edition 2.0 | January 2016 | D94558

Learn more from Oracle University at [oracle.com/education/](http://oracle.com/education/)

## **Authors**

Customer Experience (CX)  
Curriculum Development

## **Technical Contributors and Reviewers**

Customer Experience (CX)  
Product Development Team

## **Publishers**

Sumesh Koshy  
Sujatha Nagendra

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

### **Disclaimer**

This document contains proprietary information and is protected by copyright and other intellectual property laws. You may copy and print this document solely for your own use in an Oracle training course. The document may not be modified or altered in any way. Except where your use constitutes "fair use" under copyright law, you may not use, share, download, upload, copy, print, display, perform, reproduce, publish, license, post, transmit, or distribute this document in whole or in part without the express authorization of Oracle.

The information contained in this document is subject to change without notice. If you find any problems in the document, please report them in writing to: Oracle University, 500 Oracle Parkway, Redwood Shores, California 94065 USA. This document is not warranted to be error-free.

### **Restricted Rights Notice**

If this documentation is delivered to the United States Government or anyone using the documentation on behalf of the United States Government, the following notice is applicable:

#### **U.S. GOVERNMENT RIGHTS**

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose these training materials are restricted by the terms of the applicable Oracle license agreement and/or the applicable U.S. Government contract.

### **Trademark Notice**

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

# Contents

## **1 Course Introduction**

- Lesson Agenda 1-2
- Instructor and Class Participants 1-3
- Training Site Information 1-4
- Course Audience 1-5
- Course Prerequisites 1-6
- Course Goal 1-7
- Course Objectives 1-8
- Course Methodology 1-9
- Course Materials 1-10
- Information Sources 1-11
- Course Agenda 1-12

## **2 Overview of Open UI Customization**

- Objectives 2-2
- Siebel Open UI 2-3
- The Open UI Framework 2-4
- Siebel Open UI Client 2-5
- Parts of the Siebel Open UI Framework 2-6
  - 1. Client Proxy 2-7
  - 2. Presentation Models (PMs) 2-8
  - 3. Physical Renderers (PRs) 2-9
  - 3. Plug-in Wrappers 2-10
  - 4. Cascading Style Sheets (CSS Files) 2-11
- Customizing the Open UI Framework 2-12
- Development Roles 2-13
- The JavaScript Application Programming Interface (JSAPI) for Open UI 2-14
- Lesson Highlights 2-15
- Practice 2-16

## **3 Administering the Manifest**

- Objectives 3-2
- Review: The Siebel Open UI Manifest 3-3
- Review: Manifest Files 3-4
- Review: Manifest Expressions 3-5

- Review: Manifest Administration 3-6
- Default Behavior 3-7
- Extending Default Behavior 3-8
- Extending Existing Objects 3-9
- Expressions 3-10
- Create a New Conditional Expression 3-11
- Multiple Conditional Expressions 3-12
- Expression Groups 3-13
- Expressions within an Expression Group 3-14
- Review: Administer the Manifest 3-15
- 1. Identify the Name of the Object to Customize 3-16
- 2. Ensure the File to Apply is Registered 3-17
- 3. Select the Object to Customize 3-18
- 4. Associate the Object with a Condition 3-20
- 5. Associate the Object/Condition with a File 3-21
- 6. Apply the Modified Manifest 3-22
- 7. Test the Customization 3-23
- Lesson Highlights 3-24
- Practice 3-25

#### **4 Using jQuery**

- Objectives 4-2
- jQuery 4-3
- jQuery in JavaScript 4-4
- Selecting HTML Elements in jQuery 4-5
- Selecting Child HTML Elements in jQuery 4-7
- Performing Actions on the Selected Set 4-8
- Function Actions 4-9
- CSS Function Example 4-10
- Setting a Style Defined in a CSS Style Sheet 4-11
- html Function Example 4-12
- after/before Function Example 4-14
- Attaching an Event Handler Function to the Selected Set 4-15
- Example of a Click Event Handler 4-16
- Storing Selected Elements in a Variable 4-17
- Chaining Actions 4-18
- Choosing a jQuery Selector 4-19
- Disabling the Cache 4-22
- Caching versus Re-Reading the Manifest 4-23
- Use jQuery 4-24
- 1. Identify the Control or Field to be Manipulated 4-25

- 2. Add one or more jQuery selector/action statements 4-26
- 3. Test Your New Customization 4-27
- Lesson Highlights 4-28
- Practice 4-29

## **5 Customizing a Presentation Model for a Form Applet**

- Objectives 5-2
- Review: The Presentation Model 5-3
- Review: File Format for a Presentation Model 5-4
- Presentation Model Templates 5-5
- Example Presentation Model Template File 5-6
- Customizing a PM with a New Method 5-10
- Using AddMethod() to Hook into a Framework Method 5-11
- Settings Arguments to AddMethod() 5-12
- Example: Define your Custom Method 5-13
- The Collection of Controls 5-14
- Finding Control Key and FieldName Values 5-15
- Methods Used in a PM to Work with Controls in an Applet 5-16
- Alternate Ways to Access a Single Control 5-17
- Example of Getting Controls: The ShowSelection() Method 5-19
- Accessing Controls in a Method Bound to ShowSelection() 5-20
- Accessing an Individual Field Value for ShowSelection() 5-21
- Customize a PM for a Form Applet 5-22
- Caveat: Client-Side Data Validation 5-23
- 1. Create the Basic PM file 5-24
- 2. Use AddMethod() to bind a custom method to the appropriate internal framework method 5-25
- 3. Write a new method to implement the business requirement 5-26
- 4. Save the file with its new file name 5-27
- 5. Apply the New PM to the Applet 5-28
- 6. Test the Solution 5-29
- Lesson Highlights 5-30
- Practice 5-31

## **6 Customizing a Physical Renderer for a Form Applet**

- Objectives 6-2
- Review: The Physical Renderer (PR) 6-3
- Review: File Format for a Physical Renderer 6-4
- Physical Renderer Templates 6-5
- Example Physical Renderer Template File 6-6
- Additional Physical Renderer Methods 6-8

Example Physical Renderer Template File 6-9  
The ShowUI() Method 6-12  
The BindData() Method 6-13  
The BindEvents() Method 6-14  
The EndLife() Method 6-15  
Custom Physical Renderers 6-16  
Some of the JavaScript API Methods Used in a PR 6-17  
Accessing a Specific Control/Field in the PR 6-18  
Customize a PR for a Form Applet 6-19  
1. Create a Template PR File for a Form Applet 6-20  
2. Modify the File to Achieve the Business Goal 6-21  
3. Apply the New PR to the Applet 6-23  
4. Test the Solution 6-24  
Lesson Highlights 6-25  
Practice 6-26

## **7 Presentation Model / Physical Renderer Interaction**

Objectives 7-2  
Separation of Concerns in the Open UI 7-3  
Advantages of SoC in Siebel Open UI 7-4  
PM / PR Interoperation 7-5  
Example of MVC Interoperation Using a Property in the PM Bound in the PR 7-6  
Methods Used to Create Interactions Between PR and PM 7-7  
Creating a Custom Property in the PM 7-8  
Changing the Custom Property in the PM 7-9  
Binding a Callback Method in the PR to a Property Change in the PM 7-10  
Accessing the Custom PM Property Value in the Callback Method in the PR 7-11  
Tips on Locating Code 7-12  
Tips on Locating Code: Details 7-13  
Create PM/PR Interaction 7-14  
1. Add a Property to the PM 7-15  
2. In the PM, Add Code to Set the Property As Needed 7-16  
3. In the PR, Bind a Callback to the Property 7-17  
4. In the PR, Code the Callback Method 7-18  
5. Test the Results 7-19  
Lesson Highlights 7-20  
Practice 7-21

## **8 Debugging**

Objectives 8-2  
Sources of Problems in Scripts 8-3

Verify that the Script was Loaded at Runtime 8-4  
Errors that Can Prevent Script Loading 8-5  
Check for Syntax Errors in the JavaScript that Prevent Execution from Starting 8-6  
Check for Runtime Exceptions that Halt Execution 8-7  
Search the Code for Logical Errors 8-8  
Debugging Tools 8-9  
Viewing Variable Values 8-11  
Inspecting the Call Stack 8-13  
Stepping Through Code 8-14  
Ensure that the Cache Is Disabled or Cleared 8-15  
Text Logging and Interactive Alerts 8-16  
Debug a Script 8-17  
1. Verify that the Cache is Not Active 8-18  
2. Verify that Script Has Been Loaded 8-19  
3. Verify that There are No Syntax Errors in the JavaScript 8-20  
4. Verify that There are No Runtime Errors in the JavaScript 8-21  
5. Use Debugging Tools to Search for Logical Errors 8-22  
Lesson Highlights 8-24  
Practice 8-25

## **9 Customizing a List Applet**

Objectives 9-2  
Structure of a List Applet 9-3  
Placeholder 9-4  
Record Set 9-5  
An Individual Record 9-6  
Accessing an Individual Row by HTML ID 9-7  
Accessing a Field within a Row 9-8  
A Basic PR for a List Applet 9-9  
Search for a Row in a List Applet 9-10  
Search for a Field Value in a List Applet 9-11  
Alternate Looping Techniques 9-12  
BindData() Method in a List Applet 9-13  
ShowUI() Method 9-14  
A PM for List Applets 9-15  
Customize a List Applet 9-16  
1: Create the Basic PR File 9-17  
2: Add Code to the Appropriate Method 9-18  
3: Administer the New PR 9-20  
4: Test Your Changes 9-21

Lesson Highlights 9-22

Practice 9-23

## **10 Helper Techniques**

Objectives 10-2

Helpers 10-3

Additional Features 10-4

Plug-in Wrappers 10-5

Additional Methods of Plug-in Wrappers 10-7

Example AttachPW 10-9

Use a Plug-in Wrapper to Extend a Control on an Applet 10-10

1. Write the Plug-in Wrapper Code 10-11

2. Attach the Plug-in Wrapper 10-15

3. Administer the Plug-in Wrapper 10-16

4. Test the Results 10-17

Additional Features 10-18

The EventHelper() Object 10-19

Implement an EventHelper() Object 10-20

Example: Highlight a Field when the Mouse Enters it 10-21

Working Example of Event Manager 10-22

Additional Features 10-23

HTML Template Manager 10-24

Example: HTML Template Manager 10-25

Working Example of HTML Template Manager 10-26

Additional Features 10-27

Responsive Web Pages 10-28

Dynamic Layouts 10-29

Default Widths 10-30

Widths in Web Template Files 10-31

Resulting Behavior 10-32

Lesson Highlights 10-33

Practice 10-34

## **11 Customization of Non-Applet Objects**

Objectives 11-2

Applet and Non-Applet UI Elements 11-3

postload 11-4

Adding a Listener for postload 11-5

Application-Level Customizations 11-6

Example Global Customizations Appropriate for Use in Postload 11-8

Making Applets Collapsible 11-9

Sorting in jQuery 11-10  
A Custom Sorting Function 11-11  
Customize a Non-Applet Element using postload 11-12  
1. Create a Custom File with a Listener for the postload Event 11-13  
2. Write Custom Code in the Event Handler 11-14  
3. Administer the File 11-16  
4. Test your Changes 11-17  
Lesson Highlights 11-18  
Practice 11-19

## **12 Using External Libraries and Web Sites**

Objectives 12-2  
Using External Code 12-3  
External jQuery Libraries 12-4  
Using External jQuery Libraries 12-5  
Copy the External Package to Your Site 12-6  
Include the External jQuery Package as a Dependency in Your Code 12-7  
Follow the Documentation for the External Package 12-8  
JavaScript Code Accessed via the Web 12-9  
Accessing Code from a URL Asynchronously 12-10  
Steps to Access Code Online via a URL 12-11  
Asynchronously Load the API 12-12  
Load Module/Packages with a Callback 12-15  
Check that Code Is Available 12-16  
Accessing a Web-Based Application 12-17  
Calling an External Web Application 12-18  
Steps to Open an External Web Application in a New Window Based on a User  
Action in the UI 12-19  
1. Create a String with an Appropriate URL 12-20  
2. Populate the URL from Values from the Applet 12-21  
3. Write JavaScript to Open a Window to the URL 12-22  
4. Call JavaScript When the User Action is Detected in the UI 12-23  
Creating a New Link Dynamically 12-24  
Access an External Web Site with Data from the Siebel Application 12-25  
1. Create a Presentation Model (PM) 12-26  
2. Add a Property and Methods to the PM 12-27  
3. Create a Physical Renderer (PR) 12-28  
4. Add HTML that will be Used as a Link 12-29  
5. Add a Binding 12-30  
6. Apply the New PR / PM 12-32  
7. Test the New Functionality 12-33

Lesson Highlights 12-34

Practice 12-35

### **13 Siebel Business Services and Profile Attributes**

Objectives 13-2

Siebel Business Service 13-3

Accessing a Siebel Business Service 13-4

Make the Service Available to the Application 13-5

Write Code to Access the Service 13-6

Solution Steps 13-8

1. Identify the Service, its Methods, and the Parameters for Each Method 13-9

2. If Necessary, Configure the Application to Use the Business Service 13-11

3. Write Code for the Business Requirement 13-12

3. Write Code in a Custom Postload Handler 13-13

4. Apply the New Code to the Application 13-20

5. Test Your Change 13-21

Lesson Highlights 13-22

Practice 13-23

### **14 Using Siebel Tools with Siebel Open UI**

Objectives 14-2

Add Presentation Model Properties to a View, Applet, or Control 14-3

Presentation Model Properties as User Properties 14-4

Read Presentation Model Properties in the Presentation Model Code 14-6

1. Load the Siebel Constants 14-7

2. Get the Presentation Model Properties 14-8

3. Parse the Property Set 14-9

Complete Code Example: Applet 14-10

Result 14-11

Special Property: EnableDragAndDropInList 14-12

Enable Drag-and-Drop Imports 14-13

1. Add the User Property to the Applet 14-14

2. Prepare the Spreadsheet 14-15

3. Test the Results 14-16

Lesson Highlights 14-18

Practice 14-19

### **15 User Preferences**

Objectives 15-2

User Visualization 15-3

User Visualization in Siebel Open UI 15-4

Selecting Default User Visualization	15-5
Configuring Visualizable Applets	15-6
Visualizable Web Template	15-7
Physical Renderer and Presentation Model	15-8
Create a Visualization Model	15-10
1. Add the Web Templates to the Applet	15-11
2. Edit the Web Layouts	15-12
3. Administer the Web Templates	15-13
4. Administer the Manifest	15-14
5. Test the Results	15-15
Additional Visualizations	15-16
Other User Preferences	15-17
User Preferences	15-18
Read a User Preference	15-19
Set a User Preference	15-20
Example: A Button that Hides and Shows Applets	15-22
1. Add the Button to the User Interface	15-23
Verify the Button	15-24
2. Add the Script that Reacts to the Button	15-25
3. Test the Results	15-28
Lesson Highlights	15-29
Practice	15-30

## **16 Deploying to a Production Environment**

Objectives	16-2
Technical Challenge	16-3
Technical Solution: Migrate Customizations to the Server	16-4
1. Use Standard Techniques to Migrate Siebel Repository Changes	16-5
2. Copy the Files	16-6
File Considerations	16-7
3. Administer the Manifest for the Enterprise	16-8
Manifest Considerations	16-9
4. Test the Results	16-10
Lesson Highlights	16-11
Practice	16-12

