



Using Siebel Tools

Siebel Innovation Pack 2017, Rev. A
March 2018

Copyright © 2005, 2018 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, 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.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Contents

Chapter 1: What's New in This Release

Chapter 2: Using Siebel Tools

Overview of Using Siebel Tools	15
Roadmap for Setting Up and Using Siebel Tools	16
Process of Setting Up the Development Environment	16
Using the Object Explorer and Object List Editor	20
Overview of the Object Explorer and Object List Editor	20
Locating and Modifying Object Definitions in the Object List Editor	23
Displaying Object Types in the Object Explorer	25
Using Lists and Tabs in the Object Explorer	26
Using the Object List Editor	28
Using Menus and Running Queries	30
Using the Menu Bar	30
Using a Right-Click Menu	30
Running Queries	31
Getting Documentation About an Object	33
Using Windows, Wizards, and Toolbars	34
Using the Properties Window	34
Using the Applets Window	35
Using the Controls/Columns Window	36
Using the Palettes Window	36
Using Web Templates	36
Using the Multi Value Property Window	37
Using the Bookmarks Window	37
Displaying, Docking, and Stacking Windows	37
Using a New Object Wizard	39
Using the Toolbars	40
Using Editors in Siebel Tools	41
Using a Layout Editor	41
Using an Editor That Includes a Canvas	42
Using a Script Editor	44
Using the Command Line	44

Chapter 3: Using Web Tools

About Web Tools	47
Benefits of Web Tools	48
About Delivering in Siebel Tools and Web Tools	48
Siebel Tools Features Included in Web Tools	48
Siebel Tools Objects Included in Web Tools	49
Siebel Tools Features Excluded From Web Tools	50
Siebel Tools Objects Excluded From Web Tools	50
	50

Chapter 4: Using Workspaces in Siebel Tools

Overview of the Workspaces Mode for Siebel Tools	51
Workspaces Overview	51
Workspace Users	52
Workspace Dashboard in Siebel Tools	53
Using Workspaces in Siebel Tools	56
Creating New Workspaces	56
Opening Existing Workspaces	57
Editing Workspace-Enabled Repository Objects	58
Deleting Development Workspaces	59
Refreshing the Workspace Explorer Pane	60
Tracking Repository Object Changes in Workspaces	60
Performing the Checkpoint Version Process	61
Reverting to Previous Workspace Versions	62
Rebasing Workspaces	62
Detecting Conflicts in Workspaces and Applying Resolutions	64
Submitting Workspaces for Delivery	67
Canceling the Workspace Delivery Process	68
Delivering Workspaces	69
Exporting Workspace Objects to Archive Files	69
Comparing Workspace Versions	70
Flattening Workspace Versions	70
Configuring Non-Workspace Objects	71
Using Workflows in Workspaces	71
Using Workflows in Workspaces Overview	72
Creating New Workflows	73
Modifying Existing Workflows	74
Importing Workflows into Workspaces	74

Exporting Workflows from Workspaces	75
Validating Workflows	75
Debugging Workflows	76
Invoking Workflows	77
Delivering Workspaces that Use Workflows	77
Workspaces Administration	77
Enabling the Workspaces Mode for Siebel Tools	78
Enabling Workspaces in Siebel Tools and Editing the Repository Objects	81
Canceling the Workspace Delivery Process	82
Delivering Workspaces	83
Flattening Workspace Versions	85
Controlling Access on Non-Workspace Objects	87
Publishing Tables	88
Using Workspaces for Seed Data	89
Support for NON-ENU Language in Rich Text Control	92
Adding the Workspace Prefix in System Preferences	94
Chapter 5: Setting up Web Tools	
HTML Tags Migrated to Web Tools	97
Manually Running the Migration Script	97
Troubleshooting and Verifying the Results of the Web Template Migration Process	98
Publishing Changes to Siebel Web Templates	99
Workspaces Dashboard in Web Tools	99
Enabling Workspaces in Web Tools and Editing the Repository Objects	102
Navigating to the Workspace Dashboard in the Web Tools Mode Application	103
106	
Chapter 6: Parallel Development Using Workspaces	
Administering Parallel Development using Workspaces	107
Enabling the Parallel Development Feature for Siebel Tools Workspaces	108
Understanding Integration Workspaces in Siebel Tools	109
Creating Integration Workspace in the Workspaces Dashboard	110
Creating Integration Workspace Using Siebel Tools	111
Setting an Integration Workspace as the Default Workspace Branch	112
Setting System Preferences for Parallel-Development Workspaces	112
Launching the Web Client with a Specified Workspace	114
Using the Parallel Development Tools for Workspaces	114

- Applying Parallel Development Using Workspaces Hierarchy 115
- Understanding Development Workspaces in Siebel Tools 115
- Creating Development Workspaces in the Workspaces Dashboard 116
- Creating Development Workspaces Using Siebel Tools 117

Chapter 7: Using Siebel Script Editors

- Using the Siebel Script Editor 119
 - Overview of Using Siebel Script Editor 119
 - Setting Options for the Siebel Script Editor 121
 - Opening the Siebel Script Editor 122
- Using the ST eScript Engine 123
 - Overview of Using the ST eScript Engine 123
 - Enabling the ST eScript Engine 127
 - Setting Options for the ST eScript Engine 127
 - Opening the Script Assist Window 128
 - Using Fix and Go 128
 - Using Running Tool Tip 129
 - Using Script Libraries with the ST eScript Engine 130
- Using the Siebel Debugger 132
 - Overview of Using the Siebel Debugger 132
 - Setting Debug Options 133
 - Accessing the Siebel Debugger 134
 - Validating Script Syntax 134
 - Using Breakpoints 135
 - Using the Calls Window 135
 - Using the Watch Window While it Monitors a Script 136
 - Tracing a Script 136
- Using the Script Profiler 138
 - Overview of the Script Profiler 139
 - Enabling the Script Profiler and Line Profiler 141
 - Running the Script Profiler 142

Chapter 8: Customizing Objects

- Overview of Customizing Objects 145
- Creating, Modifying, Copying, Validating, and Deleting Objects 145
 - Creating Objects 146
 - Modifying Objects 147
 - Copying Objects 147
 - Validating Objects 148
 - Deleting Objects 149

Examining Objects	150
Searching the Repository	150
Viewing Object Relationships	152
Comparing and Synchronizing Objects Between Repositories and Archives	153
Comparing Different Versions of a Workflow Process or Task UI	156
Determining When Siebel CRM Created or Updated a Record	157

Chapter 9: Managing Repositories

Viewing, Renaming, Comparing, and Configuring Repositories	159
Overview of Managing Repositories	159
Files That You Use to Manage Repositories	160
Viewing Information About the Current Repository	160
Renaming Repositories	162
Configuring Siebel CRM to Read Data from a Single Repository	163
Restricting the Objects That Developers Can Modify	165
Adding Identification Numbers to Repository Modifications	168
Exporting and Importing Repositories	169
Overview of Exporting and Importing Repositories	169
Importing Repositories in Windows Environments	171
Importing Repositories in UNIX Environments	172
Exporting Repositories	173
Importing or Exporting Repositories at a Later Time	173
Upgrading Repositories	173
Identifying Conflicts That Occur During Upgrades	174
Process of Migrating Repositories	176
Preparing to Migrate Repositories	176
Using the Database Configuration Wizard to Migrate Repositories	177
Updating Siebel Remote Databases	180
Migrating Non-repository Configurations and Data	181
Using the Repository Import and Export Utility	183
Guidelines for Migrating Repositories	185
Deleting Repositories	186

Chapter 10: Localizing Strings and Locale Data

Configuring Symbolic Strings	189
Overview of Configuring Symbolic Strings	189
Modifying the Configuration File to Support Symbolic Strings	191
Creating a Project for Your Symbolic Strings	191
Modifying a Predefined Symbolic String	192
Creating a New Symbolic String	192

- Modifying a Symbolic String to Globally Update Display Values 193
- Setting a Symbolic String Reference 194
- Entering a String Override 196
- Converting Symbolic Strings 197
 - Consolidating Symbolic Strings 202
 - Using a Batch File to Convert Strings 204
- Configuring Locale Data 205
 - Configuring Nontranslatable Locale Object Properties 205
 - Displaying Controls and List Columns According to Locale 206
 - Fixing Orphaned String References After an Upgrade 208
- Using the Locale Management Utility 209
 - Finding Untranslated Text Strings 210
 - Finding Existing Translations 211
 - Finding Objects That the Locale Management Utility Modifies 212
 - Finding Objects Modified Since the Last Export 212
 - Exporting Text Strings and Locale Properties to a File 214
 - Importing Text Strings and Locale Properties 215
 - Modifying the Value in All Strings for a Language 216
 - Using the Command Line to Run the Locale Management Utility 216

Chapter 11: Testing and Troubleshooting Your Customizations

- Testing and Troubleshooting Your Modifications 219
 - Setting Debug Options to Open the Siebel Client 219
 - Recovering from a Failed Development Server 220

Chapter 12: Archiving Objects

- Overview of Archiving Objects 221
- Exporting Objects to an Archive 222
 - Using the Command Line to Export Objects to an Archive 223
- Importing Objects from an Archive 223
 - Preparing the Siebel Tools Environment to Import Objects from an Archive 224
 - Importing an Archive 224
 - Importing Multiple Archives 226
 - Using the Command Line to Import an Archive 227
 - Using the Review Conflicts and Actions Dialog Box of the Import Wizard 229
- Using the Application Deployment Manager 230

Index

Glossary

Appendix A: Reference Materials for Siebel Tools

Menus and Menu Items on the Menu Bar	243
File Menu	244
Edit Menu	245
View Menu	246
Screens Menu	248
Go Menu	249
Query Menu	249
Format Menu	250
Debug Menu	250
Tools Menu	251
Workspace Menu	254
Window Menu	255
Help Menu	256
Buttons on the Toolbars	257
Edit Toolbar	257
List Toolbar	258
History Toolbar	259
Debug Toolbar	260
Simulate Toolbar	261
Format Toolbar	261
WF/Task Editor Toolbar	263
Configuration Context Toolbar	263
Buttons in the Palettes Window	264
Dialog Boxes That Validate Objects	265
Elements of the Validate Dialog Box	266
Elements of the Validation Options Dialog Box	267
Dialog Boxes That Compare Objects	268
Elements of the Compare Objects Dialog Box	268
Dialog Boxes That Set Development Options	269
Development Options for Visualization Views	270
Development Options for Scripting	270
Development Options for Debugging	272
Parameters That Convert Symbolic Strings	273
Parameters You Use with the Conversion Export Utility	274
Parameters That You Can Use with the Conversion Import Utility	275

1

What's New in This Release

What's New in Using Siebel Tools, Innovation Pack 2017, Rev. A

NOTE: Siebel Innovation Pack 2017 is a continuation of the Siebel 8.1/8.2 release.

No new features have been added to this guide. The following topics were revised to improve the technical accuracy of this guide:

- ["Using Workspaces in Siebel Tools" on page 56](#). Certain sections were updated to reflect the user interface in Siebel Tools and Web Tools.

Additional Changes

This book also contains the following general changes:

- The term Inspect has been retained in the book only where it references Siebel application.
- This guide has been updated to correct or remove obsolete product and component terms.

What's New in Using Siebel Tools, Innovation Pack 2017

[Table 1](#) lists the changes in this revision of the documentation to support this release of the software.

NOTE: Siebel Innovation Pack 2017 is a continuation of the Siebel 8.1/8.2 release.

Table 1. New Product Features in Using Siebel Tools, Siebel Innovation Pack 2017

Topic	Description
"Using Siebel Tools" on page 15	<p>Modified chapter. These sections were moved from this chapter to the Using Workspaces in Siebel Tools chapter:</p> <ul style="list-style-type: none">■ Using Workspaces in Siebel Tools■ Workspaces Administration <p>Updated sections related to the following features as per changes in Web Tools:</p> <ul style="list-style-type: none">■ Object Explorer and Object List Editor■ Menus and Queries■ Web Template applet■ Layout Editors
"Using a New Object Wizard" on page 39	Modified section. It includes a note reminding users to use the Edit Web Layout function to edit the web layout.
"Using Web Tools" on page 47	New chapter. It includes sections that were in the Using Siebel Tools chapter in the previous release.

Table 1. New Product Features in Using Siebel Tools, Siebel Innovation Pack 2017

Topic	Description
<p>“Siebel Tools Features Included in Web Tools” on page 48</p> <p>“Siebel Tools Features Excluded From Web Tools” on page 50</p>	Modified topics. It describes new features included in and excluded from Web Tools.
<p>“Siebel Tools Objects Included in Web Tools” on page 49</p> <p>“Siebel Tools Objects Excluded From Web Tools” on page 50</p>	Modified topic. It lists the objects included in and excluded from Web Tools.
<p>“Using Workspaces in Siebel Tools” on page 51</p>	New chapter. It includes the sections that discuss the Workspaces feature, including the new section that discusses how to use workflow objects in workspaces and the sections that were included in the Using Siebel Tools chapter in the previous release.
<p>“Workspace Users” on page 52</p>	New topic. This topic describes the roles and the responsibilities of workspace users.
<p>“Deleting Development Workspaces” on page 59</p>	Modified topic. It includes a note for administrators when they delete the integration workspaces.
<p>“Configuring Non-Workspace Objects” on page 71</p>	Modified topic. It includes the non-workspace objects that are introduced in Siebel Tools IP2017.
<p>“Using Workflows in Workspaces” on page 71</p>	New section. This section describes workflow objects, and discusses how to manage and use workflow objects in Siebel Tools Workspace.
<p>“Enabling the Workspaces Mode for Siebel Tools” on page 78</p>	Modified topic. It includes the notes, arguments, parameters, and so on, that are new in Siebel IP2017.
<p>“Enabling Workspaces in Siebel Tools and Editing the Repository Objects” on page 81</p>	New topic. It discusses how to enable the workspaces in Siebel Tools and edit the repository objects.
<p>“Flattening Workspace Versions” on page 85</p>	Modified topic. It includes the information to clarify the methods that are used to flatten workspace versions.
<p>“Controlling Access on Non-Workspace Objects” on page 87</p>	Modified topic. It includes the non-workspace objects that are introduced in Siebel Tools IP2017. It also has the notes to clarify the different behaviors of workflow object in Siebel Tools IP2016 and IP2017.
<p>“Publishing Tables” on page 88</p>	New topic. It discusses how to publish table when you use and manage workspaces.
<p>“Using Workspaces for Seed Data” on page 89</p>	New topic. It discusses how to enable Siebel Workspace for the seed data.

Table 1. New Product Features in Using Siebel Tools, Siebel Innovation Pack 2017

Topic	Description
“Support for NON-ENU Language in Rich Text Control” on page 92	New topic. It discusses how to enable the support for NON-ENU language in rich text control.
“Setting up Web Tools” on page 97	Deleted topics. The following topics were deleted due to changes in enabling the application component and high number of SWE tags being migrated: <ul style="list-style-type: none"> ■ About Setting Up Web Tools ■ Process of Setting Up Web Tools ■ SWE Tags Migrated to Web Tools ■ Enabling the Composer Mode for the Selected and Opened Workspace to Edit Objects ■ Setting up Web Tools Application Edit ■ Information in Siebel Web Template Files Mapped to Fields in Web Tools
“Manually Running the Migration Script” on page 97	Modified topic. It lists the steps required for migration to Web Tools.
“Workspaces Dashboard in Web Tools” on page 99	New section. This section illustrates and describes the Workspace dashboard in Web Tools.
“Enabling Workspaces in Web Tools and Editing the Repository Objects” on page 102	New section. It discusses how to enable workspaces in Web Tools and edit the repository objects.
“Navigating to the Workspace Dashboard in the Web Tools Mode Application” on page 103	Modified topic. It includes the workspace references in the example command lines and in the steps that are used to enable the Workspace feature in the Web Tools mode.
“Parallel Development Using Workspaces” on page 107	New chapter. It describes the Parallel Development Using Workspaces feature and how to use this feature for the Siebel Tools workspaces.
“Using a Wizard to Create Objects” on page 146	Modified section. It includes the information regarding the steps required after the view and applet are created using wizard.
“Deleting Repositories” on page 186	Modified section. Starting from Siebel Tools IP2017, users can use the RRCleanup utility to delete the repositories. This section includes the steps used to run the RRCleanup utility.
“Archiving Objects” on page 221	Modified chapter. It includes the information about BatchExport that is used to export objects to SIR file. Also, workspace name is mentioned in the example command lines.
“Exporting Objects to an Archive” on page 222	Modified topic. Updated topic to include archiving in Web Tools.

Table 1. New Product Features in Using Siebel Tools, Siebel Innovation Pack 2017

Topic	Description
“Using the Command Line to Export Objects to an Archive” on page 223 “Using the Command Line to Import an Archive” on page 227	Modified topics to include the command lines that reference workspace names.
“Example of Using the Command Line to Import an Archive” on page 228	Modified topic. The command lines are updated to include workspace references.
Specifying the Web Template Editor	Deleted topic. This editor is no longer supported in Siebel Tools and is part of the layout editor in Web Tools.
About the User Interface for Web Tools	Deleted topic. The user interface for Web Tools is covered in chapters on Siebel Tools, Web Tools, and Workspaces.
Using the Command Line to Compile Objects	Deleted section. Siebel Tools IP2017 no longer requires the Siebel Repository File (SRF) to start Siebel application.
Using the Command Line to Export Objects to a Hotfix	Deleted section. This section is no longer applicable in Siebel Tools starting from IP2017.
Checking Out and Checking In Projects and Objects Appendix A: Dialog Boxes That Check Out and Check In Projects and Objects	Deleted chapter and topic. Check-out and check-in are no longer supported, starting from Siebel Tools IP2017.

Additional Changes

This book also contains the following general changes:

- Content from the *Siebel Composer Developer Preview Guide* for Innovation Pack 2016 was merged into this guide.
- The following terms were changed throughout the book:
 - *Web Tools* replaces *Siebel Composer*.
 - *Deliver* replaces *Publish* wherever your workspace changes are merged into parent workspace.

2

Using Siebel Tools

This chapter describes how to use Oracle's Siebel Tools. It includes the following topics:

- [Overview of Using Siebel Tools on page 15](#)
- [Roadmap for Setting Up and Using Siebel Tools on page 16](#)
- [Using the Object Explorer and Object List Editor on page 20](#)
- [Using Menus and Running Queries on page 30](#)
- [Using Windows, Wizards, and Toolbars on page 34](#)
- [Using Editors in Siebel Tools on page 41](#)
- [Using the Command Line on page 44](#)

Overview of Using Siebel Tools

Siebel Tools is an integrated development environment that you can use to customize Siebel CRM. You can use it to modify predefined Siebel objects or to create custom objects that meet your business requirements. For example, you can use Siebel Tools to modify the data model, modify business logic, or customize the user interface that the Siebel client shows. Siebel Tools allows you to develop a single configuration that you can use to do the following:

- Deploy Siebel CRM across multiple types of clients
- Support multiple Siebel Business Applications and languages
- Upgrade Siebel product releases
- Maintain Siebel CRM

Declarative configuration is a type of programming technique that uses objects and object properties in the Siebel repository to implement the logic that your business requires. Siebel Tools uses declarative configuration to create and modify the object definitions that define a Siebel application. Siebel Tools is not a programming environment. You do not modify the source code or write SQL. Siebel CRM uses the terms object and object definition differently than developers who use programming languages that use similar terms, such as object, object class, or object instance. For more information about the Siebel repository, see [Chapter 9, "Managing Repositories."](#) For more information about the objects, object definitions, and the object hierarchy that Siebel CRM uses, see *Configuring Siebel Business Applications*.

NOTE: The *Siebel Bookshelf* is available on Oracle Technology Network (<http://www.oracle.com/technetwork/indexes/documentation/index.html>) and Oracle Software Delivery Cloud. It might also be installed locally on your intranet or on a network location.

Roadmap for Setting Up and Using Siebel Tools

To set up and use Siebel Tools, do the following tasks:

- 1 [Process of Setting Up the Development Environment on page 16](#)
- 2 [Creating, Modifying, Copying, Validating, and Deleting Objects on page 145](#)
- 3 [Testing and Troubleshooting Your Modifications on page 219](#)
- 4 [Exporting Objects to an Archive on page 222](#)

Process of Setting Up the Development Environment

To set up the development environment, do the following tasks:

- 1 [Enabling Workspaces in the Development Environment on page 16](#)
- 2 [Installing Siebel Tools on page 16](#)
- 3 [Choosing the Browser That the Applet Layout Editor Uses on page 17](#)
- 4 [Controlling How Siebel Tools Handles Versions for a Workflow Process or Task UI on page 18](#)
- 5 [Setting the Language Mode on page 18](#)
- 6 [Controlling How Siebel Tools Displays the Confirmation Dialog Box on page 19](#)
- 7 [Resetting Development Tools Options on page 19](#)

This process is a step in [“Roadmap for Setting Up and Using Siebel Tools” on page 16](#).

Enabling Workspaces in the Development Environment

In order to set up the Siebel development environment, the Siebel Administrator has to enable workspaces. For more information about how to enable Workspace for Siebel Tools, see [“Enabling the Workspaces Mode for Siebel Tools” on page 78](#).

Installing Siebel Tools

In this guide, *SIEBEL_TOOLS_ROOT* represents the folder where you install Siebel Tools. This folder is C: \Siebel \release_number\Tools, by default.

For more information about:

- How to install Siebel Tools, see the *Siebel Installation Guide* for the operating system you are using. For example, see *Siebel Installation Guide for Microsoft Windows*.

- System requirements, such as supported versions of Microsoft Windows, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

NOTE: For Siebel CRM product releases 8.1.1.9 and later and for 8.2.2.2 and later, the system requirements and supported platform certifications are available from the Certifications tab on My Oracle Support. For information about Certifications, see article 1492194.1 (Article ID) on My Oracle Support.

This task is a step in [“Process of Setting Up the Development Environment”](#) on page 16.

Choosing the Browser That the Applet Layout Editor Uses

You can choose the browser that Siebel Tools uses to display applets in the preview mode of the Applet Layout Editor. You can include conditional tags in Web templates that it shows for some browsers but not for others. Choosing a browser allows you to preview how Siebel CRM displays an applet layout for a specific browser. For more information, see [“Using Editors in Siebel Tools”](#) on page 41.

This task is a step in [“Process of Setting Up the Development Environment”](#) on page 16.

To choose the browser that the Applet Layout Editor uses

- 1 In Siebel Tools, click the View menu, click Toolbars, and then click Configuration Context.

For more information, see [“Using the Toolbars”](#) on page 40.

- 2 On the Configuration Context toolbar, click the Target Browser drop-down list, and then click Target Browser Config.

Siebel Tools displays the Target Browser Configuration dialog box, which includes the sections in the following table.

Section	Description
Available Browsers	Lists browsers that are available, such as Internet Explorer and Mozilla.
Selected Browsers for Layout Editing	Lists browsers that the Web Layout Editor affects.
Capability Name and Value	Lists the capabilities and values for the browser that you choose in the Selected Browsers for Layout Editing section.

- 3 To add a browser to the list browsers, click it in the Available Browsers list, and then click the right arrow.

You can also double-click a browser in the Available Browsers list.

- 4 Click OK.

Siebel Tools displays each browser that you add as a value in the Target Browser drop-down list.

Controlling How Siebel Tools Handles Versions for a Workflow Process or Task UI

You can control how Siebel Tools handles versions for a workflow process or task UI that you open with an editor, depending on the value of the Status property.

This task is a step in [“Process of Setting Up the Development Environment”](#) on page 16.

To control how Siebel Tools handles versions for a workflow process or task UI

- 1 In Siebel Tools, click the View menu, and then click Options.
- 2 In the Development Tools Options dialog box, click the General tab.
- 3 In the Workflow and Task Configurations section, set the options by using the descriptions in the following table, and then click OK.

Option	Description
Automatic Revision in WF/Task Editor and Version Check	<p>If this check box:</p> <ul style="list-style-type: none"> ■ Includes a check mark, and if you use an editor to open a workflow process or task UI, and if the Status property of this workflow process or task UI is Completed, then Siebel Tools creates a copy of this workflow process or task UI, sets the status for this copy to In Progress, and then opens an editable version of it in the editor. ■ Does not include a check mark, then Siebel Tools opens the editor, and then displays a version of the workflow process or task UI that you cannot edit.
Automatically Close All the Previous WF/Task Versions	<p>If this check box includes a check mark, and if you use an editor to open a workflow process or task UI, and if the Status property of this workflow process or task UI is Completed, Not In Use, or Expired, then Siebel Tools closes any prior versions of this workflow process or task UI.</p>

Setting the Language Mode

The *language mode* is a type of mode that allows you to configure Siebel CRM to display text in a language other than English. For example, the German (DEU) language mode allows you to view text in some objects of Siebel Tools in German, such as the text that a string contains or object definitions that the Object List Editor shows. The language mode determines the set of locale data that Siebel Tools uses when it compiles the repository, and during checkin and checkout. If you add a language to the Siebel database that does not come predefined with Siebel CRM, then this language must use all capital letters. For more information, see [“About Predefined Objects”](#) on page 23 and *Siebel Global Deployment Guide*.

This task is a step in [“Process of Setting Up the Development Environment”](#) on page 16.

To set the language mode

- 1 Make sure the repository includes the language data that Siebel Tools must show.

- 2 In Siebel Tools, click the View menu, and then click Options.
- 3 In the Development Tools Options dialog box, click the Language Settings tab.
- 4 In the Tools Language Mode section, choose a value from the Language drop-down list, and then click OK.
- 5 (Optional) To enable language override, make sure the Enable and Use Language Override check box includes a check mark.

For more information, see [“Enabling Language Override” on page 19](#).

- 6 Click OK.

Enabling Language Override

A *language override* is a nontranslatable, locale property that you can configure differently for different locales. For example, you can configure Siebel CRM to use a specific height for the address field in French and to use a different height in English. If you enable language override, then Siebel CRM might create more locale records in the repository. To avoid unnecessary records, it is recommended that you enable language override only if your deployment requires it. For more information, see [“Configuring Nontranslatable Locale Object Properties” on page 205](#) and *Configuring Siebel Business Applications*.

Controlling How Siebel Tools Displays the Confirmation Dialog Box

You can control how Siebel Tools displays the confirmation dialog box.

This task is a step in [“Process of Setting Up the Development Environment” on page 16](#).

To control how Siebel Tools displays the confirmation dialog box

- 1 In Siebel Tools, click the View menu, and then click Options.
- 2 In the Development Tools Options dialog box, click the General tab.
- 3 In the Editing Confirmation Dialogs section, configure Siebel Tools to do one of the following:
 - **Display a confirmation dialog box.** Add a check mark to a check box.
 - **Do not display a confirmation dialog box.** Remove a check mark from a check box.
- 4 Click OK.

Resetting Development Tools Options

If Siebel Tools behavior is not consistent with the preferences you set, then you can reset them.

This task is a step in [“Process of Setting Up the Development Environment” on page 16](#).

To reset development tools options

- Do one of the following:
 - Reset preferences in the Development Tools Options dialog box:

- ❑ Click the View menu, and then click Options.
- ❑ In the Development Tools Options dialog box, click the various tabs and reset preferences, as necessary.
- Delete the file that stores your preferences:
 - ❑ Close Siebel Tools.
 - ❑ Navigate to the `SIEBEL_TOOLS_ROOT\BIN` folder, and then delete the `devtools.prf` file.
 - ❑ Log in to Siebel Tools, click the View menu, and then click Options.
 - ❑ Set preferences in the Development Tools Options dialog box.

Siebel Tools stores the preferences that you set in the Development Tools Options dialog box. It stores these preferences in the `devtools.prf` file. If you delete this file, then Siebel Tools resets all your preferences to their default values.

Using the Object Explorer and Object List Editor

This topic describes how to use the Object Explorer and Object List Editor. It includes the following information:

- [Overview of the Object Explorer and Object List Editor on page 20](#)
- [Locating and Modifying Object Definitions in the Object List Editor on page 23](#)
- [Displaying Object Types in the Object Explorer on page 25](#)
- [Using Lists and Tabs in the Object Explorer on page 26](#)
- [Using the Object List Editor on page 28](#)

Overview of the Object Explorer and Object List Editor

NOTE: The *Object Explorer* is available in both Siebel Tools and Web Tools.

The *Object Explorer* is a window that displays the Siebel object hierarchy. It allows you to use a tree structure to navigate through the object types that the hierarchy contains. The *Object List Editor* is a window that displays the object definitions of the object type that you choose in the Object Explorer. It allows you to view and modify the object definitions.

How Siebel Tools Displays Object Types and Object Definitions

Figure 1 illustrates how Siebel Tools includes an Object Explorer window and one or more Object List Editor windows. In this figure, the Applications list in the Object List Editor is shown. For more information about the object hierarchy that Siebel CRM uses, see *Configuring Siebel Business Applications*.

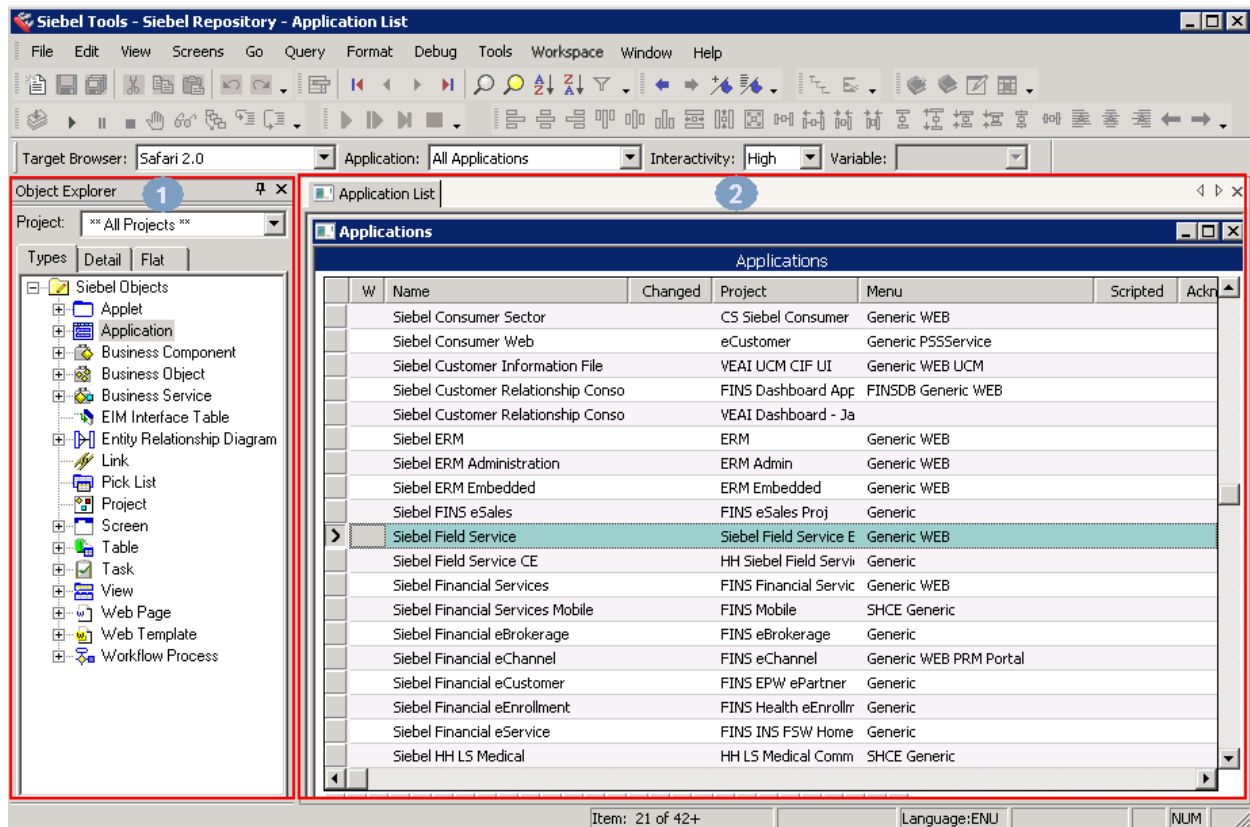


Figure 1. Object Explorer and Object List Editor

Explanation of Callouts

Figure 1 includes the following items:

- 1 Object Explorer.** A window that displays object types. Allows you to navigate between each group of object definitions of an object type.
- 2 Object List Editor.** A window that displays object definitions. For more information, see [“Using the Object List Editor” on page 28](#).

An *object type* is an entity that includes a predefined set of properties. You can use it as a template to create an object definition. An application is one kind of object type. For more information, see [“About Predefined Objects” on page 23](#).

An *object definition* implements one piece of the software. This object definition consists of *object properties*, which are characteristics of this piece of the software. The Object Explorer and Object List Editor display all the object definitions that the repository contains. Siebel Field Service is an example of an object definition of an application. Siebel CRM uses the terms *attribute* and *property*. These terms have the same meaning.

If an object type is not visible in the Object Explorer, then you can display it. For more information, see “[Displaying Object Types in the Object Explorer](#)” on page 25.

NOTE: Web Tools follows a similar arrangement with object types listed in Object Explorer and their corresponding object definitions in Object List Editor.

How Siebel Tools Displays the Siebel Object Hierarchy

Figure 2 illustrates a *parent and child relationship*, which is a type of hierarchical relationship between one object type and another object type. In this figure, the Page Tabs object definitions for the Siebel Field Service application are shown.

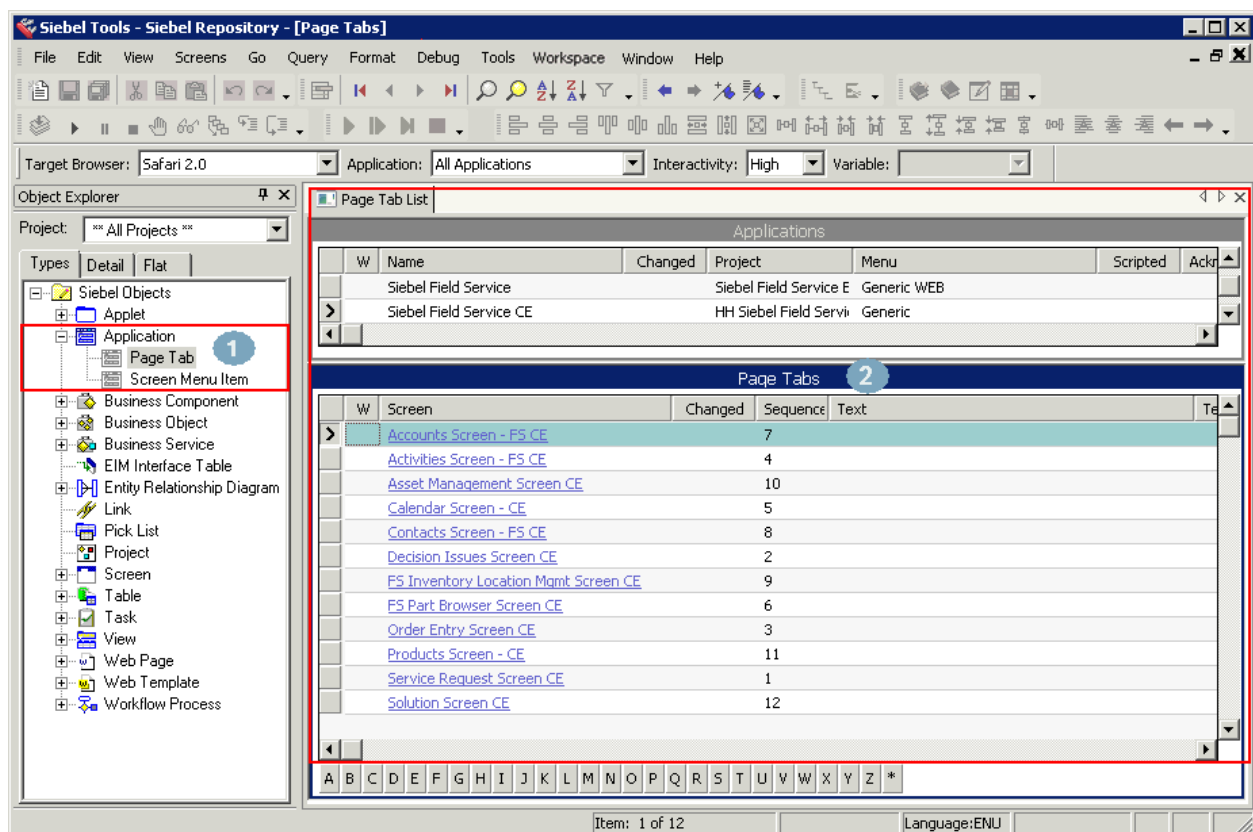


Figure 2. How Siebel Tools Displays Relationships Between Object Types

Explanation of Callouts

Figure 2 includes the following items:

- 1 **Object type hierarchy.** In this example, the Page Tab object type and the Screen Menu Item object type are child objects of the parent Application object type. The Object Explorer displays this relationship as a hierarchical tree that you can expand or collapse. For more information, see [“Using the Types Tab” on page 26](#).
- 2 **Object definition hierarchy.** The Object List Editor uses two windows to display this parent and child relationship. In this example, the Applications list displays the object definition for the parent Siebel Field Service application. The Page Tabs list displays the object definitions of the child page tabs that the repository contains for the Siebel Field Service application.

If you click the Types tab in the Object Explorer, then Siebel Tools arranges folder icons in a hierarchy. An object type that the Object Explorer shows succeeding another object type and indented after it is the child in a parent and child relationship. The object type that the Object Explorer shows preceding the child object type is the parent. A parent object type can include multiple child object types.

NOTE: Web Tools follows a similar hierarchy. When you click an object type in Object Explorer, the object displays in Object List Editor in two applets as the parent and its child.

About Predefined Objects

This guide uses the term *predefined* to describe an object or configuration that comes already defined when you first install Siebel CRM or Siebel Tools. Each object that the Object List Editor shows after you install Siebel Tools, but before you make any modification, is a predefined object. A *custom object* is a new object that you create. A *modified object* is a predefined object or custom object that you modified.

Locating and Modifying Object Definitions in the Object List Editor

The example in this topic locates the object definitions for a single object type in the Object List Editor, and then modifies the value in a property of this object definition. You use the Object Explorer and Object List Editor to modify the Comment property of the Siebel Call Center application, and then examine the page tabs that this application uses. For information about locating object definitions for multiple object types, see [“Searching the Repository” on page 150](#).

To locate and modify object definitions in the Object List Editor

- 1 Log in to Siebel Tools.
- 2 Click the View menu, and then click Object Explorer.

You can also press CTRL+E.

- 3 In the Object Explorer, click Application.

For more information, see [“How Siebel Tools Displays Object Types and Object Definitions” on page 21](#).

- 4 In the Applications list, query the Name property for Siebel Call Center.

For more information, see [“Using the Query Menu to Run a Query” on page 31](#).

If the object definition exists, then the Object List Editor displays it in the Applications list. Querying for a record in this way causes Siebel Tools to display a single, isolated record. This technique helps to make sure that you choose the correct object definition and that it remains chosen while you use the Object Explorer and Object List Editor to navigate the object hierarchy, or if you do other work, such as delivering or revising. It reduces the possibility that you might mistakenly modify another object definition. For more information, see [“How Siebel Tools Displays Object Types and Object Definitions” on page 21](#).

- 5 Right-click the Siebel Call Center record, and then click Lock Object.

NOTE: In Web Tools, locking a project is still required before you edit the non-workspace objects. The non-workspace objects are Table, Task, Workflows, Repository, Type, EIM Table, Projects, Dock objects, Schema Maintenance, and Server Components. For more information, see [“Using Workspaces in Siebel Tools” on page 56](#).

- 6 Locate the property you want to modify, and then type a new value in this property, choose a value from a drop-down list, or add a check mark to a check box.

In this example, locate the Comments property. Use the scroll bar at the end of the Object List Editor to scroll toward the end.

- 7 Enter the following text in the Comments property:

This is the object definition for the Siebel Call Center application.

For more information, see [“Using the Object List Editor” on page 28](#).

- 8 To save your modifications, click anywhere outside of the row.
- 9 In the Object Explorer, expand the Application tree, and then click Page Tab.
- 10 Notice the records that the Object List Editor shows in the Page Tabs list.

For more information, see [“How Siebel Tools Displays the Siebel Object Hierarchy” on page 22](#).

Modifying Multiple Records in the Object List Editor

You can modify multiple records in the Object List Editor of Siebel Tools.

To modify multiple records in the Object List Editor

- 1 In the Object List Editor, locate the objects you want to modify.
- 2 To choose multiple records, hold down the CTRL key while you click each record you want to modify.
- 3 Click the Edit menu, and then click Change Records.
- 4 In the Change Selected Records dialog box, choose the field you want to modify, and then enter a value for that field.
- 5 Click OK.

Displaying Object Types in the Object Explorer

This topic describes how to display object types in the Object Explorer.

To display object types in the Object Explorer

- 1 Log in to Siebel Tools.
- 2 Click the View menu, and then click Options.
- 3 Click the Object Explorer tab.

NOTE: In Web Tools, click the pencil icon from the menu before the Object Explorer tabs.
- 4 Scroll down through the Object Explorer Hierarchy window until you locate the object type you want to display.
- 5 Make sure the object type you want to display includes a check mark, and use the values from the following table.

Check Box State - Web Tools	Check Box State - Siebel Tools	Description
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Display this object and all child objects of this object.
<input type="checkbox"/>	<input type="checkbox"/>	Hide this object and all child objects of this object.
<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Display this object. Display some child objects and hide some child objects.

If you add a check mark to a top-level object type, such as Applet, then Web Tools and Siebel Tools add a check mark to all child objects that the Applet contains. You can expand the parent tree to display only some child objects. A *top-level object type* is an object type at the start of the object hierarchy. The object types that the Object Explorer shows immediately after you install Siebel Tools or Web Tools, but before you expand any object types in the Object Explorer, are top-level object types.

- 6 (Optional) To restore the Object Explorer to the settings when you first install Siebel Tools or Web Tools, click Default.

NOTE: In Web Tools, click the Refresh icon to restore the default settings.

- 7 Click OK.

NOTE: In Web Tools, click Save.

Displaying Hidden Object Types and Properties

This topic describes how to display hidden object types and properties. A *hidden object type* is an object type that Siebel CRM does not show in the Siebel Web Client. For more information, see *Siebel Object Types Reference*.

To display hidden object types and properties

- 1 Close Siebel Tools.
- 2 Use a text editor to open the tool s. cfg file.
- 3 In the Siebel section, set the value of the following parameter to All:
`ClientConfigurationMode`
- 4 Save the file, and then restart Siebel Tools.

Using Lists and Tabs in the Object Explorer

This topic describes how to use lists and tabs in the Object Explorer.

Using the Project Drop-Down List

The Object Explorer includes the Project drop-down list that allows you to filter objects according to project. The example in this topic describes how to use the Project drop-down list so that the Object Explorer displays only the object types that reference the Account project.

NOTE: In the Object Explorer pane of Web Tools, most of the object types are workspace-enabled, except the following ones: Table, Task, Repository, Type, EIM Table, and Project. For more information on behavior of Project in Web Tools, see [“Using Workspaces in Siebel Tools” on page 56](#).

To use the Project drop-down list in Siebel Tools

- 1 In the Object Explorer, click the Project drop-down list.
- 2 Choose Account.

You can choose All Projects in the Project drop-down list to reset the Object Explorer so that it displays objects types for all projects.

Using the Types Tab

Siebel Tools displays the Types tab, by default. You can use it to navigate the object hierarchy. The Object Explorer does not display all object types, by default. For more information, see [“Displaying Object Types in the Object Explorer” on page 25](#). For more information about the object hierarchy, see *Configuring Siebel Business Applications*.

To use the Types tab

- 1 In the Object Explorer, click the Types tab.
Siebel Tools displays the top-level object types in alphabet order.
- 2 To display child object types, expand the tree for an object type.
To expand the tree, you click the plus sign (+) that the Object Explorer shows next to the object type. For example, expand the Business Component tree.

- 3 To collapse an object type, click the minus sign (–).

NOTE: In Web Tools, click the arrow sign to expand and collapse the Object Explorer tree.

Using the Detail Tab

The Object Explorer includes a Detail tab that allows you to view and expand an object type. If you use the Detail tab, then Siebel Tools displays object properties in the Object List Editor only for the object that you choose.

NOTE: Web Tools has only the Type and Flat tabs.

Using the Flat Tab

The Object Explorer includes a Flat tab that displays all object types in a single, alphabetic list. It does not display a hierarchy. You can use it to do the following:

- Find a child object with an unknown parent. For example, you create a new field but do not remember the business component that this field references. You can click the Flat tab, click the Field object type, and then query the Name property for the field name. Each record that Siebel Tools returns includes a parent property that displays the business component name.
- Determine how Siebel CRM typically uses objects and properties, such as how it uses a predefault value or the format that it uses in a calculated field.

To use the Flat tab

- 1 In the Object Explorer, click the Flat tab.

The Object Explorer displays a list of all the object types in the repository. It displays this list in alphabetic order and with no hierarchy.

- 2 In the Object Explorer, click Field.

The Object List Editor displays all the object definitions in the repository for the field object. This example describes how to use the flat tab to identify all the business components that include a field named Account Address.

- 3 In the Fields list, query the Name property for Account Address.

The Object List Editor displays a list of all the field object definitions in the repository with the name Account Address.

- 4 Right-click in the Fields list.

NOTE: In Web Tools, use the Columns Displayed menu from the gear icon or press CTRL+SHIFT+K.

- 5 In the Columns Displayed dialog box, scroll to the end of the Displayed Columns window, choose Parent Business Component, click the move to first arrow, and then click OK.

The Parent Business Component column in the Object List Editor lists all the business components that include a field named Account Address.

- 6 In the Parent Business Component property, click Order Entry - Orders.

The Object List Editor displays the object definition for the Order Entry - Orders business component.

Using the Object List Editor

You can use the Object List Editor to edit the properties of an object definition. For example, if you choose the Application object type in the Object Explorer, then the Object List Editor displays a list of all the applications that the repository contains. One row in the Object List Editor represents one object definition. For example, the values that the Object List Editor shows in the properties of the Siebel Call Center application represent one object definition.

The Object List Editor displays each object property as a column in the list. Information that you enter in each column represents a *property value*. You can modify the property values in an object definition. You cannot modify the set of properties that constitute an object definition. You can also use the Properties window to edit these properties. Modifying the value of a property in the Properties window also modifies the corresponding value of that property in the Object List Editor window. For more information, see [“Using the Properties Window” on page 34](#).

Controlling How Siebel Tools Displays the Object List Editor

You can control how Siebel Tools displays the Object List Editor.

To control how Siebel Tools displays the Object List Editor

- 1 In Siebel Tools, click the View menu, and then click Options.
- 2 In the Development Tools Options dialog box, click the List Views tab, and then set the options using the information from the following table.

Option	Description
Small, Normal, or Large	Sets the size of the font that Siebel Tools shows.
Tight, Normal, or Loose	Sets the spacing that Siebel Tools uses between rows.
Horizontal Grid Lines	Displays or hides horizontal grid lines in the list.
Vertical Grid Lines	Displays or hides vertical grid lines in the list.
Alternating Row Color	Displays a different color for every other row.
Mouse Focus Rectangle	Displays or hides a dotted line around the record that you choose.

Using the W Property

If an object is locked and editable, then Siebel Tools displays a pencil icon in the W property in the Object List Editor. The pencil icon in [Figure 2 on page 22](#) indicates that all objects in the Object List Editor are locked and editable. If you lock a parent object, then Siebel Tools locks all child objects of this parent throughout the hierarchy. If you lock a child object, then Siebel Tools locks all objects that are parents of this child throughout the hierarchy.

Using the Changed Property

If you edit a record, then Siebel Tools adds a check mark in the Changed property. This check mark indicates that you modified this object definition since a specific date and time. If the Changed property does not include a check mark, then no modification has occurred since the date and time that you specify in the General tab of the Development Options dialog box. If you modify a child object, then Siebel Tools also adds a check mark to the Changed property of all objects that are parents of this child in the hierarchy. You can control how Siebel Tools displays the check mark in the Changed property.

To use the Changed property

- 1 In Siebel Tools, click the View menu, and then click Options.
- 2 In the Development Tools Options dialog box, click the General tab.
- 3 In the Changed Date section, set the Date and Time fields, and then click OK.

If a modification occurs on a record that the Object List Editor shows, and if this modification occurs:

- **On or after the date you specify.** Siebel Tools displays a check mark in the Changed property.
- **Before the date you specify.** Siebel Tools does not display a check mark in the Changed property.

NOTE: For more information on how the change is handled in Web Tools, see [“Workspaces Dashboard in Web Tools” on page 99](#).

Using the Link

If the property value in the Object List Editor references the name of another object, then Siebel Tools displays this value as a link that you can click. If you click this link, then Siebel Tools displays the object definition of the item you clicked.

To use the link

- Make sure you are assigned the Developer responsibility in the Administration - Application screen, Responsibilities view in the Siebel client.

For more information, see *Siebel Security Guide*.

Using the Inactive Property

If the Inactive property is TRUE, then Siebel Tools and Web Tools deactivate the record in the repository the next time you deliver your modifications. For more information, on how to deliver your workspaces see [“Delivering Workspaces” on page 69](#).

To use the Inactive property

- If an object definition becomes obsolete due to an update or due to a new requirement, then you must not delete the unused objects. Instead, it is recommended that you set the Inactive property of this object definition to TRUE.

Siebel CRM does not reference an inactive object.

Defining the Name of the User Property Object Type

It is not necessary to enter the name of a valid user property if you add a user property to an object, such as an applet or business component. Instead, you can use a drop-down list in the Name field of the user property. Siebel Tools then displays a dialog box that lists the valid user properties that you can define for the object. For more information about user properties, see *Siebel Developer's Reference*.

Using Menus and Running Queries

This topic describes how to use menus and run queries.

Using the Menu Bar

The menu bar follows the title bar of the Siebel Tools interface. It includes the same menus that a typical Microsoft Windows application includes, such as File, Edit, and Help. You click a menu on the menu bar to display menu items. Siebel Tools disables the menu items that are not available. For more information, see [“Menus and Menu Items on the Menu Bar” on page 243](#).

To use the Menu Bar

- Log in to Siebel Tools, and then click a menu on the menu bar.

Using a Right-Click Menu

A *right-click menu* is a type of context-sensitive menu that appears if you right-click an object or an element. It allows you to do the following:

- Create, copy, or delete a record.
- Undo modifications that you make to a record.

- Display the name and status of a toolbar. You can right-click any toolbar. You can also customize how a toolbar appears.
- Check out or lock an object.
- Add an object to an archive.
- Access wizards.

NOTE: Right-click is available only in Siebel Tools.

Running Queries

This topic describes how to run a query in the Object List Editor. If you use the Query Menu, then Object List Editor searches for objects according to the conditions that you enter in one or more properties. You can use a simple query or a compound query. You can create, refine, or activate a query from the Query menu or from the List toolbar. You can also use shortcut keys instead of using the Query menu.

Using the Query Menu to Run a Query

This topic describes how to use the Query Menu to run a query.

To use the Query Menu to run a query

- 1 In the Object Explorer, click the object type you want to locate.
- 2 Click the Query menu, and then click New Query.

Siebel Tools displays an empty record in the Object List Editor.

NOTE: In Web Tools, search using the Filter preceding the list of objects or use the object's menu in the Object List Editor.

- 3 In the Object List Editor, enter the query.

For more information, see ["Using a Simple Query" on page 32](#) and ["Using a Compound Query" on page 33](#).

- 4 Click the Query menu, and then click Execute Query.

Siebel Tools displays the records that meet the query criteria that you entered.

- 5 (Optional) To refine the query, do the following:
 - a Click the Query menu, and then click Refine.
 - b In the Object List Editor, add more query conditions.
 - c Click the Query menu, and then click Execute Query.

Using Shortcut Keys to Run a Query

You can use a shortcut key to run a query.

To use shortcut keys to run a query

- 1 In the Object Explorer, click the object type that you want to locate.
- 2 In the Object List Editor, enter CTRL+Q in Siebel Tools or ALT+Q in Web Tools.
- 3 In the Object List Editor, enter the query.

For more information, see [“Using a Simple Query” on page 32](#) and [“Using a Compound Query” on page 33](#).

- 4 Press the ENTER key.

Siebel Tools displays the query results.

Using a Simple Query

[Table 2](#) describes the operators that you can use to create a simple query. A *simple query* is a type of query that locates records according to one condition. A check mark in a property that can contain a Boolean value represents TRUE. The Changed property is an example of a property that contains a Boolean value. For more information about query operators and Siebel data types, see *Siebel Developer’s Reference*.

Table 2. Simple Query Operators

Operator	Description
=	Equal to.
<	Less than.
>	Greater than.
<>	Not equal to.
<=	Less than or equal to.
>=	Greater than or equal to.
*	A wildcard. An asterisk (*) can represent any number of characters, including no characters.
?	A wildcard. A question mark (?) can represent any single character.
IS NOT NULL	Queries for a property that is not empty.
IS NULL	Queries for an empty property.
LIKE	Queries for a value that begins with the string that you enter.
NOT LIKE	Queries for a value that does not start with the string that you enter.
" "	Queries for a string that includes a special character. For example, enter "MyQuery's Text" to query for the MyQuery's Text string. If your query text includes a special character, then you must use quotes to enclose the query.
EXISTS ()	Queries for a value in a multi-value group.
~	Forces the case of the text string to use the case that follows the tilde (~).

Using a Compound Query

A *compound query* is a type of query that locates records according to more than one condition. You can use parentheses to control the order that Siebel Tools uses to do a compound query. Siebel Tools runs the query according to the expression that you enter. It starts with the inside parentheses and then proceeds to the next, similar to the order in which you read English.

[Table 3](#) describes operators that you can use for a compound query. You can combine simple conditions and compound conditions in a single query. For more information, see *Siebel Developer's Reference*.

Table 3. Compound Query Operators

Operator	Description
AND	All the conditions that the AND operator connects must be true.
OR	At least one of the conditions that the OR operator connects must be true.
NOT	The condition that the NOT operator precedes must be false.

To use a compound query, do one of the following:

- Enter conditions in two or more properties.
- When you run the query, Siebel Tools uses an AND operator between the conditions that you enter.
- Use OR, AND, and NOT operators in a single property.
- Use any combination of the conditions that you enter for [Step n](#) and [Step n](#).

Removing Query Results from the Object List Editor

You can remove query results from the Object List Editor.

To remove query results from the Object List Editor

- 1 Click the Query menu, and then click New Query.
- 2 Do not enter any values in the Object List Editor.
- 3 Click the Query menu, and then click Execute Query.

Getting Documentation About an Object

This topic describes how to get documentation about an object.

To get documentation about an object

- 1 Open Siebel Tools.

- 2 In the Object Explorer, click an object type, such as Business Component, and then press the F1 key.

Siebel Tools Online Help appears with documentation about the selected object type, including information about its properties. The content in this online help also appears in *Siebel Object Types Reference*.

Using Windows, Wizards, and Toolbars

This topic describes how to use windows, wizards, and toolbars. It includes the following information:

- [Using the Properties Window on page 34](#)
- [Using the Applets Window on page 35](#)
- [Using the Controls/Columns Window on page 36](#)
- [Using the Palettes Window on page 36](#)
- [Using Web Templates on page 36](#)
- [Using the Multi Value Property Window on page 37](#)
- [Using the Bookmarks Window on page 37](#)
- [Displaying, Docking, and Stacking Windows on page 37](#)
- [Using a New Object Wizard on page 39](#)
- [Using the Toolbars on page 40](#)

Using the Properties Window

The *Properties window* is a window that displays the properties and the property value for the object that you choose in the Object List Editor. It displays the name of this object and the name of the property in the one column and the property value in the next column. Siebel Tools displays properties in the Properties window in alphabetic order. You can click the Categorized tab to view these properties grouped according to category. The Properties window does not display the Project property or the Changed property.

In this example, you use the Properties window to modify the Comments property of the Siebel Call Center application.

To use the Properties window

- 1 Locate the Siebel Call Center application in the Applications list.
For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
- 2 Click the View menu, Windows, and then click Properties Window.
- 3 In the Properties window, enter the following text in the Comments property:

I can modify values in the Properties window or in the Object List Editor.

- 4 To verify your modifications, examine the Comments property in the Applications list. It includes the values you entered.

Using the Applets Window

The *Applets window* is a window that displays information about a view and allows you to add applets to that view. It includes the List tab and the Icons tab. The Icons tab performs the same function as the List tab except it displays a generic visual representation of the applet type. For more information about editing views and applets, see *Configuring Siebel Business Applications*.

The Applets window includes the same items as the Controls/Columns window except it displays the name of the business object that the view references instead of the applet. For more information, see [“Using the Controls/Columns Window” on page 36](#).

The example in this topic describes how to use the Applets window to add a form applet to the Account Address view.

To use the Applets window

- 1 In the Views list, locate the Account Address view.
For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
- 2 Right-click the view that you located in [Step 1](#), and then click Lock Object.
- 3 Right-click the view that you located in [Step 1](#), and then click Edit Web Layout.
Siebel Tools displays the Applets window and the View Layout Editor. For more information, see [“Using a Layout Editor” on page 41](#).
- 4 Click a form applet in the Applets window and, with the mouse button depressed, move it to the following placeholder in the View Layout Editor and then release the mouse button:
Child Applet
- 5 In the Pick Record dialog box, click Name in the drop-down list, enter the following query in the Starting With window, and then click Go:
Address
- 6 Click Account Service Address Map, and then click Pick.
Siebel Tools adds the Account Service Address Map applet to the View Layout Editor.
- 7 (Optional) In the View Layout Editor, right-click the applet that you added in [Step 4](#), and then click Edit Applet Web Layout.

You can now edit the Account Service Address Map applet. You can use the Controls/Columns window to add items. For more information, see [“Using the Controls/Columns Window” on page 36](#).

Using the Controls/Columns Window

The *Controls/Columns window* is a window that displays the controls and list columns that you can add to an applet layout if you use the Applet Layout Editor. Click a control or list column in the Controls/Columns window, and with the mouse button depressed, move it to a placeholder in the Applet Layout Editor and then release the mouse button.

If you choose a control or a list column in the Controls/Columns window, then the Properties window displays the properties for the object that you choose. The Controls/Columns window includes the following items:

- **Applet.** A field that displays the name of the applet.
- **Template.** A field that displays the Web template that the view references.
- **Change Template.** A button that opens the Choose Template dialog box that allows you to choose another Web template.
- **Mode.** A drop-down list that displays the view mode, such as Base or Edit.

For an example that describes usage that is similar to usage for the Controls/Columns window, see [“Using the Applets Window” on page 35](#).

Using the Palettes Window

The *Palettes window* is a window that allows you to add items to an object. For more information, see [“Editors That Include a Canvas” on page 44](#).

To use the Palettes window

- 1 Open the editor that you use to edit an object.
For example, to edit an applet layout, you open the Applet Layout Editor. When you open an editor, Siebel Tools displays the Palettes window. For an example that uses the Applet Layout Editor, see [“Using the Applets Window” on page 35](#).
- 2 Click the item with in the Palettes window and, with the mouse button depressed, move it to the editor and then release the mouse button.

Using Web Templates

Web Template is an applet in Web Tools that displays the HTML code of a Siebel Web Template in an editor and also lets you preview the object design.

To use the Web Templates applet

- 1 In Web Tools, navigate to Object Explorer and then click the Web Template object.
To filter the templates, you can use the filter preceding the list of objects or run a query using the gear icon in the Object List Editor applet.

- 2 Drill down on the Web template that you want to view or modify.

Web Tools displays the HTML code for Siebel Web Template in Object List Editor. To modify, you must first open a workspace. To do so, see [“To access and open the Workspace Dashboard view in the Web Tools mode” on page 104.](#)

- 3 If the Web template is editable, you can edit the HTML code in the editor.

You can customize the editor with options such as auto-close tags, hints, display of line numbers or Object Design hierarchy, and indentation. The settings you select are automatically available whenever you access the editor.

- 4 Click the Preview (eye-shaped) icon to preview the object design.

You can toggle between the editor and preview.

- 5 Click the Back button to exit the preview.

NOTE: To learn more about layout editors, see [“Using a Layout Editor” on page 41.](#)

Using the Multi Value Property Window

The Multi Value Property Window allows you to view and set values for multi-value properties when you use one of the following designers:

- Entity Relationship Designer
- Task Designer
- Workflow Process Designer

For more information about multi-value properties and how to use the Multi Value Property Window, see *Siebel Business Process Framework: Task UI Guide* and *Siebel Business Process Framework: Workflow Guide*.

Using the Bookmarks Window

The *Bookmarks window* is a window that allows you to navigate directly to an object that you use frequently. For more information, see [“History Toolbar” on page 259.](#)

Displaying, Docking, and Stacking Windows

This topic describes how to display, dock, and stack windows.

Displaying a Window

You can display or hide a window.

To display a window

- In Siebel Tools, click the View menu, click Windows, and then click the window that you want to display.

You can also click the Go menu, and then click Bookmarks List to display the Bookmarks window.

- To hide a window, do one of the following:
 - Click the Close button in the title bar of the window.
 - Right-click the window that you want to hide, and then click Hide.

Docking a Window

You can dock a window in a corner of the main window.

To dock a window

- Relocate the window to the area of the main window where you want to dock the window.
- To undock a window, you can right-click the window, and then click Docked.
- To prevent Siebel Tools from docking a window when you move it, you can hold down the CTRL key while you move the window.

Displaying a Window as a Tab

You can display a docked window, including the Object Explorer, as a tab. You can open, close, or dock a tabbed window.

To display a window as a tab

- Right-click the window, and then click Hide.

Siebel Tools collapses the window and displays a tab for it at the margin of Siebel Tools.

- To display a tabbed window:
 - **Temporarily.** Place your cursor over the tab that represents the window.
 - **Permanently.** Place your cursor over the tab that represents the window, and then click the push pin in the window title bar.

Stacking Windows

You can stack windows over each other if they are floating.

To stack windows

- Relocate a floating window to another floating window.

Siebel Tools creates a single window that includes both windows, and it adds a tab that represents each window. You can click the tabs to navigate between these stacked windows.

Resetting Windows

You can reset windows to the display that Siebel Tools uses immediately after you first install it.

To reset windows

- In Siebel Tools, click the View menu, and then click Reset Windows.

Siebel Tools closes all windows except the Object Editor and the Object List Editor. It pins the Object Editor and the Object List Editor to their placeholders and resizes them to their original size.

Using a New Object Wizard

A *new object wizard* is a feature that you can use that guides you through the steps of creating a new object. It prompts you to configure the properties and objects that Siebel Tools requires according to the object type that you are creating. You can use a new object wizard to create the following objects:

- **General object.** For example, a business component, table, view, or applet method menu item.
- **Applet object.** For example, a list applet, form applet, MVG Applet, or chart applet.
- **EAI object.** For example, an integration object.
- **Task object.** For example, a task, task applet, task view, or transient business component.

For more information about how to use a wizard for a particular object type, see *Configuring Siebel Business Applications*.

NOTE: You can use the Edit Web Layout function, which is available only in Web Tools, to edit the web layout.

To use a new object wizard

- 1 Do one of the following:
 - Click the File menu, and then click New Object.
Siebel Tools displays the New Object Wizards dialog box.
 - Right-click an object in the Object List Editor, and then click New Object Wizards.
Siebel Tools displays a list of wizards that are specific to the object type that you choose.
- 2 Choose a wizard.

Using the Toolbars

Figure 3 illustrates the toolbars that you can use in Siebel Tools. A toolbar is active only if the object type or window that uses it is active. You can relocate toolbars using the mouse. You can display and hide toolbars.

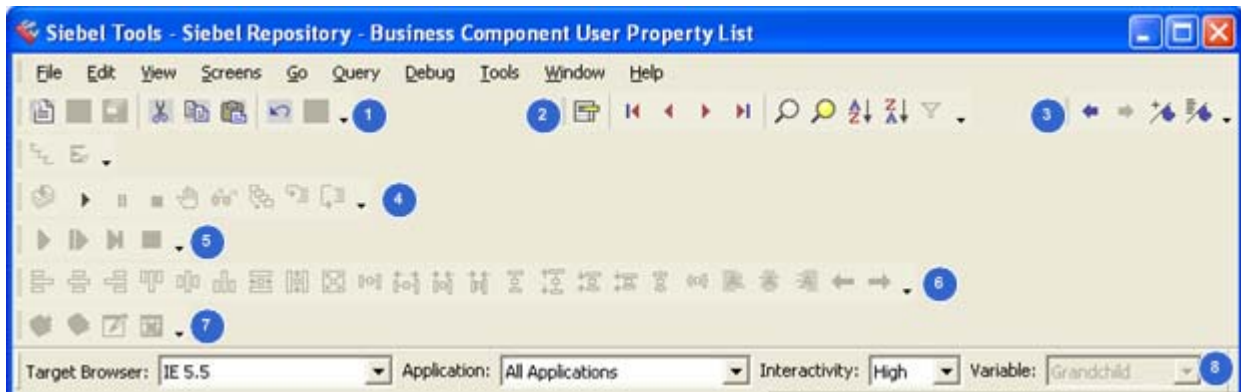


Figure 3. Toolbars You Can Use in Siebel Tools

Explanation of Callouts

Siebel Tools includes the following toolbars:

- 1 Edit Toolbar.** Allows you to use edit tools, the New Object wizard, and undo and redo options. You can right-click a field in the Object List Editor to display a menu of edit tools. For more information, see ["Using a Layout Editor" on page 41](#).
- 2 List Toolbar.** Allows you to insert a new record, move forward and backward through records, query records, and sort records. It manages records that the Object List Editor shows.
- 3 History Toolbar.** Allows you to retrace your steps, and to create and navigate to bookmarks.
- 4 Debug Toolbar.** Allows you to access debug tools that you can use with Siebel VB and Siebel eScript. For more information, see [Chapter 7, "Using Siebel Script Editors."](#)
- 5 Simulate Toolbar.** Allows you to simulate a workflow process.
- 6 Format Toolbar.** Allows you to apply a format to an applet control. This control uses a Web template that uses a grid layout.
- 7 WF/Task Editor Toolbar.** Allows you to publish, revise, or expire a workflow process or task UI. You can activate a workflow process if you are the owner of the root workspace and the workflow is delivered into it.
- 8 Configuration Context Toolbar.** Allows you to define settings for Web browser layout and scripting.

Displaying a Toolbar

You can display a toolbar.

To display a toolbar

- In Siebel Tools, click the View menu, click Toolbars, and then click the menu item that represents the toolbar that you want to display.

If Siebel Tools displays a check mark next to the menu item that represents the toolbar, then the toolbar is visible.

Displaying the Status Bar

You can display the status bar.

To display the status bar

- In Siebel Tools, click the View menu, and then click Status Bar.

If Siebel Tools displays a check mark next to the Status Bar menu item, then the Status Bar is visible.

Using Editors in Siebel Tools

This topic describes how to use some of the editors that you can use in Siebel Tools. It includes the following information:

- [Using a Layout Editor on page 41](#)
- [Using an Editor That Includes a Canvas on page 42](#)
- [Using a Script Editor on page 44](#)

Using a Layout Editor

Web Tools supports the following layout editors for both edit and preview of the code:

- Applet Layout Editor
- View Layout Editor
- Web Page Layout Editor

Siebel Tools supports only Applet Menu Layout Editor.

A layout editor allows you to do the following:

- Add or map a control or list column to an applet layout.

- Modify an existing view or create a new view. Click an applet and, with the mouse button depressed, move the applet to the View Layout Editor and then release the mouse button. You can view a list applet, a form applet, or the container page in Preview mode.

NOTE: To map view, applet, or control in Web Tools, first identify the unmapped placeholders and their Item Identifiers from the preview in Web Template of View or Applet objects. Next, click the required Web Template Item of that object and add field values in the child applet. When you preview the Web Template again, the field will be in bold to indicate that it is now mapped.

- Add or delete a control from a Web page template, modify a control property, or map a control to a placeholder.
- Edit the menu structure of a Siebel application. You right-click an applet in the Object List Editor, and then click Edit Web Menus.
- Preview an approximation of how Siebel CRM displays an applet or a Web page at run time.

For more information about using layout editors, see *Configuring Siebel Business Applications*.

NOTE: You should use Web Tools to preview applet, view, and web page.

To use a layout editor

- 1 In the Object List Editor, locate the object that you want to modify.

You can open a layout editor from an applet, view, or Web page in Web Tools. For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).

- 2 Click the object that you located in [Step 1](#), and then click the Preview (eye-shaped) icon to preview the design.
- 3 To exit the preview, click the Back button.

NOTE: For more information on the features of an editor and how to use it, see [“To use the Web Templates applet” on page 36](#).

Using an Editor That Includes a Canvas

A *canvas* is a background that appears in different designers. It includes a work area that allows you to move or rearrange objects. For example, the Task Designer allows you to move a Siebel Operation step from the Palette window to the canvas. The canvas also indicates whether or not you can edit an object.

The example in this topic uses the canvas that appears in the Task Designer for the FS Asset To Contract Task.

To use an editor that includes a canvas

- 1 In the Object Explorer, click Task.

- 2 In the Tasks list, locate the FS Asset To Contract Task.
Siebel Tools displays two records for the FS Asset To Contract Task. For more information, see [“How Siebel Tools Displays Object Types and Object Definitions” on page 21](#).
- 3 Use multiselect to choose both records in the Tasks list.
For more information, see [“Choosing More Than One Record in the Object List Editor” on page 43](#).
- 4 Right-click the tasks that you chose in [Step 3](#), and then click Lock Object.
- 5 Use the canvas for an object that you can edit:
 - a Right-click the task that includes In Progress in the Status property, and then click Edit Task Flow.
Siebel Tools displays the canvas as a solid grid, which indicates that you can edit the task. It displays task objects over this grid.
 - b To verify that you can edit the task, do the following:
 - Move a Siebel Operation step from the Palette window to the canvas. For more information, see [“Using the Palettes Window” on page 36](#).
 - Move an existing object.
 - c Click the File menu, and then click Close.
 - d In the Confirm dialog box, click No.
- 6 Use the canvas for an object that you cannot edit:
 - a Right-click the task that includes Completed in the Status property, and then click Edit Task Flow.
Siebel Tools displays the canvas as a solid background with no grid, which indicates that you cannot edit the task. Some editors, such as the Workflow Process Designer, use a colored canvas to indicate that you cannot edit the object.
 - b To verify that you cannot edit the task, do the following:
 - Attempt to move a Siebel Operation step from the Palette window to the canvas.
 - Attempt to move an existing object.
 - c Click the File menu, and then click Close.
- 7 Right-click the tasks that you chose in [Step 3](#), and then click Unlock Object.

Choosing More Than One Record in the Object List Editor

You can use multiselect to choose more than one record in the Object List Editor.

To choose more than one record in the Object List Editor

- Do one of the following:
 - To choose records that are not consecutive, hold down the CTRL key while you click each record.
 - To choose multiple records that are consecutive, click a record, hold down the SHIFT key, and then click another record.

Editors That Include a Canvas

Table 4 describes the editors that include a canvas. You can use these editors with the Palettes Window

Table 4. Editors That Include a Canvas

Editor	Description
Applet Layout Editor	Allows you to add interface controls to an applet layout. Button, Label, and CheckBox are examples of controls that the Palettes window shows. For more information, see “Buttons in the Palettes Window” on page 264.
Entity Relationship Designer	Allows you to diagram the business entities that your configuration uses and represent relationships between these entities. For more information, see <i>Configuring Siebel Business Applications</i> .
Task Designer	Allows you to create a user interface that assists the user in completing a job task. Allows you to add steps and connectors to a task UI. For more information about task UI and using the Task Designer, see <i>Siebel Business Process Framework: Task UI Guide</i> .
Workflow Process Designer	Allows you to define, manage, and enforce the business processes that your company uses. Allows you to add steps and connectors to a workflow process. For more information about workflow processes, using the Workflow Designer, and using the Workflow Simulator, see <i>Siebel Business Process Framework: Workflow Guide</i> .

Using a Script Editor

If you cannot use declarative configuration to meet your requirements, then you can use the Server Script Editor and the Browser Script Editor in Siebel Tools to create a script. You use these editors to add a script to a Siebel object. You can use the following types of script:

- Siebel VB
- Siebel eScript
- Browser Script

For information about declarative configuration, see [“Overview of Using Siebel Tools” on page 15](#). For information about scripting, see [Chapter 7, “Using Siebel Script Editors.”](#)

Using the Command Line

For more information about using the command line, see the following information:

- [Using the Command Line to Validate Objects on page 149](#)
- [Using the Command Line to Run the Locale Management Utility on page 216](#)
- [Using the Command Line to Export Objects to an Archive on page 223](#)
- [Using the Command Line to Import an Archive on page 227](#)

3

Using Web Tools

This chapter describes how to use Oracle's Siebel Tools. It includes the following topics:

- [About Web Tools on page 47](#)
- [Benefits of Web Tools on page 48](#)
- [About Delivering in Siebel Tools and Web Tools on page 48](#)
- [Siebel Tools Features Included in Web Tools on page 48](#)
- [Siebel Tools Objects Included in Web Tools on page 49](#)
- [Siebel Tools Features Excluded From Web Tools on page 50](#)
- [Siebel Tools Objects Excluded From Web Tools on page 50](#)

About Web Tools

Oracle is gradually migrating from Siebel Tools to Web Tools to simplify and expedite the process of configuring Siebel Business Applications. Web Tools is a Web-enabled application that runs on a server, and you access it in a browser by using a Siebel Open UI client. Siebel Tools is a stand-alone application that runs on Microsoft Windows.

In Web Tools, you currently have access to some of the functionality in Siebel Tools, so you can use Web Tools to perform some of the tasks that in the past you performed in Siebel Tools. In both Siebel Tools and Web Tools, you need to be added to the database to access these tools. In addition, for Web Tools, you need to have the Composer Administrator responsibility to access the Workspace Dashboard to perform developer or configuration operations. To learn more about workspaces in Web Tools, see ["Setting up Web Tools" on page 97](#).

Using Web Tools, note that:

- You must ensure that the Workspace features are enabled on the environment before you work on Web Tools.

For more information about how to use the Workspace feature in Siebel Tools, see ["Using Workspaces in Siebel Tools" on page 56](#).

- Web Tools has the same client requirements as other Siebel Open UI clients. For more information, see *Deploying Siebel Open UI*.

NOTE: In Innovation Pack 2017, Web Tools must be run using the ENU language parameter. Full translation for Web Tools is planned for future innovation pack releases. The *Siebel Bookshelf* is available on Oracle Technology Network (<http://www.oracle.com/technetwork/documentation/siebel-087898.html>) and Oracle Software Delivery Cloud. It might also be installed locally on your intranet or on a network location.

Benefits of Web Tools

Web Tools provides the following benefits:

- More elements for composition of the user interface.
- More efficient rendering of Web pages.

The DOM (Document Object Model) structure in Siebel Business Applications is simplified, and the DOM is assembled using a single HTTP response, not multiple relays between the client and the server.

When you use Siebel Tools, content is trimmed only from the client. Also, post-response JavaScript execution trims content. When you use Web Tools, the server evaluates client capabilities, such as resolution and form factors. Content is tailored from the server, and more efficient options are available to manage content.

- Decreased network traffic.

Only the content that is necessary to currently render the user interface is included in responses. Extraneous content that the client must later discard or ignore is not included in responses. This decrease in content decreases the network traffic.

- More control in the client.
- Improved file loading.

The framework files that are needed to start a Siebel application are loaded from the manifest. The files for the presentation model and physical renderer that are needed to render Web pages are also loaded from the manifest.

About Delivering in Siebel Tools and Web Tools

In Siebel Tools, you deliver modifications to implement your configuration changes in the user interface of the Siebel application. You deliver the modifications that you make to objects directly into the Siebel application. To deliver modifications, see [“Delivering Workspaces” on page 69](#).

In Web Tools, you deliver modifications to implement your configuration changes in the user interface of the Siebel application. You deliver the modifications that you make to objects directly into the Siebel application.

Siebel Tools Features Included in Web Tools

Web Tools currently supports the following features in Siebel Tools:

- Locking and unlocking projects in the local repository.
- Configuring symbolic strings.
- Adding to archive.

- Sorting on objects.
- List of Values, Analytic Strings, and System Preferences.
- Settings for Web Tools.
- Using Object Explorer and Object List Editor.
- Using Layout Editor to modify templates.
- IO deploy/undeploy.

Siebel Tools Objects Included in Web Tools

Table 5 shows the objects in Siebel Tools that Web Tools currently supports.

CAUTION: Operations are not allowed on the read-only objects: Dock Object, the Schema Maintenance objects, Task, Type, Repository, Table, and EIM Interface Table.

Table 5. Objects in Siebel Tools That Web Tools Supports

Applet	Find	Screen	Workflow Process
Application	HTML Hierarchy Bitmap	Symbolic String	
Business Component	Icon Map	System Activity Object	
Assignment Attribute	Import Object	Table	
Assignment Criteria	Integration Object	Task	
Bitmap Category	Link	Task Group	
Business Object	Menu	Toolbar	
Business Service	Message Category	Type	
Class	Pick List	View	
Command	Project	Web Page	
Content Object	Repository	Web Template	
DLL	Schema Maintenance Phase	Workflow Policy Column	
Dock Object	Schema Maintenance Process	Workflow Policy Object	
EIM Interface Table	Schema Maintenance Step	Workflow Policy Program	

Siebel Tools Features Excluded From Web Tools

Web Tools currently does not support the following features in Siebel Tools:

- Using various windows and wizards (such as New Object and Synchronize) that are only available in Siebel Tools.
- Tagging objects to manage developer changes.
- Validating, examining, and comparing objects.
- Using the Siebel Script Editor, the ST eScript Engine, the Siebel Debugger, and the Script Profiler.
- Configuring locale data and using the Locale Management utility to manage locales and Advanced Compile to localize repositories.
- Setting debug options.
- Importing objects from an archive and using Application Deployment Manager (ADM).

Siebel Tools Objects Excluded From Web Tools

This is the list of the objects in Siebel Tools that Web Tools currently does not support:

- Entity Relationship Diagram
- Help Id
- Pager Object
- Server Component Type
- Search Category
- Search Engine
- Search Index

4

Using Workspaces in Siebel Tools

This chapter describes the Workspaces feature and how to use workspaces in Oracle's Siebel Tools. It includes the following topics:

- ["Overview of the Workspaces Mode for Siebel Tools" on page 51](#)
- ["Using Workspaces in Siebel Tools" on page 56](#)
- ["Using Workflows in Workspaces" on page 71](#)
- ["Workspaces Administration" on page 77](#)

Overview of the Workspaces Mode for Siebel Tools

This section describes the Workspaces feature and the Workspaces dashboard. It includes the following topics:

- ["Workspaces Overview" on page 51.](#)
- ["Workspace Users" on page 52](#)
- ["Workspace Dashboard in Siebel Tools" on page 53.](#)

Workspaces Overview

The Workspace feature provides users a new way to manage configurations of repository artifacts in Siebel Tools. This feature allows multiple developers to work on the same repository objects in the Siebel database.

A workspace provides a user with a sandbox for editing and publishing (delivering) configuration changes until these changes are ready to be delivered into the main workspace (parent, root, or master workspace). This feature ensures isolation from other users making changes to either the same objects or other objects in the application. It is also an alternative to using local databases to make changes to the repository data.

Users can preview and test repository updates made in a developer's environment (workspace) in Web Tools without affecting other developers. They can do this when they are added to the database and also have the Composer Administrator responsibility to access the Workspace Dashboard to perform developer or configuration operations. Until all updates are delivered, these updates will not impact other developers. Once the changes in developer workspace are delivered then Siebel runtime repository tables will be updated with the new values. To learn more about Web Tools and workspaces in Web Tools, see ["Using Web Tools" on page 47](#) and ["Setting up Web Tools" on page 97](#).

In other words, the Workspace feature in Siebel Tools provides the following capabilities:

- A user can make configuration changes to the repository application without any impact on other users.
- Configuration changes are persistent but in isolation from the metadata of the deployed application.
- Only modifications to the configuration (delta) are tracked and stored, while other configurations are referenced from the base repository in order to render a consistent view of the modified application.
- Persistence of configuration changes enables configuration to be a long-running operation or transaction.
- Concurrency is achieved so that multiple users can each modify the same application in isolation from one another.

Workspace is also a feature of Web Tools, which is an alternative to the traditional Siebel Tools client. Workspace administrators can enable the Workspace feature for Siebel Tools at the database level by running the executable utility. After the Workspaces mode is enabled for Siebel Tools, it cannot be reverted back to the non-workspace mode. For more information about how to enable Workspace for Siebel Tools, see [“Enabling the Workspaces Mode for Siebel Tools” on page 78](#).

NOTE: Starting from Siebel Tools Innovation Pack 2017, all Siebel Tools developers must use workspaces because there is no local database.

For more information about how to use workspaces for Web Tools, see [“Setting up Web Tools” on page 97](#), and [“Navigating to the Workspace Dashboard in the Web Tools Mode Application” on page 103](#).

Workspace Users

Workspace users are categorized into these two groups:

- Workspace Owner is the owner of a workspace and also the creator of that workspace.
- Workspace Administrator is the owner of the MAIN workspace who has the privileges to deliver workspaces to MAIN, undo submit for delivery, flatten, etc. The owner of the MAIN workspace is set at the time of enabling Siebel Workspace on the database using the EnableWorkspace utility.

NOTE: Only the Siebel (system) Administrator can create and manage the responsibility called *Workspace Administrator*.

For more information on how to enable Siebel Workspace on the database, see [“Enabling the Workspaces Mode for Siebel Tools” on page 78](#).

To create the responsibility Workspace Administrator in Siebel application

- 1 Ensure that Workspaces and Parallel Development Using Workspaces features are enabled in the system.
- 2 Navigate to *Administration - Application, Responsibilities*.
- 3 Click the New Record button in the *Responsibilities* applet.
- 4 Add a new responsibility called *Workspace Administrator*.

- 5 In the *Users* applet, click the New Record button to add the user IDs that will have the authority to perform the administrative tasks.
- 6 Save the record.

Workspace Dashboard in Siebel Tools

The Workspace dashboard enables users to view the list of all workspaces that were created and currently presented in the database. From the workspace dashboard, you can perform various operations on workspaces.

Figure 4 illustrates the workspace dashboard in Siebel Tools.

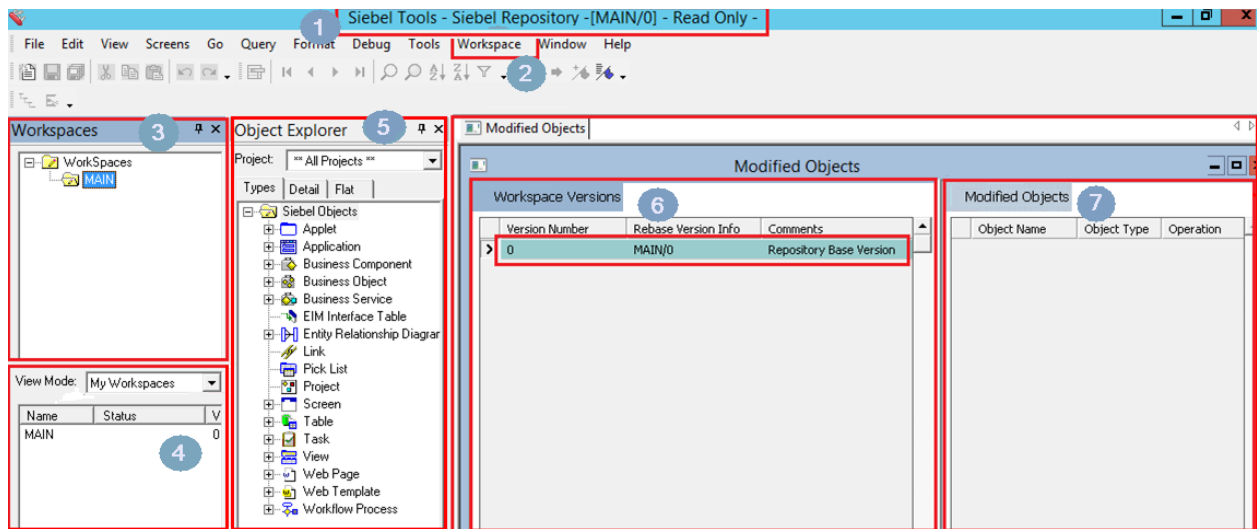


Figure 4. Example of Siebel Tools in Workspace mode displaying the workspace dashboard

Explanation of Callouts

Figure 4 includes the following information:

- 1 The Siebel Tools title bar displays the title that includes the name, the latest version, and the status of the opened workspace. For example:

Siebel Tools - Siebel Repository - [MAIN/0] - Read Only

NOTE: You have the *Editable* access on your workspace after you create it, but you have *Read only* access after you change its status to *Submit for Delivery*.

- 2 On the menu bar, the Workspace menu is between the Tools and the Window menus.

When you select the Workspace menu, a list of different options that are used to configure and manage workspaces appears. This table lists and describes the options under the Workspace menu:

Option	Description
Create	Select this option to create a new workspace. For more information, see "Creating New Workspaces" on page 56.
Checkpoint	Select this option to check in the changes that you made to the current version of the workspace. For more information, see "Performing the Checkpoint Version Process" on page 61.
Revert	Select this option to revert the changes that you made to the current version of that workspace. For more information, see "Reverting to Previous Workspace Versions" on page 62.
Rebase	Select this option to apply the changes that were made in the parent workspace into the current workspace. For more information, see "Rebasing Workspaces" on page 62.
Merge Reports	Select this option to view and resolve the conflicts that occurred during the rebase process. For more information, see "Rebasing Workspaces" on page 62 and "Detecting Conflicts in Workspaces and Applying Resolutions" on page 64.
Submit for Delivery	Select this option to change the status of the workspace and make it ready for delivering the changes to the MAIN workspace. For more information, see "Submitting Workspaces for Delivery" on page 67.
Deliver	Select this option to deliver the changes in the workspace to the MAIN workspace. NOTE: This option is available only to the user who is also the owner of the MAIN workspace. For more information, see "Delivering Workspaces" on page 69.
Compare	Select this option to view the differences that are made to the objects in two selected workspace versions. For more information, see "Comparing Workspace Versions" on page 70.

Option	Description
Undo Submit for Delivery	Select this option to cancel the workspace delivery process. For more information, see “Canceling the Workspace Delivery Process” on page 82.
Workspace Explorer	Select this option to display the Workspace explorer pane.

- 3 The Workspace explorer pane is displayed next to the window.

After you select the Workspace Explorer option under the Workspace menu, this pane appears listing all available workspaces in the database. This section is expandable and collapsible. You can select a workspace and then view the versions of the selected workspace and the list of modified objects for the selected version.

By default, Siebel Tools opens the latest version of the root workspace and this workspace is set at the session. Also, the root workspace is the only workspace in the Workspace explorer pane and it is always a read-only workspace.

Depending on the workspace that you select, different context menu options are available in the Workspace dashboard.

- 4 The View Mode pane is also displayed next to the window following the Workspace explorer pane. This pane lists the workspace information (including the names, current statuses, and versions) based on the selected filter option in the View Mode drop-down list.

This table describes the filter options under the View Mode drop-down list:

Option	Description
My Workspaces	Select this option to view all workspaces that you own regardless of their status.
My In-progress Workspaces	Select this option to view the workspaces that you own and that have the status of <i>Edit In Progress</i> .
All Workspaces	Select this option to view all workspaces that were created by all users of the current development environment.

- 5 In the Object Explorer pane, most of the object types are workspace-enabled, except the following ones: Table, Dock Object, EIM Interface Table, Repository, Workflow Policy Column, Workflow Policy Object, Workflow Policy Program, Task, Task Group, and Project.
- 6 The Workspace Version pane lists the versions of the selected workspace and the rebase version information. For more information, see [“Rebasing Workspaces” on page 62.](#)
- 7 The Modified Objects pane lists the objects that were modified in the current version of the workspace. This pane displays the object name, object type, and operation type performed on the modified objects. For more information, see [“Tracking Repository Object Changes in Workspaces” on page 60.](#)

Using Workspaces in Siebel Tools

This topic describes how to use the Workspaces feature in Siebel Tools. These tasks are performed by all users who have the permissions to use the Workspace feature. This topic includes the following information:

- [“Creating New Workspaces” on page 56](#)
- [“Opening Existing Workspaces” on page 57](#)
- [“Editing Workspace-Enabled Repository Objects” on page 58](#)
- [“Deleting Development Workspaces” on page 59](#)
- [“Refreshing the Workspace Explorer Pane” on page 60](#)
- [“Tracking Repository Object Changes in Workspaces” on page 60](#)
- [“Performing the Checkpoint Version Process” on page 61](#)
- [“Reverting to Previous Workspace Versions” on page 62](#)
- [“Rebasing Workspaces” on page 62](#)
- [“Detecting Conflicts in Workspaces and Applying Resolutions” on page 64](#)
- [“Submitting Workspaces for Delivery” on page 67](#)
- [“Canceling the Workspace Delivery Process” on page 68](#)
- [“Delivering Workspaces” on page 69](#)
- [“Exporting Workspace Objects to Archive Files” on page 69](#)
- [“Comparing Workspace Versions” on page 70](#)
- [“Flattening Workspace Versions” on page 70](#)
- [“Configuring Non-Workspace Objects” on page 71](#)

Creating New Workspaces

This topic describes how to create new workspaces.

To create a new workspace

- 1 In Siebel Tools, select the Workspace menu and then select the Create option.
The Create Workspace dialog box appears.

- 2 Enter the workspace name in the Enter Workspace Name field.

Workspace names can only contain lower case alphabetic, numeric, hyphen, and underscore characters. Also, workspace names must start with the value that is set for Workspace Prefix in the System Preferences, appended by `_<login userid>_`.

For example, if you log into Siebel Tools using user name *sadmin*, the workspace name must be *dev_sadmin* where *dev* is the value set for the Workspace Prefix system preference. For more information on how to add Workspace Prefix in system preferences, see [“Adding the Workspace Prefix in System Preferences” on page 94](#).

- 3 Note the values in the Parent Workspace and Parent Workspace Version fields.

By default, the newly recreated workspace is always branched out from the latest version of the MAIN workspace; therefore, the Parent Workspace field is populated with the value *MAIN* and the Parent Workspace Version field is populated with the latest check-pointed version of the MAIN workspace.

- 4 (Optional) Use the Parent Workspace Version drop-down list to select a previous version to branch out from the previous versions of the MAIN workspace.

- 5 Enter the description for the workspace in the Add Description field.

- 6 Click the OK button and note that:

- In the Workspaces pane, the newly created workspace is listed under the MAIN workspace.
- In the View Mode list, the status of the workspace is updated to *Created*.
- The title bar text in Siebel Tools is updated using this format:

Siebel Tools - Siebel Repository -[workspace name>/0] - Editable -

For example:

Siebel Tools - Siebel Repository -[dev_sadmin_test/0] - Editable -

Opening Existing Workspaces

You must open a workspace before you can make any configuration changes. Remember that you can open only one workspace at any given time within your current session.

- If you are the owner of the opened workspace, then you can edit (read and write) that workspace.
- If you are not the owner of the opened workspace, then you can only read (without editing) that workspace.

Opening a workspace sets a session-level workspace context; therefore, while the workspace is opened, all configuration changes that you made are saved as parts of the opened workspace.

To open an existing workspace

- 1 In the workspace dashboard, navigate to the Workspace explorer pane.

- 2 Select the workspace that you want to open and then right-click.

A list of available options appears. These options are Open, Delete, Refresh, Docked, and Hide.

TIP: Select the Hide option to hide the Workspace explorer pane or select the Close (X) icon to close this pane.

- 3 Select the Open option.

The workspace dashboard opens the selected workspace and displays its information.

Editing Workspace-Enabled Repository Objects

You can edit workspace-enabled repository objects directly without locking any project. Note that only workspace-enabled repository objects can be edited inside workspaces. By default, all workspace-enabled objects are locked at the database level, so you are not required to lock projects when you edit the objects or perform any type of CRUD (create, read, update, delete) operations.

However, locking a project is still required before you edit the non-workspace objects. The non-workspace objects are Table, Task, Workflows, Repository, Type, EIM Table, Projects, Dock objects, Schema Maintenance, and Server Components.

As in the previous releases of Siebel Tools:

- You can update existing records by selecting the attribute and child record.
- You can create new child records of the parent objects by drilling down on the child objects using the tree hierarchy in the Object Explorer pane.

All context menu options that are applicable in the previous releases of Siebel Tools for the child level objects are still applicable when the Workspace mode is enabled.

- You cannot delete any parent objects or child-level objects if any of them are workspace-enabled objects.

Only the new parent objects that were created in one workspace version can be deleted in that version.

To edit a workspace-enabled repository object

- 1 In the workspace dashboard, navigate to the Workspace explorer pane.
- 2 Right-click on the workspace that you want to edit and then select the Open option.
- 3 Note that:
 - In the Siebel Tools title bar text, the status *Editable* is added at the end of the title.
 - In the View Mode pane, the status of the workspace is shown as *Edit-In-Progress*.
- 4 Edit the repository object as needed.

NOTE: In workspace mode, the concept of checking out objects does not apply to workspace-enabled repository objects, explicitly. As in the previous releases of Siebel Tools, you can perform all context menu options such as inserting new records, copying records, and other default options that are specific to the selected objects.

5 (Optional) Create new object records using the following steps:

- a In the workspace dashboard, select the File menu and then select the New Object option. Alternatively, click the New icon in the tool bar.
- b In the New Object Wizards dialog box, select an object template.

TIP: When Workspace mode is enabled, project or object locking is not mandatory. You can select an appropriate object by selecting one option in the New Object Wizards dialog box.

6 Save the changes.

If the workspace is in *Edit-In-Progress* state, then the changes are tracked under the latest checked-out version. Otherwise, a new version is created and changes are tracked under that new version until that version is checkpointed.

If you edit the repository object again after it is checkpointed, the current version is incremented by 1 and all new changes are tracked under the current version until the new version is checkpointed.

For more information on how to enable workspaces in Siebel Tools and then edit the repository objects, see [“Enabling Workspaces in Siebel Tools and Editing the Repository Objects” on page 81](#).

For more information on how to configure non-workspace objects, see [“Configuring Non-Workspace Objects” on page 71](#).

Deleting Development Workspaces

You can delete the development workspaces that you own, but you cannot delete workspaces or parent workspaces that are owned by other users. In addition, development workspaces that are in the status of *Submitted for Delivery* or *Delivered* cannot be deleted from the repository.

NOTE: Only an administrator can delete the integration workspaces if these integration workspaces do not have any children workspaces. In other word, an administrator cannot delete the integration workspaces if any child development workspaces exist and have some changes.

To delete a workspace

- 1 In the workspace dashboard, navigate to the Workspace explorer pane.
- 2 Select the workspace that you want to delete and then right-click.
A list of available actions appears. These options are Open, Delete, Refresh, Docked, and Hide.
- 3 Select the Delete option.
The selected workspace is removed from the Workspace explorer pane.

NOTE: You can delete a workspace in Web Tools as well.

Refreshing the Workspace Explorer Pane

At the time you open the Workspace explorer pane, this pane lists all available workspaces that are in the database. After that, if other users use other workspace sessions to modify their workspaces in the same database, such as adding new workspaces or deleting the current ones, you need to refresh the Workspace explorer pane to display these changes.

To refresh the Workspace explorer pane

- 1 In the workspace dashboard, navigate to the Workspace explorer pane.
- 2 Select the root (MAIN) workspace and then right-click.
A list of available options appears. These options are Open, Delete, Refresh, Docked, and Hide.
- 3 Select the Refresh option.
The Workspace explorer pane is refreshed displaying the latest changes for all workspaces in the database.

NOTE: You can hide or close the Workspace Explorer pane by selecting the Hide or the Close (X) option from the action list.

Tracking Repository Object Changes in Workspaces

You can use the Modified Objects option under the Workspace menu to view the changes that you have made to objects in the current workspace.

To track repository object changes in a workspace

- 1 In Siebel Tools, navigate to the Workspace explorer pane.
- 2 To see the list of modified objects in a workspace without opening a workspace, select that workspace and then select a version from the Workspace Version pane.
- 3 To see only the changes in the repository, open a version of the workspace.

When you open a particular version, all versions succeeding the selected version are also opened. For example, if the workspace *dev_sadmin_demo* has 10 versions and you open the fifth version, then you can view the repository changes of version one through version five.

- 4 View the detailed changes in the Modified Objects pane.

The Modified Objects pane lists the object names, object types, and operations that were performed for each modified object.

This table describes the values listed in the Operation column on the Modified Objects pane.

Option	Description
Update	Indicates an operation of updating a parent object record. Also indicates an operation of inserting, updating, or enabling the Inactive option for a child record.
Insert	Indicates an operation of inserting a new parent object record.
Delete	Indicates an operation of enabling the Inactive option for a parent object record.

NOTE: You cannot delete any existing record in workspace-enabled objects, but the new objects that were created in the latest workspace version can be deleted in that version.

Performing the Checkpoint Version Process

The checkpoint operation commits the changes that you made to the current workspace version of the selected workspace and sets that version to be non-editable. After checkpointing the workspace, you are no longer able to make object changes to the current version of the workspace. All subsequent object changes will be tracked under the next version of the workspace.

To perform a checkpoint workspace version process

- 1 In Siebel Tools, select the Workspaces menu and then select the Checkpoint option.
The Enter Comment dialog box appears.
- 2 Enter the comments, which will be displayed in the Comment column on the Workspace Versions pane.
- 3 Click the OK button.
A message appears saying the checkpoint process is completed successfully.
- 4 Click the OK button.
After a workspace version is check pointed, its status changes to *Checkpointed* in the My Workspaces pane and the Submit for Delivery option for that workspace is enabled under the Workspaces menu.

Reverting to Previous Workspace Versions

When the changes that were made since the last checkpointed version of the workspace cause issues, you can revert or roll back to the most recent checkpointed version. You can also revert or roll back a checkpointed version to the previous checkpointed version.

- If you perform the revert process on a workspace with the status set to *Edit-In-Progress* or *Checkpointed*, then all changes that you made in the latest version are lost and the workspace reverts back to the last checkpointed version.
- If you perform the revert process on the first version of the workspace, then all changes that you made in the first version are lost and the status of the workspace reverts back to *Created*.

To revert to a previous version of a workspace

- 1 In Siebel Tools, select the Workspace menu and then select the Revert option.
A revert confirmation appears.
- 2 Click the Yes button to revert all changes back to the last checkpointed version.
NOTE: All changes that were made in the latest version will be lost.
Alternatively, click the No button to cancel the revert operation.

Rebasing Workspaces

Rebasing (merging) a workspace is applying the changes that were made in the parent workspace (as a result of deliveries by other workspaces that have branched off the same parent) into the current workspace.

Only the owner of the workspace can perform the rebase process for that workspace.

You can perform the rebase process only after you perform the checkpoint operation at least on the first version of the workspace. If you perform the rebase process for the workspace that has no changes in the root (MAIN) workspace since it was created, then the rebase process fails and this error message appears: *The rebase failed with error: No changes found to rebase/deliver/checkpoint.*

To rebase a workspace

- 1 In the workspace dashboard, select the Workspace menu and then select the Rebase option.
The Rebase Workspace dialog box appears displaying the From Workspace, To Workspace, and Merge Status fields and the Start Rebase button.
- 2 Click the Start Rebase button to start the rebase process.
The Rebase Workspace dialog box displays the merge details and the rebase process bar in the Merge Status section.

- 3 Confirm that the rebase process bar shows the process is completed.

The Rebase Workspace dialog box updates the message in the Merge Status section to *Rebase Completed*. The status of the workspace in the View Mode pane is updated to *Rebase-in-Progress* and the version of the workspace is incremented by 1.

- 4 If there is any conflict, the Rebase Workspace dialog box displays the Resolve Conflicts button and you must click this button to resolve the conflicts.

NOTE: If you close the Rebase Workspace dialog box without clicking the Resolve Conflicts button, then all workspace-enabled objects will be read-only even though the workspace title indicates that the workspace is editable. In this case, you can change its status to *Editable* by running the Merge Report process using the Merge Reports option under the Workspace menu. For more information on how to detect conflicts in workspaces and how to apply the resolutions, see [“Detecting Conflicts in Workspaces and Applying Resolutions” on page 64](#).

- 5 In the workspace dashboard, click the Finish button on the Merged Workspaces pane to complete the merge process.

Alternatively, click the Cancel button on the Merged Workspaces pane to cancel the merge process. The workspace will be reverted to the last checkpointed version and its status is set to *Checkpointed* in the View Mode pane.

NOTE: If you are the owner of the workspace, you can set the Override option and use the Finish or Cancel buttons in the Merged Workspaces pane for that workspace.

- 6 After you click the Finish button, enter comments using the Enter Comment dialog box and click the OK button.

A message appears confirming the completion of the merge process.

NOTE: If you close the Merged Workspaces pane without clicking the Finish button while the workspace status is *Rebase-In-Progress*, then you cannot do any object changes (even though the title of the workspace includes *Editable*) until you resolve the conflicts by clicking the Finish button or clicking the Cancel button to cancel the merge process. You can reopen the Merged Workspaces pane by selecting the Merge Reports option under the Workspace menu.

- 7 Click the OK button on the confirmation message.

After the Rebase process is completed successfully:

- The status of the workspace in the View Mode pane is updated to *Checkpointed*. In the My Workspaces pane, the version of the workspace is incremented by 1 and the comments that you previously entered are also displayed.
- Workspace resolution is set to the *From Version* value by default. When you click the Finish button, the value of *From Version* is saved in your workspace. If you select the Override option, then the workspace resolution changes to *To Version* and the value of *To Version* is saved in your workspace when you finish the rebase process.

NOTE: While checking the Override flag, keep in mind that you are overwriting other users' changes that are already checked into the MAIN workspace. Therefore, you must use this option carefully. For details on various errors and conflict scenarios during the deliver and rebase processes, see [“Detecting Conflicts in Workspaces and Applying Resolutions” on page 64](#).

Detecting Conflicts in Workspaces and Applying Resolutions

In a development environment where multiple users work in parallel making configuration changes concurrently in their own private workspaces, conflicts might arise when the same objects and attributes are modified by different users in their own private workspaces, and then the changes from these users are delivered to the main branch. These situations often lead to non-trivial merges because of the conflicting changes between different users.

For workspaces, the system detects the possibility of conflicts prior to workspace delivery. If conflicts or errors are found, the system displays the issues along with the default resolutions. You must review the conflicts and apply the resolutions to resolve the issues.

In other words, during the rebase workspace process, the system enables you to:

- Identify the conflicts by viewing the conflict flags on the attributes. Notice that some conflicts must be explicitly resolved before you can submit the workspace for delivery again.
- Identify the errors by viewing the status message of the objects or attributes.
- View the status message that describes the error or conflict in details for each object or attribute.

If the system cannot resolve the conflict or the delivery process encounters an error while merging the changes, an error message appears. Although an attempt is made on all objects, the overall delivery process fails even if a single error is encountered.

NOTE: Overriding the default resolutions is only permitted while the status of rebase is *Pending Resolution*. After the status changes to *Finished*, modifications in the Merge Conflicts dialog box will not be applied.

TIP: For further analysis on errors for Siebel Tools and the Mobile Web Client, see the log file under Siebel/log: *WS_REBASE_<wsname>_<rebase attempt number>.log*. For Siebel Web Client setup of Web Tools, the log file is available in the standard Siebel log with the minimum loglevel2.

Conflicts and errors can be broadly categorized as:

- Attribute-level conflicts and errors.

The default resolutions for the attribute conflicts can be overridden by selecting the Override option. You must select the Override option with caution because you might overwrite other users' changes that were already checked into the MAIN workspace. If you select the Override option by mistake, then you can clear it to resolve the issue.

Errors can happen on attributes when the merge process is unable to set the correct value. Common errors are when there are inconsistencies in the repository and a foreign key is missing.

You can analyze the error further by viewing the appropriate log files.

- Object-level conflicts and errors, which are name conflicts, object inactive conflicts, and index violation errors.

■ Name Conflicts

A name conflict occurs when there are two different objects of the same type that have the same combination of name and parent ID. A potential violation of user key is detected as a result of the rebase operation, and the merge process selects a default resolution of renaming the object that belongs to the child workspace by appending the string *-[Rebase]*. Both object and name attribute are marked with a conflict flag in the Merge Conflicts dialog box.

If a name conflict is not resolved, then subsequent delivery fails by identifying the system-generated name. This conflict cannot be resolved by selecting the Override option, but you must finish the rebase process and resolve it in either one of the two methods described in this example:

- Suppose that both User1 and User2 branch from MAIN/5 and create their workspaces *WorkSpace_U1* and *WorkSpace_U2*, respectively.
- User1 renames and creates a new business component with the name *NewBC123*. This user then delivers changes to the MAIN workspace by creating a new version *MAIN/6*.
- User2 is unaware of User1's changes, so he renames and creates a new business component with the same name, *NewBC123*, in *WorkSpace_U2/1*.
- User2 cannot deliver his branch before starting a rebase process because this is a non-trivial merge.
- Upon doing a rebase, the merge process detects that there is a name conflict, so it renames the object in *WorkSpace_U2* to *NewBC123-[Rebase]* and issues a conflict flag for this object in the Merge Conflicts dialog box.

To resolve this conflict, both users can intend to either create two different objects or create the same object being duplicated, as in these cases:

- Case-1: Two logically different objects. In this case, both users can click the Finish button to finish the rebase process, navigate to the corresponding object in the Object Explorer pane, and then re-name the object to a valid name; for example, *NewBC123_ProjA*.
- Case-2: The same object being duplicated. In this case, both users have duplicated the same record because they were not aware of the changes made by the other. Neither user should perform the RevertObject operation from the *WorkSpace_U2* branch at this point. Instead, to enable the objects to merge appropriately, User2 should finish the rebase process, open *WorkSpace_U2/1* version explicitly, navigate to the *NewBC123* workspace and perform a Siebel Archive File (SIF) export, and open the *WorkSpace_U2* branch and import the SIF file back into the repository. Subsequently, both users must perform the RevertObject operation or inactivate the *NewBC123-[Rebase]* business component.

■ Object Inactive Conflicts

An object inactive conflict occurs when the target workspace being rebased has a change to an object that is inactivated in the source workspace. Even if the source workspace has an inactive parent object, this conflict is still alerted to let users know that the change that is made in the workspace will not be applied at runtime because either this object or its parent is inactivated.

Users can either ignore this conflict or resolve this conflict by checking the Override option for the Inactive attribute.

For example:

- Both User1 and User2 have branched from MAIN/5 and created their workspaces *WorkSpace_U1* and *WorkSpace_U2*, respectively.
- User1 inactivates the contact business component and delivers this change to the MAIN workspace by creating a new version MAIN/6.
- User2 modifies the account status *Field* by making it force active. He then tries to deliver this change to the MAIN workspace, but the system enforces a rebase.
- On rebase, the merge process detects that the changes on the account status *Field* are no longer applied as the parent contact *Business Component* is inactivated. Hence a conflict is logged against the contact *Business Component* and the corresponding inactive attribute.

To resolve the conflict, users can select the Override option for the inactive attribute on the contact *Business Component*.

■ Index violation errors

An index violation error conflict occurs when there are two different objects of the same type having the same combination of contents. A potential violation of the index is detected as a result of the rebase and the merge process cannot select any resolution. Hence the system displays an error on the object and fails the rebase process.

Users must fix this error either by manually modifying the fields involved in the index or by performing the RevertObject operation on the object (on a single version or the entire workspace). This fix ensures that there is no longer a violation and then users can run the rebase process again.

For example:

- Both User1 and User2 have branched from MAIN/5 and created their workspaces *WorkSpace_U1* and *WorkSpace_U2*, respectively.
- User1 modifies and creates a Workflow Policy Component Column with these attributes: *Name* - '*NewCompCol*', *Workflow Column Name* - '*COL1*', *Workflow Object Name*. This user then delivers this change to the MAIN workspace by creating a new version MAIN/6.
- User2 modifies and creates a Workflow Policy Component Column under the same Workflow Policy Component with these attributes: *Name* - '*AnotherCompCol*', *Workflow Column Name* - '*COL1*'.

- During the rebase process, the merge process detects the possibility of a unique key violation on the index involving the field Workflow Column Name (Workflow Column Name, Workflow Object Name). The object is marked with the key violation error in the Merge Conflicts dialog box.

To resolve this conflict, User2 must perform either of the following resolutions to resolve the error and ensure that there is no longer an index violation, and then perform the rebase process again.

- Case-1: Modify the index-based fields. User2 modifies the Workflow Policy Component Column name from *COL1* to *COL2* on the *AnotherCompCol* object.
- Case-2: Run the RevertObject process. User2 runs the RevertObject process on the *AnotherCompCol* object either for a single version or for all versions in the workspace.

To view the merge conflicts and resolutions

- 1 In the workspace dashboard, select the Workspace menu and then select the Merge Reports option.

The Merged Workspaces pane appears with the following sections:

- The Merged Workspaces section displays the status of merge, resultant version, base version, *From* version, and *To* version of the merge process.
 - The Object Differences section displays the object name, object type, top-level parent name and type, and the status of the selected object after the merge process is completed.
 - The Attribute Differences - Critical Conflicts section displays the attribute information including the workspace resolution, base version value, *To* and *From* version values, conflict, override, and so on.
- 2 (Optional) Select the Override flag to override the default workspace resolution.
 - 3 Click the Finish button to complete the merge process.

Alternatively, click the Cancel button to cancel the merge process.

Submitting Workspaces for Delivery

After you perform the checkpoint process on the latest workspace version, you can use the Submit for Delivery option under the Workspace menu to deliver the changes of your workspace to the MAIN workspace.

The Submit for Delivery process initiates the governance flow that may require one or more levels of review and approval. After the changes are submitted for delivery, the workspace is locked or made read-only to prevent further configuration changes while the workspace is being reviewed. After the workspace is approved, it is delivered into the MAIN workspace.

NOTE: When you select the Submit for Delivery option, the system identifies whether the rebase process is required with the current modifications. If the merge process is non-trivial, then an error message appears reminding you to run the rebase process before the system converts the workspace to read-only mode. When there is a trivial merge, the system changes the workspace status to *Submitted for Delivery* and converts the workspace to read-only mode.

To submit a workspace for delivery

- 1 In the workspace dashboard, select the Workspace menu and then select the Submit for Delivery option.

A confirmation message appears enabling you to confirm the submission.

- 2 Click the Yes button to start the Submit for Delivery process.

The workspace status in the View Mode pane changes to *Submitted for Delivery* and the workspace title indicates the workspace is read-only. You cannot perform any operation on the objects in the Object Explorer pane. Except for the Deliver option, other options under the Workspace menu such as Checkpoint, Revert, and Rebase are also disabled.

Alternatively, click the No button to cancel the submission. Then you can continue to make changes on the objects.

- 3 If there is a non-trivial merge issue, then an error message appears and you must perform the rebase process and then submit the workspace for delivery again.

For more information, see [“Rebasing Workspaces” on page 62](#).

Canceling the Workspace Delivery Process

Before delivering a workspace, workspace administrators must change the status of that workspace to *Submitted for Delivery* to make it ready for the delivery process. After the status of the workspace is changed to *Submitted for Delivery*, it is a read-only workspace and end users or developers cannot perform any other repository changes on it until the merging process is completed. Any repository changes to that workspace would make the delivery process fail.

To cancel the workspace delivery process, avoid the delivery failures, and allow end users or developers to be able to make repository changes on the workspaces with the status of *Submitted for Delivery*, workspace administrators must revert the status of that workspace back to *Checkpointed* and *Editable* by using the Undo Submit for Delivery option under the Workspace menu or running the UndoSubmitForDelivery command from the Command Prompt window.

NOTE: Only a workspace administrator who is also the owner of the MAIN workspace can cancel the workspace delivery process. If workspace users or workspace developers need to cancel their workspace delivery processes, they must submit the cancel requests to the workspace administrator. After the requests are approved, the workspace administrator will cancel the delivery processes as requested. For more information on how to cancel the workspace delivery process, see [“Canceling the Workspace Delivery Process” on page 68](#).

Delivering Workspaces

Delivering a workspace, or merging workspace changes, is applying the changes that are made in the workspace into its parent workspace such as the MAIN workspace or the root workspace. You must always perform the workspace delivery process to the immediate parent workspace. You cannot deliver the workspace to any arbitrary workspace.

If you are the owner of the parent workspace or if you are a workspace administrator who is also the owner of the parent workspace, then you can run the workspace delivery process. For more information on how to deliver workspaces, see [“Delivering Workspaces” on page 83](#).

Exporting Workspace Objects to Archive Files

You can export all workspace objects to an archive file, such as object definitions of the object types Business Component, Applet, View, Screen, Integration Object, Business Object, and so on.

To export workspace objects to an archive file

1 In the workspace dashboard, select and open a workspace.

2 Select the Tools menu and then select the Export Workspace to Archive option.

The Export Workspace to Archive dialog box appears. By default, the Workspace To Version field always displays the latest version of the workspace.

TIP: Optionally, you can open a specific workspace version in the Workspace Version pane to export that workspace’s object changes into an archive file. For example, when a workspace has ten versions and you want to export the workspace’s object changes through the seventh version, you can open the seventh version in the Workspace Version pane and then click the Export Workspace to Archive option under the Workspace menu.

3 On the Export Workspace to Archive dialog box, select one version from the Workspace From Version drop-down list.

- 4 Click the Export button.

The Export to Archive File dialog box appears. The status bar in this dialog box indicates the child objects that are included in the export process. When the process finishes, this dialog box displays the top-level objects in the Objects to Archive list.

NOTE: If you attempt to export a workspace that has no changes to an archive, then an error message appears saying that no objects are modified to be exported.

Comparing Workspace Versions

You can use the Compare option under the Workspace menu to identify the changes that are made to the objects in two selected workspace versions.

To compare workspace versions

- 1 In the workspace dashboard, select the Workspace menu and then select the Compare option.

The Compare Workspace dialog box appears. In the First Selection group box, the value in the Workspace Name field is populated using the value of the opened workspace name and the Version field is populated using the value of the opened version of that workspace.

- 2 In the Second Selection group box:

- a Select a workspace name from the Workspace Name drop-down list.
- b Select a version from the Version drop-down list (which includes all versions of the selected workspace.)

- 3 Click the OK button.

The status bar indicates that the process is running. When the process finishes, the Compare Objects dialog box appears, displaying all top-level objects that have differences in the two selected workspace versions.

- 4 Select one object that has such differences.

The Compare Objects dialog box displays the object properties, including the similarities and the differences for each attribute.

- 5 Close the Compare Objects dialog box by clicking the Close button.

Flattening Workspace Versions

In general, flattening workspace versions is one of the workspace administrator's tasks. A workspace administrator who is also the owner of the MAIN workspace (root, parent, or master workspace) can perform the process of flattening workspace versions using the command prompt. No other users can perform this process.

When you access Siebel Tools after the flattening workspace version process has successfully completed:

- Siebel Tools is opened in the non-workspace mode (as in the previous releases of Siebel Tools).
 - The database is set back to the workspace-disabled status with data for MAIN\LATEST in Main\0.
- For more information on how to flatten workspace versions, see [“Flattening Workspace Versions” on page 85](#).

Configuring Non-Workspace Objects

The following object types and their child objects are non-workspace objects:

- Repository
- Project
- Table
- Task
- Type
- EIM table
- Dock objects
- Schema maintenance
- Server components

Only one single public version of the object is available for all users. Hence, configuration on instances of this object is centrally controlled by the workspace administrator, who can lock all relevant projects and objects. Developers are able to configure the respective objects or projects after their requests to release the locks on these objects or projects are granted by the workspace administrator.

When developers configure non-workspace objects, they must develop and test the non-workspace objects against a dedicated environment and deliver changes (or import using the SIF-OUT/SIF-IN command) directly to the master database by requesting that the workspace administration team unlock the project.

For more information on how to control the access on non-workspace objects, see [“Controlling Access on Non-Workspace Objects” on page 87](#).

NOTE: In Siebel Tools Innovation Pack 2016, workflow objects were considered non-workspace objects. However, workflow objects are also workspace objects in Siebel Tools Innovation Pack 2017 and the later releases.

Using Workflows in Workspaces

In Siebel Tools Innovation Pack 2016, workflow objects were considered non-workspace objects. However, workflow objects are also workspace objects in Siebel Tools Innovation Pack 2017 and the later releases.

This section describes how to use Siebel workflows in Siebel workspaces. It includes the following topics:

- [“Using Workflows in Workspaces Overview” on page 72.](#)
- [“Creating New Workflows” on page 73.](#)
- [“Modifying Existing Workflows” on page 74.](#)
- [“Importing Workflows into Workspaces” on page 74.](#)
- [“Exporting Workflows from Workspaces” on page 75.](#)
- [“Validating Workflows” on page 75.](#)
- [“Debugging Workflows” on page 76.](#)
- [“Invoking Workflows” on page 77.](#)
- [“Delivering Workspaces that Use Workflows” on page 77.](#)

Using Workflows in Workspaces Overview

Using Siebel workflows in Siebel Workspace, note that:

- You cannot perform any operation on workflows that are in the MAIN/Integration branch.
You have to create the developer workspaces under the Main/Integration branch to perform the operations on workflows.
- Workflows can be imported within a workspace and they can be exported.
For more information, see [“Importing Workflows into Workspaces” on page 74](#) and [“Exporting Workflows from Workspaces” on page 75](#).
- Workflows cannot be deleted.
- The statuses of all workflows are always *In Progress* by default. Hence the Status column of the workflows is no longer required and not available in Siebel Tools starting with Siebel Tools Innovation Pack 2017.
- If you are a developer who is not the owner of the root workspace, you cannot activate a workflow but you can publish the workflow. When you click the Publish button in the Workflow Task Editor toolbar, the system moves the workflow to the *Completed* state.
If you are a developer who is also the owner of the root workspace, you can publish and activate the workflow. When you click the Publish/Activate button, the system moves the workflow to the *Completed* state and also activates the workflow. As you activate the workflow, the version number is incremented. The version changes can be viewed in the thin or thick client platform, but the version number always remains 0 (zero) in Siebel Tools.
- Only the Siebel (system) Administrator can activate the Workflows feature by either of the following actions:
 - Through Siebel Tools on the root (Main) workspace or

- Launching Siebel application and then navigating to Administration, Business Process, Workflow Deployment.

Other workflow users can search for the workflows that need to be activated and then click the Activate button on the retrieved records to activate the workflows.

For detailed information on Siebel workflows and how to manage workflows in Siebel Tools, see the *Siebel Business Process Framework: Workflow Guide*.

Creating New Workflows

You can use the New Record option under the Workflow Object List to create new workflows.

To create a new workflow

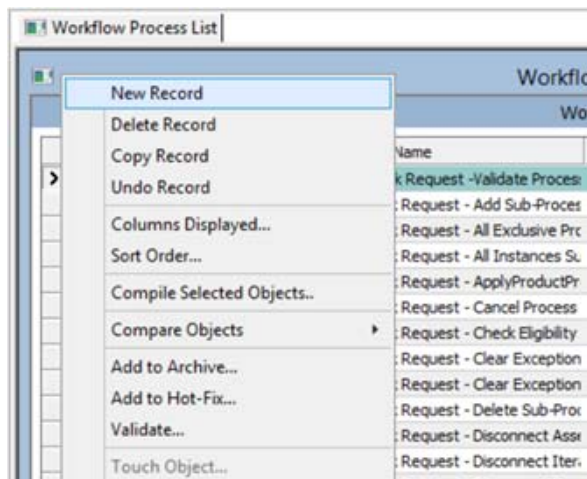
- 1 In Siebel Tools, create a new developer workspace under the Main/Integration workspace.

For more information on how to create a new developer workspace, see [“Creating New Workspaces” on page 56](#).

- 2 Under the newly created workspace, click the Workflow Process List from the Object Explorer section.

- 3 In the Workflow Process List, right-click your mouse and select the New Record option.

This example illustrates the New Record option under the Workflow Process List.



- 4 Enter the required information for the new record.
- 5 Right-click your mouse and select the Edit Workflow Process option.
- 6 Add all required workflow process steps using the Workflow Process Editor window.

For more information on how to add workflow process steps, see *Siebel Business Process Framework: Workflow Guide, Adding Workflow Process Steps*.

- 7 Save the record.
- 8 Click the Publish button in the Workflow Task editor toolbar.
- 9 From the *Workspace* menu, choose *Checkpoint*, *Submit for Delivery*, and then *Deliver*.
- 10 Click *Start Merge* in the Deliver Workspace dialog box.
- 11 After the merge finishes, open the object in MAIN and then click the *Publish/Activate* button in the Workflow Task editor toolbar.

NOTE: If you are a developer who is not the owner of the root workspace, you cannot activate a workflow but you can publish the workflow. If you are a developer who is also the owner of the root workspace, you can publish and activate the workflow.

Modifying Existing Workflows

You can use the Edit Workflow Process option under the Workflow Object List to modify existing workflows.

To modify an existing workflow

- 1 In Siebel Tools, open an existing developer workspace under the Main/Integration workspace.
For more information on how to open an existing developer workspace, see [“Opening Existing Workspaces” on page 57](#).
- 2 Click the Workflow Process List from the Object Explorer section.
- 3 Select the workflow that you want to modify, right-click, and select the Edit Workflow Process option.
- 4 In the Workflow Process Editor window, modify the workflow as needed.
- 5 Save the record.
- 6 Click the Publish button in the Workflow Task editor toolbar to activate the workflow.
- 7 From the Workspace menu, choose Checkpoint, Submit for Delivery, and then Deliver.
- 8 Click Start Merge in the Deliver Workspace dialog box.
- 9 After the merge finishes, open the object in MAIN and then click the Publish/Activate button in the Workflow Task editor toolbar.

NOTE: If you are a developer who is not the owner of the root workspace, you cannot activate a workflow but you can publish the workflow. If you are a developer who is also the owner of the root workspace, you can publish and activate the workflow.

Importing Workflows into Workspaces

You can use the Import Workflow Process option under the Workflow Object List to import a workflow into a workspace.

To import a workflow into a workspace

1 In Siebel Tools, to create a new workspace, select the Workspace menu, select the Create menu option, enter all required values, and click the OK button.

NOTE: You can also modify an existing workspace by selecting a workspace in the Workspace Explorer pane. Next, right-click the workspace and select the Open option.

2 Click the Workflow Process List from the Object Explorer section.

3 In the Workflow Process List window, right-click your mouse and select the Import Workflow Process option.

4 From the list of all available workflows, select the workflow that you want import into the current developer workspace.

5 Select the project for import.

6 Save your record.

NOTE: You can import only into an undelivered workspace. To import back the same workflow in to a different project, you still need to change both the Process Name and Display Name.

Exporting Workflows from Workspaces

You can use the Export Workflow Process option under the Workflow Object List to export a workflow from a workspace to a defined location.

To export a workflow from a workspace

1 In Siebel Tools, open the developer workspace with the workflow you want to export.

2 Click the Workflow Process List from the Object Explorer section.

3 Select the workflow to export, right-click, and select the Export Workflow Process option.

4 Specify the location path to save the exported workflow.

5 Save your record.

NOTE: If XML export is not successful, insert these values in Tools.cfg:

```
[EAI FileTransportConfigSubsys]
```

```
EAI FileTransportFolders = <path to the version_Tools\BIN folder>
```

Validating Workflows

You can use the Validate option under the Workflow Object List to validate workflows.

To validate a workflow

1 In Siebel Tools, click the Workflow Process List from the Object Explorer section.

- 2 In the Workflow Process List, select the workflow to validate, right-click, and select the Validate option.

The Validate wizard opens. If there are any errors in the workflow, the wizard displays the error information and the location of the log file.

- 3 Optionally, save or download the error log file.
- 4 Correct the errors as required.

For more information on how to fix the conflict errors and apply resolutions, see [“Detecting Conflicts in Workspaces and Applying Resolutions” on page 64](#).

- 5 Click the Start button to start the Validate Workflow process.

Alternatively, click the Cancel button to cancel the Validating Workflow process and return to the Siebel Tools window.

Debugging Workflows

Siebel Tools launches the thick client with the user-provided database details to simulate the workflow wizard. You can use the Debug tab under the Development Tools Options window to debug workflows.

To debug a workflow

- 1 In Siebel Tools, select the Development Tools Options from the View menu bar.
The Development Tools Options window opens.
- 2 Select the Debug tab from the Development Tools Options window.
- 3 Under the *Run-time start up information* and *Login Information* sections, enter the following required information:

Field Name	Example Value
Executable	<i>C:\17_17\Dclient\BIN\siebel.ext</i>
CFG file	<i>C:\17_17\Dclient\BIN\enu\uagent.cfg</i>
Working Directory	<i>C:\17_17\Dclient\BIN</i>
Arguments	<i>/h</i>
Show Workflow Primary Business Component Data	<Selected>
User Name	<i>SADMIN</i>
Password	<i>SADMIN</i>
Data source	<i>ServerDataSrc</i>

- 4 Save your work and return to the Siebel Tools window.

- 5 In the Workflow Process List, select a workflow to simulate, right-click your mouse, and select the Simulate Workflow Process option.
- 6 Click the Next button in the simulator to proceed to the next step.

The Watch window appears showing the values of workflow properties at each simulating step.

For a detailed example that uses the Watch window, see information about defining a workflow process that closes obsolete service requests in *Siebel Business Process Framework: Workflow Guide*.

Invoking Workflows

You can use one of the following methods to invoke a workflow:

- Calculating fields through the InvokeSiebelMethod
- Applet User Property such as Named Method
- RTEs
- Running a script
- Using other workflows
- Applying workflow policies
- Applying web services

Delivering Workspaces that Use Workflows

Performing the Deliver Workspace process does not activate the workflow. You must click the Publish/Activate button in the Workflow Task editor toolbar to activate the workflow.

For the detailed information on how to deliver workspaces, see [“Delivering Workspaces” on page 69](#).

Workspaces Administration

This topic describes how to perform Workspace administrative tasks in Siebel Tools. These tasks are performed only by the users who are assigned the user role Workspace Administrator. This topic includes the following information:

- [“Enabling the Workspaces Mode for Siebel Tools” on page 78](#)
- [“Enabling Workspaces in Siebel Tools and Editing the Repository Objects” on page 81](#)
- [“Canceling the Workspace Delivery Process” on page 82](#)
- [“Delivering Workspaces” on page 83](#)
- [“Flattening Workspace Versions” on page 85](#)
- [“Controlling Access on Non-Workspace Objects” on page 87](#)

- [“Publishing Tables” on page 88](#)
- [“Using Workspaces for Seed Data” on page 89](#)
- [“Support for NON-ENU Language in Rich Text Control” on page 92](#)
- [“Adding the Workspace Prefix in System Preferences” on page 94](#)

Enabling the Workspaces Mode for Siebel Tools

Workspace administrators can enable the Workspace feature (or Workspace mode) for Siebel Tools at the database level by running the executable utility. After the Workspace mode is enabled, the root workspace is added to the S_WORKSPACE table, a repository-based version record is added to the S_WS_VERSION table, and the required indexes are added to the repository tables.

Before you enable the workspaces mode in Siebel Tools, note that:

- After the Workspace mode is enabled, you should not revert back to the non-workspace mode (local database).
If you revert back to the non-workspace mode, you must flatten the workspaces before you run the Revert process or you would lose all workspace information.
- Only one mode of development (traditional or workspace-enabled) can be active at a time, but you can switch between these modes by running the Flattening process or the EnableWorkspace utility respectively.
- If workspaces are activated in the development environment, they do not need to be activated in other environments before importing the repository.
- The EnableWorkspace utility can be used to enable run-time or design repository objects and seed data. You should run this utility for both repository objects and seed data at the same time to reduce the number of performing steps.
- Before you export a repository and move it from the development environment to any other environments, you must run the Flattening process.

For more information about the Revert process, see [“Reverting to Previous Workspace Versions” on page 62](#).

For more information about the Flattening process, see [“Flattening Workspace Versions” on page 85](#).

For more information about the responsibilities of a Workspace Administrator, see [“Workspace Users” on page 52](#).

NOTE: EnableWorkspace utility and its commands are applied for all platforms, including Unix, Windows, local, server, and so on. You can call EnableWorkspace.exe or EnableWorkspace (without .exe) for the Windows platform; however, you must call EnableWorkspace (without .exe) for other non-Windows platforms.

To enable the Workspace feature for Siebel Tools

- 1 Ensure that Siebel Tools Client is installed successfully.

For more information on how to install Siebel Tools, see the *Siebel Installation Guide*.

- 2 Open the Command Prompt window from your computer by clicking the Start button and then selecting the Run option.

The Run window appears.

- 3 Enter the value *CMD* in the Open field.

The Command Prompt window appears.

- 4 Change the directory in the Command Prompt window to the *<TOOLS_HOME>/BIN* folder using this command:

```
cd <TOOLS_HOME>/BIN
```

OR

```
cd <Siebsrvr>/BIN
```

The Command Prompt window displays the following arguments and parameters, which you can use to run the EnableWorkspace utility:

- -s Siebsrvr/Tools Installation path specified (required)
 - -t Siebel Table Owner (required)
 - -u TBLO Username (required)
 - -p TBLO Password (required)
 - -o ODBC Data Source (required)
 - -d DB Platform Name (Oracle, MSSQL, DB2UDB or DB2390)
 - -l Log File Name (default: EnableWorkspace.log)
 - -w Workspace Owner Username (required)
 - -r Repository Name (default: "Siebel Repository")
 - -a Action ((R)epository, (S)eed, (B)oth) (default: (B))
 - -b RepositoryType ((R)unTime, (D)esignTime, (B)oth) (default: (B))
 - -f Seed inp file
 - -y Siebel UserName (required)
 - -z Siebel User Password (required)
 - -i Integration Branch (default: All integration branches unless passed integration branch name with this parameter)
 - -c Workspace Checkpoint Status (default: N)
 - -k Log directory (default: "Current Directory")
 - -e Language Code
- 5 Use the listed arguments and parameters to run the EnableWorkspace utility.

This example shows the arguments and parameters that are used to run the EnableWorkspace utility:

```
C:\Siebel\Tools\BIN> EnableWorkspace -s "T:\Siebel\17.0.0.0\Tools_23067" -u
ORAXXX -p ORAXXX -t ORAXXX -r "Siebel Repository" -d "Oracle" -o ORAXXX -w SADMIN
-y SADMIN -z SADMIN
```

or

```
C:\Siebel\Tools\BIN> EnableWorkspace -s "C:\Siebel\17.0.0.0\Tools" -u MSFXI219 -
p MSSQL -t dbo -r "Siebel Repository" -d "MSSQL" -o MSFXI219 -w SADMIN -y SADMIN -
z MSSQL
```

6 View the output stages, representing the progress of the tasks performed by the utility.

The Command Prompt window lists the following output stages after you run the EnableWorkspace utility:

- Stage 1 of 8: Setting the Repository ID
- Stage 2 of 8: Validate the Workspace Data
- Stage 3 of 8: Generate the Workspace Data
- Stage 4 of 8: Update the Workspace Data in all Repository Tables
- Stage 5 of 8: Update the indexes
- Stage 6 of 8: Update the Workspace Data in all Seed Tables
- Stage 7 of 8: Update the statistics for all Workspace enabled Repository/Seed Tables
- Stage 8 of 8: Post Run Cleanup

7 Confirm that the EnableWorkspace utility has processed successfully on your environment using the following steps:

- a Access and open Siebel Tools.
- b Confirm that the Siebel Tools title bar displays this title:
Siebel Tools -Siebel Repository -[MAIN/0] - Read Only-
- c On the menu bar, confirm that the Workspace menu appears between the Tools menu and the Window menu.
- d Click the Workspace menu and then select the Workspace Explorer option.
Next to the window, the Workspaces dashboard (Workspaces explorer pane) precedes the visibility View Mode pane.
- e Note that the MAIN workspace (also known as the root, parent, or master workspace) is selected, the Workspace Versions and Modified Objects panes are displayed.

NOTE: By default, Siebel Tools opens the latest version of the MAIN workspace and this workspace is set at the session. Also, the MAIN workspace is the only workspace in the Workspace explorer pane and it is always a read-only workspace.

Enabling Workspaces in Siebel Tools and Editing the Repository Objects

This topic discusses how to enable the workspaces in Siebel Tools and edit the repository objects.

To enable the workspaces in Siebel Tools and edit the repository objects

- 1 Enable the workspaces by running the EnableWorkspace utility.
For more information on how to enable workspaces, see [“Enabling the Workspaces Mode for Siebel Tools” on page 78](#).
- 2 Log into the Siebel Tools environment.
By default, the MAIN workspace or the last-modified workspace opens at the last-updated session.
- 3 Open the Workspace tab to view the existing workspaces.
- 4 Create a new workspace or edit an existing one.
To create a new workspace, select the Workspace menu, select the Create menu option, enter all required values, and click the OK button.
To modify an existing workspace, select a workspace in the Workspace Explorer pane, right-click and select the Open option.
- 5 Modify, insert, or delete any repository object as required.
NOTE: Users can modify an object without locking that object.
- 6 Perform the Checkpoint process to check the changes that you just made by selecting the Workspace menu bar and the Checkpoint option.
- 7 After all changes are checked, run the Rebase process with the parent workspace by selecting the Workspace menu bar and the Rebase option.
- 8 Confirm that the Rebase process completes successfully.
- 9 Navigate to the Conflicts Resolution view if this view is not displayed automatically.
By default, the values in the *From Version Value* list are committed to the target workspace's *To Version Value* list during the Rebase process.
- 10 Optionally, override the values in the *To Version Value* list and select the Override option.
- 11 Resolve all conflicts as needed.
- 12 Save your record.
- 13 Perform the Checkpoint process on the rebased workspace.
- 14 Click the Finish button.

For more information on how to edit workspace-enabled repository objects, see [“Editing Workspace-Enabled Repository Objects” on page 58](#).

For more information on how to enable the workspaces in Web Tools and edit the repository objects, see [“Enabling Workspaces in Web Tools and Editing the Repository Objects”](#) on page 102.

Canceling the Workspace Delivery Process

Before delivering a workspace, workspace administrators must change the status of that workspace to *Submitted for Delivery* to make it ready for the delivery process. After the status of the workspace is changed to *Submitted for Delivery*, it is a read-only workspace and end users or developers cannot perform any other repository changes on it until the merging process is completed. Any repository change to that workspace would make the delivery process fail.

To cancel the workspace delivery process, avoid the delivery failures, and allow end users or developers to be able to make repository changes on the workspaces with the status of *Submitted for Delivery*, workspace administrators must revert the status of that workspace back to *Checkpointed* and *Editable* by using the Undo Submit for Delivery option under the Workspace menu or running the UndoSubmitForDelivery command from the Command Prompt window.

NOTE: Only a workspace administrator who is also the owner of the MAIN workspace can cancel the workspace delivery process. If workspace users or workspace developers need to cancel their workspace delivery processes, they must submit the cancel requests to the workspace administrators. After the requests are approved, the workspace administrator will cancel the delivery processes as requested.

To cancel the workspace delivery process using the Undo Submit for Delivery option

- 1 In Siebel Tools, select the Workspace menu and then select the Undo Submit for Delivery option. The Undo Submit for Delivery confirmation message appears.
- 2 Click the Yes button to revert the status of the workspace from *Submitted for Delivery* to *Checkpointed* and *Editable*.

To cancel the workspace delivery process using the Command Prompt window

- 1 Open the Command Prompt window from your computer by clicking the Start button and then selecting the Run option. The Run window appears.
- 2 Enter the value *CMD* in the Open field. The Command Prompt window appears.
- 3 Change the directory in the Command Prompt window to the *<TOOLS_HOME>/BIN* folder using this command:

```
cd <TOOLS_HOME>/BIN
```

The Command Prompt window displays a list of arguments and parameters.
- 4 Run this command line to cancel the workspace delivery process:

```
siebdev /c tools.cfg /u SADMIN /p SADMIN /d ServerDataSrc /UndoSubmitForDelivery
<Workspace Name>
```

For example:

```
C:\Siebel\Tools\BIN>siebdev /c tools.cfg /u SADMIN /p SADMIN /d ServerDataSrc /
UndoSubmitForDelivery <Workspace Name>
```

After the status of the workspace is reverted, end users and developers of that workspace must perform these steps before they can make any repository change on it:

- 1 In Siebel Tools, close and then reopen the workspace.
- 2 In the View Mode pane, confirm that the reverted workspace status is now *Checkpointed* and the title of the workspace includes *Editable*.

Delivering Workspaces

Delivering a workspace, or merging workspace changes, is applying the changes that are made in the workspace into its parent workspace such as the MAIN workspace or the root workspace. You must always perform the workspace delivery process to the immediate parent workspace. You cannot deliver workspace to any arbitrary workspace.

If you are the owner of the parent workspace or if you are a workspace administrator who is also the owner of the parent workspace, then you can use one of these options to run the workspace delivery process:

- Running the Deliver command from the Command Prompt window.
 - Only workspace administrators can optionally enter comments into the command line while running the Deliver command from the Command Prompt window.
- Using the Deliver option under the Workspace menu.
 - The workspace owner and/or administrator can enter comments while running the workspace delivery process using the Deliver option.

If you are a workspace administrator without being the owner of the parent workspace, you can run the workspace delivery process by running the Deliver command from the Command Prompt window.

A workspace can be delivered once. Workspace administrators have full permissions on the MAIN workspace, including delivering the workspace. However, end users and developers have read-only access on the MAIN workspace.

To run the workspace delivery process from the Command Prompt window

- 1 On the Siebel Tools workspace dashboard, confirm that the status of the workspace is *Submitted for Delivery*.
- 2 Open the Command Prompt window from your computer by clicking the Start button and then selecting the Run option.

The Run window appears.

- 3 Enter the value *CMD* in the Open field.

The Command Prompt window appears.

- 4 Change the directory in the Command Prompt window to the *<TOOLS_HOME>/BIN* folder using this command:

```
cd <TOOLS_HOME>/BIN
```

The Command Prompt window displays a list of arguments and parameters.

- 5 If you are the owner of the parent workspace, run this workspace delivery command:

```
si ebdev /c tool s. cfg /u <User ID> /p <password> /d ServerDataSrc /Del i verWorkspace  
" <Workspace Name>"
```

For example:

```
C: \SIEBEL\TOOLS\BIN>si ebdev /c tool s. cfg /u SADMIN /p SADMIN /d ServerDabaSRC_ /  
Del i verWorkspace "<Workspace Name>"
```

If you are a workspace administrator, you can optionally enter comments into the command line. For example:

```
C: \SIEBEL\TOOLS\BIN>si ebdev /c tool s. cfg /u SADMIN /p SADMIN /d ServerDataSrc_ /  
Del i verworkspace "<Workspace Name>" "<Your Comment>"
```

NOTE: The comments in the command line will be displayed in the comment fields in the Workspace version pane of the parent workspace. If you do not provide comments in the command line, the system uses the description of the workspace for the comment fields. If the description of the workspace is not available, the system displays the value *Enter Comments* for these comment fields.

To deliver a workspace using Siebel Tools

- 1 In Siebel Tools, select the Workspace menu and then select the Deliver option.

The Enter Comment window appears.

- 2 Enter comments in the Enter Comment field.

NOTE: Entering comments is a mandatory step while delivering a workspace using the Deliver option under the Workspace menu.

- 3 Click the OK button.

The Deliver Workspace window appears displaying the From Workspace, To Workspace, Merge Status fields and the Start Merge button.

- 4 Click the Start Merge button to trigger the merging process.

The In Progress status bar and the detailed information of the merging process are displayed in the Merge Status field.

- 5 Click the Done button after the merging process is completed.

Siebel Tools reappears displaying the status and detailed information of the merging process.

- 6 In the View Mode pane, confirm that the parent workspace status is now *Checkpointed*.
- 7 (Optional) View the detailed information in the Workspace Versions pane including the merge version information and the comments that you entered while delivering the workspace.

If the merging process encounters any non-trivial merge, then the system displays this error message: *The deliver failed with error: Non-trivial merge found. Rebase needs to be done.* The View Mode pane now displays the workspace status as *Delivery Failed*. If this error message appears, you must rebase the workspace, resolve the conflicts, and then run the workspace delivery process again. For more information on how to rebase a workspace, see [“Rebasing Workspaces” on page 62](#).

The Checkpoint, Revert, Submit for Deliver, and Rebase options under the Workspace menu are disabled if the workspace deliver is completed successfully. The Rebase and Submit for Delivery options are enabled and the workspace title indicates the workspace is editable if the workspace delivery process fails.

Flattening Workspace Versions

Although the Workspace mode in the master database is enabled by default, Oracle Siebel Tools is delivered with a monolithic repository without workspaces, so there is a slight change in the preparation for the build process. In this case, you need to run a full utility directly on the master database. Because the full migration utility is not workspace-aware, the flattening workspace process has to be run on the package master database, which is derived by cloning the master database and flattening the workspaces to obtain all the repository changes from the master database without workspace information.

Thus, the flattening of workspaces essentially collapses the MAIN branch to a single version with all the latest changes and subsequently deletes all the workspaces. The flattening process also restores the time stamps throughout the hierarchy of all objects, so the upgrade process (IRM) has no impact.

A workspace administrator who is also the owner of the MAIN workspace (root, parent, or master workspace) can perform the process of flattening workspace version using the command prompt. No other users can perform this process.

After the flattening workspace version process has successfully completed, Siebel Tools will be opened in the non-workspace mode (as in the previous releases of Siebel Tools) when you access Siebel Tools.

CAUTION: In order for the flattening process to run successfully, the parameter *ServerDbODBCDataSource* under the [Siebel] section in the Tools.cfg file must point to the correct data source.

This example shows the setting of the parameter *ServerDbODBCDataSource* under the [Siebel] section in the tools.cfg file:

```
[Siebel ]
```

```
...
```

```
...
```

```
...
```

```

ReportsODBCDataSource      = Siebel Reports: Access
Local DbODBCDataSource     = m:/siebel Local DB
ServerDbODBCDataSource    = "ORAJ0141"
DockRepositoryName       = Siebel Repository
HoldExportOdbcConnection = FALSE
...
...
...

```

To flatten the workspace versions using the command prompt

- 1 Open the Command Prompt window from your computer by clicking the Start button and then selecting the Run option.

The Run window appears.

- 2 Enter the value *CMD* in the Open field.

The Command Prompt window opens.

- 3 Change the directory in the Command Prompt window to the *<TOOLS_HOME>/BIN* folder using this command:

```
cd <TOOLS_HOME>/BIN
```

- 4 Run either one of the following flattening workspace commands:

- If you want to disable workspaces on the database at the end of the flattening process, run this command to mark all indexes as non-unique indexes and flatten the workspace versions:

```
siebdev /c tools.cfg /I enu /d ServerDataSrc /u <User ID> /p <password> /
FlattenWorkspace /DisableWorkspace /DropIndexes <DB_user_name> <DB_password>
<DB_platform>
```

For example:

```
C:\Siebel\TOOLS\BIN>siebdev /c tools.cfg /I enu /d ServerDataSrc /u sadmin /p
sadmin /FlattenWorkspace /DisableWorkspace /DropIndexes sadmin sadmin ORACLE
```

NOTE: The user that uses *DB_user_name* and *DB_password* in this command must have the privileges to execute the DDL statements.

NOTE: You need to mark the indexes as non-unique index because the index columns are non-nullable, so they cannot have null values unless they are made non-unique.

- If the workspaces have to be retained on the database at the end of the flattening process, run this command to only flatten the workspace versions:

```
siebdev /c tools.cfg /u <User ID> /p <password> /d ServerDataSrc /FlattenWorkspace
```

For example:

```
C:\Siebel\TTOOLS\BIN>siebdev /c tools.cfg /l enu /d ServerDataSrc /u admin /p
admin /FlattenWorkspace
```

- 5 Confirm that the flattening workspace version process shows the process is successfully completed.

Controlling Access on Non-Workspace Objects

Currently, the following object types and their child objects remain as non-workspace objects:

- Table
- Dock Object
- EIM Interface Table
- Repository
- Workflow Policy Column
- Workflow Policy Object
- Workflow Policy Program
- Task
- Task Group
- Project

Only one single public version of the object is available for all users. Hence, configuration on instances of this object is centrally controlled by the workspace administrator, who can lock all relevant projects and objects. Developers are able to configure the respective objects or projects after their requests to release the locks on these objects or projects are granted by the workspace administrator.

For changing the existing database schema or adding new tables to the database schema, all users must follow the same process that is currently followed by all development teams; that means, respective teams need to cooperate with the workspace administrators for any change that is required in the database schema of their applications.

For Workflow and Task configurations, developers must ensure that only the 0th version on the master database exists.

NOTE: In Siebel Tools Innovation Pack 2016, workflow objects was considered the non-workspace objects. However, workflow objects are also workspace objects in Siebel Tools Innovation Pack 2017 and the later releases.

NOTE: The project must be locked for non-workspace objects before the object can be modified. The locking and unlocking of projects can be controlled by an administrator.

Publishing Tables

Tables can also be published into the S_RR_TABLE table using the same concept by which other metadata objects are published into the report repository table. A table maintains a different version number and it is tracked from the S_RR_OBJ_ITEM.TBL_VER column.

However, table versioning is different from other metadata versioning because a table is not workspace-enabled. While performing the Full Publish process, the table version is always 0 (zero), irrespective of the workspace version.

A table can be published in two ways using Siebel Tools in a Windows environment:

- Using the Apply/DDDL button.
- Using the *siebdev.exe* utility.

Publishing Table in the Source-to-Target Migration

In the source environment, a table needs to be published either by using the Apply/DDDL button or by running the siebdev.exe utility command.

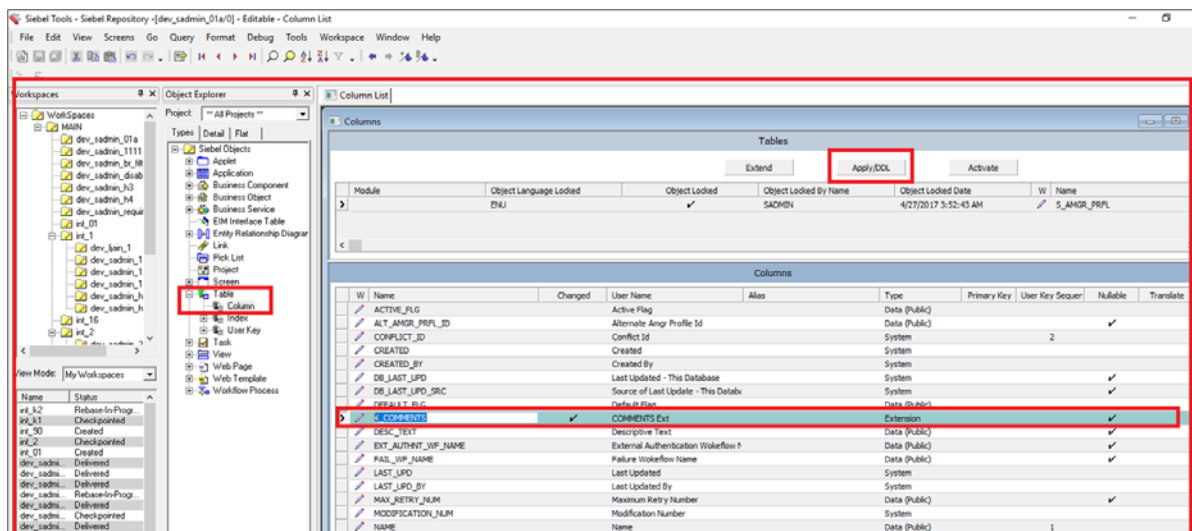
In the target environment, publishing a table is generated by the Incremental/Full source-to-target runtime repository migration. Hence, a table should not be published explicitly in the target environment.

To publish a table using the Apply/DDDL button in Siebel Tools

- 1 In Siebel Tools, update the required table schema changes as needed.
- 2 Search for the table that you want to publish.

Siebel Tools now displays a list of table row records.

This example shows the Siebel Tools window displaying the table row records after you perform the table search. The Apply/DDDL button is available after the table header.



3 Click the Apply/DDL button.

The Choose Option window appears enabling you to select the action to either apply the schema changes to the database or create the DDL file.

4 Select either the Apply or the Generate DDL option, and click the OK button to return to Siebel Tools.

- If you select the Apply option, then the Apply Schema window appears, enabling you to specify the details of table, database, and DDL. After you click the OK button, the schema is applied to the tables and the tables are republished with the next version number. The table compiled-object definitions are present in the S_RR_TABLE table in the next table version, and the list of effected tables is present in the S_RR_OBJ_ITEM table.
- If you select the Generate DDL option and click the OK button, the table will not be published but it will generate the DDL in the specified location.

To publish a table using the *siebdev.exe* utility in Siebel Tools

1 Open the Command Prompt window from your computer by clicking the Start button and then selecting the Run option.

2 In the Run window, enter the value *CMD* in the Open field.

3 In the Command Prompt window, run the following *siebdev.exe* utility command:

```
siebdev /c tools.cfg /l <LANG> /d <odbcdatasource> /u <username> /p <password> /
Incremental TablePublish <FileName>
```

The contents of *<FileName>* can be *<TableName>:<Operation>* where the operation can be *Insert*, *Update*, or *Delete*. These are the examples of the *ddl_TableSchema.txt*:

- *S_RR_OBJ_VER:Update*
- *S_RR_TEST1:Insert*
- *S_RR_TEST2>Delete*

NOTE: The *ddl_TableSchema.txt* file is always auto-generated when the schema is applied using DDLIMP. This file is generated in the same location as the log file under the file name *<log_File_Name>_TableSchema.txt*. The name of the generated file cannot be changed.

Using Workspaces for Seed Data

You can enable Siebel Workspaces for the seed data so that the seed data can be deployed within the runtime framework of the Composer Web Tools. This feature enables developers to have their work neatly isolated. It also maintains the changes that are set at the design time and the runtime as a single package. This feature requires no downtime of the seed data to deploy.

Using workspaces for the seed data, note that:

- By default, Oracle Siebel delivers seed data workspace-enabled for LOVs and you cannot disable those workspaces on seed data.

- The concept of *workspace for seed* is similar to the concept of *workspace for metadata*. The only difference is metadata changes are specific to the user whereas seed data changes apply to all users under the nearest integration branch.
- Modifying the seed data occurs in the closest integration branch; hence, all users under the same integration branch can see each other's changes immediately.
- Seed data is copied automatically when a user creates a new integration branch, so there is a separate copy of the seed data for every integration branch, including the MAIN branch.
- Modifying seed data does not create the workspace version. You must always modify metadata to create the workspace versions.
- Workspace delivery from the user-to-integration branch carries only metadata modifications because seed data modification exists in the integration branch.

However, workspace delivery from integration-to-integration workspace carries both seed data and metadata changes.

- You can access Siebel Tools or the thick client using the `\editseeddata` option to create a corporate record and then start using the workspace.

For more information on how to use the `\editseeddata` option, see *Siebel Developer's Reference Guide*.

- You cannot perform the Revert process for the changes in the seed data.

For more information on how to use database utilities to migrate metadata and seed data, see the Migration Application section in the *Siebel Database Upgrade Guide*.

To enable workspaces for seed data

- 1 Create a user workspace under the integration branch.

For more information on how to create new workspaces, see ["Creating New Workspaces" on page 56](#).

- 2 Modify the seed data and metadata as needed.

The modification of seed data in the user workspace causes the modification in the nearest integration branch.

- 3 Perform the Checkpoint process on the workspace.

For more information on how to perform the Checkpoint process, see ["Performing the Checkpoint Version Process" on page 61](#).

- 4 Submit the workspace for delivery.

For more information on how to submit the workspaces for delivery, see ["Submitting Workspaces for Delivery" on page 67](#).

- 5 Deliver the workspace to the nearest integration branch.

For more information on how to deliver workspaces, see ["Delivering Workspaces" on page 69](#).

- 6 Deliver the integration workspace to the MAIN branch.
For more information on how to deliver the integration workspace to the main branch, see [“Delivering Workspaces” on page 69](#).
- 7 If there is non-trivial change or conflict in the seed data, perform the following steps to rebase and deliver the workspace:
 - a Rebase the workspace.
For more information on how to perform the Rebase process, see [“Rebasing Workspaces” on page 62](#).
 - b Select the original value to override the current workspace value; otherwise, use the default new value.
For more information on how to fix the conflict errors and apply resolutions, see [“Detecting Conflicts in Workspaces and Applying Resolutions” on page 64](#).
 - c After the Rebase process is completed, perform the Workspace Deliver process again to deliver the workspace.
For more information on how to deliver the workspace, see [“Delivering Workspaces” on page 69](#).

Example: Inserting Seed Data Records

- 1 Create the integration workspace called INT_IP2017 under the MAIN workspace.
For more information on how to create new workspaces, see [“Creating New Workspaces” on page 56](#).
- 2 Create another workspace called EXAMPLE_WS under the INT_IP2017 workspace.
- 3 Optionally, modify the metadata record.
- 4 From the menu bar in the Siebel Tools window, choose the Screens menu, System Administration, and then select the List of Values menu item.
- 5 Change the Type value to *CUT_ACCOUNT_TYPE*.
- 6 Change the display name from *BILLING* to *BILLINGNEW*.
The new display name *BILLINGNEW* is available in the INT_IP2017 workspace.
- 7 Perform the Deliver process to deliver the EXAMPLE_WS workspace to the INT_IP2017 workspace.
For more information on how to deliver workspaces, see [“Delivering Workspaces” on page 83](#).
- 8 Perform the Deliver process again to deliver the INT_IP2017 workspace to the MAIN workspace.
- 9 Open the MAIN workspace.
- 10 Navigate to the Account List view and select the Account Class drop-down list.
The *BILLINGNEW* value is listed in the Account Class drop-down list. If the *BILLINGNEW* value is not listed, you must clear the cache by navigating to System Administration\Data>List of Values and click the Clearcache button.

11 Check to ensure that both INT_IP2017 and MAIN workspaces now list the *BILLINGNEW* value.

Support for NON-ENU Language in Rich Text Control

To enable the support for NON-ENU language in rich text control (RTC), you must run the Full Publish process using the appropriate language. In the Full Publish process, the NON-ENU language compiles all object definitions and adds those records to the respective Repository Runtime tables.

NOTE: The Full Publish process must be performed only by Siebel Tools; it cannot be performed in the Composer Web Tool. Also, not all repository objects are runtime-enabled, but only the repository objects that can be compiled are runtime-enabled.

NOTE: The database must be flattened before you perform the Full Publish process. For more information on how to flatten a database, see [“Flattening Workspace Versions” on page 70](#).

To run the Full Publish process

- 1 Confirm that Siebel Tools Innovation Pack 2017 or later release is installed in your environment. For more information about performing a new installation, see the *Siebel Installation Guide* for the operating system you are using.
- 2 Open the Command Prompt window from your computer by clicking the Start button and then selecting the Run option.
- 3 In the Run window, enter the value *CMD* in the Open field.
- 4 In the Command Prompt window, use this format to run the FullPublish process:

```
siebdev /c tools.cfg /TL <lang_code> /d <dataSource_name> /u <username> /p
<password> /FullPublish
```

For example:

```
siebedev.exe /c tools.cfg /TL ENU /d orakrv122 /u sadmi n /p sadmi n / FullPublish
```

Notice that the */d* parameter in this example refers to the ODBC data source.

Note that the *<lang_code>* parameter in the previous command can either be a single language or multiple languages. You must specify the language that needs to be published in the */TL* parameter. For more information about runtime repository business components that are language independent, see [Table 6 on page 93](#).

- This example shows the command that is used to run the Full Published process in the database with one language, English (ENU):

```
siebdev /c tools.cfg /d ServerDataSrc /u SADMIN /p SADMIN /TL ENU /FullPublish
```

- This example shows the command that is used to run the Full Published process in the database with one language, German (DEU):

```
siebdev /c tools.cfg /TL DEU /d ServerDataSrc /u sadmi n /p sadmi n /FullPublish
```

or

```
siebedev.exe /c tools.cfg /TL DEU /d orakrv122 /u sadmin /p sadmin / Full Publish
```

- This example shows the command that is used to run the Full Published process in the database with three languages: English (ENU), German (DEU), and Japanese (JPN):

```
siebdev /c tools.cfg /TL ENU, DEU, JPN /d <ServerDataSrc> /u SADMIN /p MSSQL / Full Publish
```

or

```
siebedev.exe /c tools.cfg /TL ENU, DEU, JPN /d orakrv122 /u sadmin /p sadmin / Full Publish
```

After the Full Publish process completes successfully, you can launch the Siebel application only in those languages that have the application strings and languages that are specified in the Full Publish process. In case you need to launch the Siebel application in any additional language, run Full Publish again by specifying those languages.

In the given example, Siebel application is already published in ENU, DEU, and JPN. However, if there is a requirement for FRA (French) then you need to repeat Full Publish with all the four languages together as follows:

```
siebedev.exe /c tools.cfg /TL ENU, DEU, JPN, FRA /d orakrv122 /u sadmin /p sadmin / Full Publish
```

NOTE: You cannot add or drop a new language through Incremental Publish and can do so only through Full Publish. During Full Publish, Siebel will be down.

If you need to drop a language then too repeat Full Publish. For example, you can remove JPN as follows:

```
siebedev.exe /c tools.cfg /TL ENU, DEU, FRA /d orakrv122 /u sadmin /p sadmin / Full Publish
```

Full Publish will recompile all the objects again and you can later launch Siebel in the three remaining languages.

You can also perform Full Publish in all the languages supported by Siebel by specifying ALL during FullPublish as follows:

```
siebedev.exe /c tools.cfg /TL ALL /d orakrv122 /u sadmin /p sadmin / Full Publish
```

CAUTION: Do not try the ALL command unless you require to deploy Siebel in all the languages.

This table lists all runtime repository business components that are language independent:

Table 6. Runtime repository business components that are language independent

Object Type	Runtime Object Name	Table
Prototype Applet Browser Script	Runtime Repository Prototype Applet Browser Script	S_RR_APL_BRSCPT
Prototype Applet Server Script	Runtime Repository Prototype Applet Server Script	S_RR_WEBSCRPT
Data Source Definition	Runtime Repository Data Source	S_RR_DATASRCDEF

Table 6. Runtime repository business components that are language independent

Object Type	Runtime Object Name	Table
Content Object	Runtime Repository Content Object	S_RR_CONTENTOBJ
Web Template	Runtime Repository Web Template	S_RR_WEB_TMPL
Assignment Attribute	Runtime Repository Assignment Attribute	S_RR_ASGN_ATTR
Table	Runtime Repository Table	S_RR_TABLE
Help ID	Runtime Repository Help ID	S_RR_HELP_ID
Pager Object	Runtime Repository Pager Object	S_RR_PAGER_OBJ
Search Index	Runtime Repository Search Index	S_RR_SRCH_INDEX
Text Style	Runtime Repository Text Style	S_RR_TEXT_STYLE
Icon Map	Runtime Repository Icon Map	S_RR_ICON_MAP
Integration Object	Runtime Repository Integration Object	S_RR_INT_OBJ
HTML Hierarchy Bitmap	Runtime Repository HTML Hierarchy Bitmap	S_RR_HRCHY_BMP
String Map	Runtime Repository String Map	S_RR_STRING_MAP
Link	Runtime Repository Link	S_RR_LINK
Business Object	Runtime Repository Business Object	S_RR_BUSOBJ
Type	Runtime Repository Type	S_RR_OBJ_TYPE
DLL	Runtime Repository DLL	S_RR_DLL
Search Category	Runtime Repository Search Category	S_RR_SRCH_CAT
Pick List	Runtime Repository Pick List	S_RR_PICKLIST
System Activity Object	Runtime Repository System Activity Object	S_RR_SYS_ACTOBJ

Adding the Workspace Prefix in System Preferences

To ensure all users follow standards while creating workspace names, a workspace administrator who is also the owner of the MAIN workspace needs to create a new system preference called *Workspace Prefix*.

When users create workspaces that do not have values for the *Workspace Prefix* system preference, the default prefix values are set as *dev* in the application.

To add the Workspace Prefix in system preferences

- 1 In Siebel Tools, select the Screens menu bar.

- 2 Select the System Administration option and then select the System Preferences option.
NOTE: In Web Tools, click the Tools menu and then click the System Preferences menu item.
- 3 Create a new record to add the new system preference by clicking the Add New Record button. The System Preference Administration pane appears.
NOTE: In Web Tools, click the plus icon to create a new record or select New Record from the gear icon.
- 4 Enter the value *Workspace Prefix* in the Description field.
- 5 Enter the value '*Workspace Prefix*' Value as *<text decided>* in the System Preference Name field.
- 6 Save the new system preference.

For more information about setting system preferences, see *Siebel Applications Administration Guide*.

5

Setting up Web Tools

This chapter includes information about the setup tasks that you must perform to deploy Web Tools for a new deployment and an existing deployment that you originally installed for an earlier version. It includes the following topics:

- [HTML Tags Migrated to Web Tools on page 97](#)
- [Manually Running the Migration Script on page 97](#)
- [Troubleshooting and Verifying the Results of the Web Template Migration Process on page 98](#)
- [Publishing Changes to Siebel Web Templates on page 99](#)
- [Workspaces Dashboard in Web Tools on page 99](#)
- [Enabling Workspaces in Web Tools and Editing the Repository Objects on page 102](#)
- [Navigating to the Workspace Dashboard in the Web Tools Mode Application on page 103](#)

TIP: Ensure that you enable the applicable component group while configuring the Siebel Server. For more information about configuring the components, see *Siebel Installation Guide* for the operating system you are using and *Siebel Applications Administration Guide*.

HTML Tags Migrated to Web Tools

Table 7 shows the HTML (Hypertext Markup Language) tags that are migrated to fields in Web Tools in the Web template migration process.

Table 7. HTML Tags Migrated to Web Tools

Tag Name	General Tag Purpose
div	Generates a container record for a Web template item and for its properties.
li	Generates a container record for a Web template item and for its properties.
span	Generates a container record for a Web template item and for its properties.
ul	Generates a container record for a Web template item and for its properties.

Manually Running the Migration Script

You can manually run the migration script for the Web template migration process by entering the following command in a command-line interface:

```
-jar $DbsrvrRoot\common\SWTClob.jar /s $SiebelRoot /c "$ODBCDataSource" /t $TableOwner /u $TableOwner /p $TablePassword /o $SiebelLogDir /d $DatabasePlatform /r $RepositoryName /j $WebTemplatesDir /w $WSUserName /x $WorkspaceName /b $BranchedWS /i $WebTemplateName
```

where:

- *\$DbsrvrRoot* is the path of the database server installation.
- *\$SiebelRoot* is the path of the Siebel Server installation.
- *\$SiebelLogDir* is the path to the log directory.
- *\$DatabasePlatform* is the database platform such as Oracle, DB2UDB, MSSQL, and DB2390.
- *\$RepositoryName* is the Name of the Repository such as Siebel Repository.
- *\$WebTemplatesDir* is the directory for Web Templates such as *\$SiebelRoot/Webtempl*.
- *\$WebTemplateName* is the list of template names for Incremental Migration. If not passed, all Web Templates are considered for Migration.

The following are optional parameters applicable only for Workspace Environment:

- *\$WSUserName* is the Workspace Owner user such as SADMIN.
- *\$WorkspaceName* is the Workspace Name to be created such as dev_xxxx_xxx.
- *\$BranchedWS* is the Workspace Branch under which *\$WorkspaceName* needs to be created such as MAIN.

NOTE: For Workspace-enabled environment, the New Private Workspace branch will be created and then delivered to the Integration or Main branch through the defined delivery process. The delivery process will internally publish the modified content into the RunTime Repository.

Troubleshooting and Verifying the Results of the Web Template Migration Process

After the migration script runs, you can access the SWTClob.log file to review the results of the Web template migration process. This file is available when you manually run the migration script and when the migration script automatically runs during upgrade. The log file will list the probable errors, if any, or will state the program completion message. Look for messages in the log that indicate incomplete or partial conversion or failure in conversion:

■ Incomplete conversion for <template name>

This error indicates that the template file still contains some unconverted **od** tags. The possible reason for this is that occurrences of SWE markup in the template file are not as per their documented usage. To resolve, review the remaining **od** tags in the converted template.

■ Unable to convert <template name>

This error indicates that the template could not be converted at all. The possible reason for this is an internal conversion error when you tried to migrate to ODH. To resolve, review the template content for any inconsistency.

NOTE: You can ignore the Malformed markup log that indicates that the template does not have a complete markup. It relates to tags that are not closed in the same file at the correct level.

Publishing Changes to Siebel Web Templates

The Web template migration process generates the table-based content for Siebel Web templates. If the migration installation case applies to you, then you must publish the content in these tables so that the content is implemented in the Siebel application. If the new installation case applies to you, then publication of this content is automatically completed for you.

To publish the table-based content for Siebel Web templates, enter the following command in a command window on the computer where you installed Siebel Tools:

```
siebdev /c tools.cfg /u $SIEBUSER /p $SIEBPWD /d "$ODBCDataSource" /l enu /
ful l publ i sh
```

Workspaces Dashboard in Web Tools

The Workspace dashboard enables users to view the list of all workspaces that were created and currently presented in the database. From the workspace dashboard, you can perform various operations on workspaces.

Figure 5 illustrates the Workspace dashboard in Web Tools.

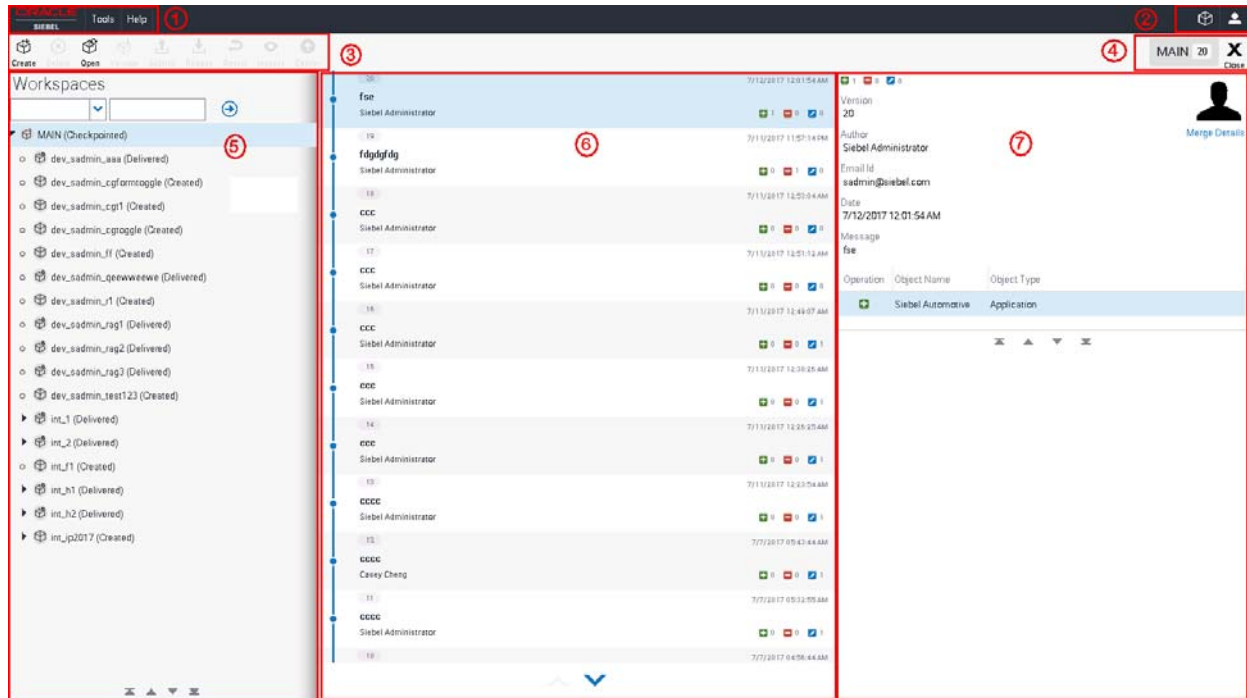


Figure 5. Example of the Workspace dashboard in Web Tools

Explanation of Callouts

Figure 5 includes the following information:

1 Toolbar and menus.

Use the toolbar and menu items to initiate various actions. The application-level menu (File, Edit, View, Navigate, Query, Tools, and Help) appears depending on your system configurations.

For more information about toolbar and menus, see the *Configuring Siebel eBusiness Applications* guide.

2 Workspace icon and user name.

- Click the Workspace icon to navigate to the Workspace Dashboard view in Siebel Tools. For more information, see [“Workspace Dashboard in Siebel Tools” on page 53](#).
- Click the User icon (a man) to display the name of current workspace user, access the Settings button, or log out. Click the Settings button to set the date and time by which updated records need to be marked as changed.

3 The Workspace menu.

The Workspace menu displays different buttons that are used to configure and manage workspaces. These buttons are active or displayed depending on the current workspace user and the selected workspace.

This table lists and describes the buttons on the Workspace menu:

Button	Description
Create	Click this button to create a new workspace.
Delete	Select a workspace and click this button to delete the selected workspace.
Open	Click this button to open an existing workspace.
Version	Click this button to view the version of the selected workspace.
Submit for Delivery	Click this button to change the status of the workspace and make it ready for delivering the changes to the MAIN workspace.
Rebase	Click this button to apply the changes that were made in the parent workspace into the current workspace.
Deliver	Click this button to deliver the changes in the workspace to the MAIN workspace. This button is available only to the user who is also the owner of the MAIN workspace.
Checkpoint	Click this button to check in the changes that you made to the current version of the workspace.
Revert	Click this button to revert the changes that you made to the current version of that workspace.
Merge Reports	Click this button to view and resolve the conflicts that occurred during the rebase process.
Compare	Click this button to view the differences that are made to the objects in two selected workspace versions.
Undo Submit for Delivery	Click this button to cancel the workspace delivery process.

4 Title of the workspace and its latest version.

This section also include the Close button that is used to close the current workspace.

5 The Workspace pane.

This pane lists all available workspaces in the database and their statuses. This pane is expandable and collapsible. You can select a workspace and then view the versions of the selected workspace and the list of modified objects for the selected version.

By default, Web Tools opens the latest version of the root workspace and this workspace is set at the session. Also, the root workspace is the only workspace in the Workspace explorer pane and it is always a read-only workspace.

Depend on the workspace that you select in this pane, different buttons are available in the Workspace menu.

This table describes the filter sections in the Workspace pane:

Option	Description
My Workspaces	Expand this section to view all workspaces that you own regardless of their status.
My In-progress Workspaces	Expand this section to view the workspaces that you own and that have the status of <i>Edit In Progress</i> .
Workspaces	Expand this section to view all workspaces that were created by all users of the current development environment.

6 The Modified Objects pane.

This pane lists the objects that were modified in the current version of the workspace. This pane displays the object name, object type, and operation type performed on the modified objects.

7 The Workspace Version pane.

This lists the versions of the selected workspace and the rebase version information. This pane also lists the author of the workspace, his/her email, date and time the workspace was modified, and the contact email, and a message of the last updated version.

Enabling Workspaces in Web Tools and Editing the Repository Objects

This topic discusses how to enable workspaces in Web Tools and edit the repository objects.

To enable the workspaces in Web Tools and edit the repository objects

- 1 Enable the workspaces by running the EnableWorkspace utility.

For more information on how to enable workspaces, see [“Enabling the Workspaces Mode for Siebel Tools” on page 78](#).

- 2 Log into the Web Tools environment.

By default, the MAIN workspace or the last-modified workspace opens.

- 3 Optionally, select Admin-Runtime Config, and then All Workspaces to view all existing workspaces.

Alternatively, select Admin-Runtime Config and then My Workspaces to view only the workspaces that you created.

- 4 Create a new workspace in the Workspaces window or open an existing one.
- 5 Modify, insert, or delete any object as required.

NOTE: Users can modify an object without locking that object.

- 6 Return to the Workspaces window.
- 7 Click the workspace version to check the changes that you just made.
- 8 Optionally, click the Edit button to continue editing the object.
- 9 Perform the Rebase process.

- 10 Perform the Deliver process to deliver the changes to the MAIN workspace.

- 11 Confirm that the Deliver process has successfully completed.

- 12 Navigate to the Conflicts Resolution view.

By default, the values in the From Version Value list are committed to the target workspace's To Version Value list during the Rebase process.

- 13 Optionally, override the values in the To Version Value list and select the Override option.

- 14 Save the record.

- 15 Resolve all conflicts as needed.

- 16 Perform the Checkpoint process on the rebased workspace.

- 17 Click the Finish button.

- 18 Click the Deliver button and then click the Start button to merge the changes to the mainline.

- 19 Click the Finish button to check in the MAIN workspace.

- 20 Optionally, open the MAIN workspace and click the Publish button to publish the newly changed values.

NOTE: If the object-level conflicts are not resolved by the system-selected solutions, then you need to open the MAIN workspace to directly edit it and perform the Publish process again.

For more information on how to enable the workspaces in Siebel Tools and edit the repository objects, see [“Enabling Workspaces in Siebel Tools and Editing the Repository Objects” on page 81](#).

Navigating to the Workspace Dashboard in the Web Tools Mode Application

You can use the Workspace (cube) icon in any Siebel application that is set in the Web Tools mode to access and open the Workspace Dashboard view, and then create or edit workspaces.

To access and open the Workspace Dashboard view in the Web Tools mode

- 1 On the Siebel application window, click the Workspace Dashboard (cube) icon after the menu bar and next to the Settings icon.
The Workspace dashboard appears.
- 2 Either create a new workspace or open an existing one.
- 3 Close the Workspace dashboard by clicking the Close (X) button in the Workspace dashboard.
The Siebel application reappears displaying the Application Edit (hammer) icon.

To enable the Workspace feature in the Web Tools mode

- 1 Ensure that Siebel Tools Client is installed successfully.
For more information on how to install Siebel Tools, see the *Siebel Installation Guide*.
- 2 Open the Command Prompt window from your computer by clicking the Start button and then selecting the Run option.
The Run window appears.
- 3 Enter the value *CMD* in the Open field.
The Command Prompt window appears.
- 4 Change the directory in the Command Prompt window to the *<TOOLS_HOME>/BIN* folder using this command:

```
cd <TOOLS_HOME>/BIN
```

OR

```
cd <Siebsrvr>/BIN
```

The Command Prompt window displays the following arguments and parameters, which you can use to run the EnableWorkspace utility:

- -s Siebsrvr/Tools Installation path specified (required)
- -t Siebel Table Owner (required)
- -u TBLO Username (required)
- -p TBLO Password (required)
- -o ODBC Data Source (required)
- -d DB Platform Name (Oracle, MSSQL, DB2UDB or DB2390)
- -l Log File Name (default: EnableWorkspace.log)
- -w Workspace Owner Username (required)
- -r Repository Name (default: "Siebel Repository")
- -a Action ((R)epository, (S)eed, (B)oth) (default: (B))
- -b RepositoryType ((R)unTime, (D)esignTime, (B)oth) (default: (B))

- -f Seed inp file
- -y Siebel UserName (required)
- -z Siebel User Password (required)
- -i Integration Branch (default: All integration branches unless passed integration branch name with this parameter)
- -c Workspace Checkpoint Status (default: N)
- -k Log directory (default: "Current Directory")
- -e Language Code

5 Use the listed arguments and parameters to run the EnableWorkspace utility.

The following example shows the arguments and parameters that are used to run the EnableWorkspace utility in a MSSQL database. In this example, *dbo* is the table owner of the MSSQL database and *MSZXM187* is the database instance that is used in this example environment.

```
EnableWorkspace -s "T:\Siebel\17.0.0.0\Tools_23067" -u ORAXXX -p ORAXXX -t ORAXXX -r "Siebel Repository" -d "Oracle" -o ORAXXX -w SADMIN -y SADMIN -z SADMIN
```

The following example shows the arguments and parameters that are used to run the EnableWorkspace utility in an Oracle database. In this example, *orakwj115* is the table owner and the user ID of the Oracle database, and these parameters are used to execute the SQL statements that exist in the EnableWorkspace utility.

```
EnableWorkspace -s "T:\Siebel\17.0.0.0\Tools_23067" -u ORAXXX -p ORAXXX -t ORAXXX -r "Siebel Repository" -d "Oracle" -o ORAXXX -w SADMIN -y SADMIN -z SADMIN
```

The following example shows the arguments and parameters that are used to run the EnableWorkspace utility in a DB2 database.

```
EnableWorkspace -s "C:\Siebel\17.0.0.0\Tools" -u DB2XXXX -p DB2 -t SIEBEL -r "Siebel Repository" -d "DB2UDB" -o DB2XXXX -w SADMIN -y SADMIN -z DB2
```

6 View the output stages, representing the progress of the tasks performed by the utility.

The Command Prompt window lists the following output stages after you run the EnableWorkspace utility:

- Stage 1 of 8: Setting the Repository ID
- Stage 2 of 8: Validate the Workspace Data
- Stage 3 of 8: Generate the Workspace Data
- Stage 4 of 8: Update the Workspace Data in all Repository Tables
- Stage 5 of 8: Update the indexes
- Stage 6 of 8: Update the Workspace Data in all Seed Tables
- Stage 7 of 8: Update the statistics for all Workspace enabled Repository/Seed Tables

■ Stage 8 of 8: Post Run Cleanup

TIP: Ensure that you enable the applicable component group while configuring the Siebel Server. For more information about configuring the components, see *Siebel Installation Guide* for the operating system you are using and *Siebel Applications Administration Guide*.

To create a new workspace in the Workspace dashboard

- 1 Click the Create button on the Workspaces dashboard.
- 2 Enter a workspace name and click the OK button.

The new workspace with the name that you entered is created and automatically opened.

To open an existing workspace in the Workspace dashboard

- 1 If necessary, expand the Main workspace (parent workspace) to display a list of the children workspaces.
- 2 Select an existing workspace from the list of the children workspaces under the Main workspace.
- 3 Click the Open button on the Workspace toolbar.

The selected workspace is opened.

6

Parallel Development Using Workspaces

Parallel Development enables Siebel workspace users to create and to work on integration branches. This tool is useful when users work in a parallel development environment. It maintains the relationship between the design time (workspaces) and the runtime repository tables even after the Parallel Development Using Workspaces feature is enabled on workspaces. Using this tool, users can create or open the workspaces from any parent or version workspace and get a consistent manifestation of the runtime metadata.

This chapter describes the Parallel Development Using Workspaces feature and how to use this feature for the Siebel Tools workspaces. These tasks are performed only by the users who have the permissions to use the Parallel Development Using Workspaces feature. This topic includes the following information:

- [“Administering Parallel Development using Workspaces” on page 107.](#)
- [“Using the Parallel Development Tools for Workspaces” on page 114](#)

Administering Parallel Development using Workspaces

This section describes how to perform the administrative tasks for the Parallel Development Using Workspaces feature. Note that only users with the administrative permissions can perform these administrative tasks:

- [“Enabling the Parallel Development Feature for Siebel Tools Workspaces” on page 108.](#)
- [“Understanding Integration Workspaces in Siebel Tools” on page 109.](#)
- [“Creating Integration Workspace in the Workspaces Dashboard” on page 110.](#)
- [“Creating Integration Workspace Using Siebel Tools” on page 111](#)
- [“Setting an Integration Workspace as the Default Workspace Branch” on page 112.](#)
- [“Setting System Preferences for Parallel-Development Workspaces” on page 112.](#)
- [“Launching the Web Client with a Specified Workspace” on page 114.](#)

For more information on how to detect errors and conflicts in the parallel-development workspaces, and then apply the resolution, see [“Detecting Conflicts in Workspaces and Applying Resolutions” on page 64.](#)

Enabling the Parallel Development Feature for Siebel Tools Workspaces

Only users that have administrative permissions can enable the Parallel Development Using Workspaces feature. In general, the Siebel administrators perform this task for all members in the development teams.

NOTE: Before you enable the Parallel Development feature for workspaces in your working environment, the Siebel environment must be in the Composer mode and the workspaces must be enabled.

For more information on how to enable workspaces in Siebel Tools, see [“Workspaces Administration” on page 77](#).

For more information on how to use the Workspaces feature in Siebel Tools as a Siebel developer or user, see [“Using the Command Line” on page 44](#).

To enable the Parallel Development feature for Siebel Tools Workspaces

- 1 As an administrator and owner of the development environment, log into the Siebel application that is set in the Composer mode.

NOTE: You can set the user preferences using either the Siebel application or Siebel Tools. To see the changes of the preferences in Siebel Tools, you must log out of Siebel Tools and then log in again. To see the changes of the preferences in Siebel application, you must restart the services of the application.

- 2 From the toolbar, choose the Screens option, System Administration and then select the System Preferences menu item.

The System Preference Administration window appears.

NOTE: In Web Tools, click the Tools menu and then click the System Preferences menu item.

- 3 In the System preferences section, set the following parameter value to Y (yes):

Enable Integration Workspaces=Y

Figure 6 illustrates Siebel Tools displaying the System Preference Administration window.

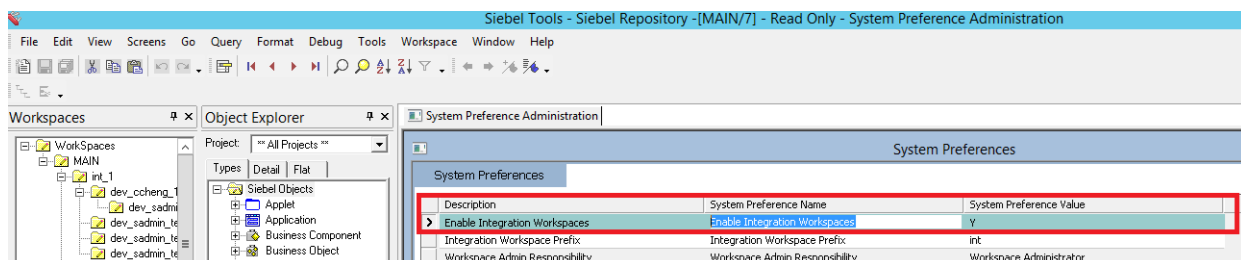


Figure 6. Example of Siebel Tools in Workspace mode displaying the workspace dashboard

- 4 After setting the parameter Enable Integration Workspaces to Y, restart the services of Siebel Server Application to reflect the new parameter value on the server side.

The environment is now ready for all users to create the parallel-development workspaces.

For information about restarting Siebel Servers, see the *Siebel System Administration Guide*.

Understanding Integration Workspaces in Siebel Tools

Users with the administrative permissions can create parallel-development integration workspaces in the production environment. Developers use these integration workspaces to set up a parallel release development environment. The names of all integration workspaces start with the prefix int_ and these workspaces have the corresponding run-time repository version.

Only administrators can perform these workspace operations on the integration workspaces: Submit for Delivery, Rebase, Delivery, and Undo Submit for delivery. By default, the Deliver and Rebase processes trigger the Publish process on an integration branch that is the parent branch of the workspace you are trying to deliver. Also, only administrators can deliver the changes in the integration workspaces to the immediate parent integration workspace, and the rebase process must be performed from the immediate parent integration workspace.

For more information on how to deliver workspace changes and rebase workspaces, see [“Using the Command Line” on page 44](#).

NOTE: Users with the developer permissions can create development workspaces but they cannot create integration workspaces in the production environment nor deliver their workspace changes to the multiple-release integration workspaces simultaneously. In fact, they can open the integration workspaces and view the version changes in read-only mode, but they cannot perform any other workspace-related actions. After the administrators successfully deliver their workspace changes into the integration workspace, developers can open the integration workspaces in the Siebel application and view the latest delivered changes.

For more information on how to create development workspaces, see [“Understanding Development Workspaces in Siebel Tools” on page 115](#), [“Creating Development Workspaces Using Siebel Tools” on page 117](#), and [“Creating Development Workspaces in the Workspaces Dashboard” on page 116](#).

After the Merge process from the source workspace to the destination workspace has completed, the Publish process runs automatically if the destination workspace is an integration workspace. The publish process is not initiated if the destination workspace is a development workspace. When the Publish process starts, processing updates all run-time repository tables with the latest metadata values. After the Publish process has successfully completed, users can see the latest compiled and published changes when they open the integration workspace in the Siebel application.

To avoid conflicts, it is recommended that you always create workspaces branching from the latest versions of the integration workspaces. For more information on how to resolve the conflicts in the emerging resolution window in Siebel Tools, see [“Detecting Conflicts in Workspaces and Applying Resolutions” on page 64](#).

The parallel-development workspaces are incremented either when:

- The child-level integration or development workspaces merge their object repository changes with them, or

- Users perform the rebase operation on an integration workspace to uptake the changes that are made in the parent integration workspace (which means the Check Point option is not available for all users, including the administrators, to manually checkpoint the version of the integration workspaces).

NOTE: You cannot perform the Revert functionality on the integration workspace.

Creating Integration Workspace in the Workspaces Dashboard

This topic describes the steps that administrators use to create an integration workspace in the Workspaces dashboard.

NOTE: Only administrators can create integration workspaces in the production environment.

To create an integration workspace in the Workspaces dashboard

- 1 As an administrator, log into the Siebel application that is set in the Composer mode.
- 2 Click the Cube icon, after the menu bar and next to the Settings icon, to navigate to the Workspace dashboard.
- 3 Select the parent integration workspace.
- 4 Click the Create button.

The Create Workspace window appears displaying the following elements:

- Name of the selected parent workspace.
- Integration Workspace option.
- Comment field.
- Cancel and Create Workspace buttons.

NOTE: The Integration Workspace option is available only for administrators. The Integration Workspace option is not available for other users, including developers.

- 5 Select the Integration Workspaces option.
In the Name field, the prefix of the workspace name changes to *int_*.
- 6 Append the workspace name text to the prefix *int_*.
For example, *int_cir4build*.
- 7 Optionally, enter a comment in the Comment field.
- 8 Click the Create Workspace button to complete the workspace creation.

For more information on how to create integration workspace using Siebel Tools, see [“Creating Integration Workspace Using Siebel Tools” on page 111](#).

Creating Integration Workspace Using Siebel Tools

This topic describes the steps that administrators use to create the integration workspaces using Siebel Tools.

NOTE: Only administrators can create the integration workspaces in the production environment.

To create a new integration workspace using Siebel Tools

- 1 Administrators log into Siebel Tools.
- 2 Navigate to the Workspace Menu bar.
- 3 Select the Create option to display the Create Workspace window.
The Create Workspace window appears displaying the following elements:
 - Integration Workspace option.
 - Enter Workspace Name field.
 - Parent Workspace drop-down list. The available options are *MAIN*, *int_1*, and *int_2*.
 - Parent Workspace Version drop-down list.
 - Add Description field.
 - OK and Cancel buttons.
- 4 Select the Integration Workspace option.
After the Integration Workspace option is selected, the prefix of the workspace name changes to *int_*.
- 5 Select one of the integration workspaces from the Parent Workspace drop-down list.
- 6 Select a version from the Parent Workspace Version drop-down list.
- 7 Optionally, enter the workspace description in the Add Description field.
- 8 Click the OK button to create the integration workspace.

For more information on how to create integration workspace in the Workspaces dashboard, see [“Creating Integration Workspace in the Workspaces Dashboard”](#) on page 110.

Setting an Integration Workspace as the Default Workspace Branch

When users access the Siebel application after they set an integration branch as the default workspace branch, the system loads the application based on the latest metadata changes of the integration workspaces that they have set as the default workspace instead of the values in the MAIN workspace. This setting enables the Siebel application to display the latest metadata values that are stored in the run-time repository tables. Every time the integration workspace is incremented by one version, the latest metadata values are stored in the run-time repository tables. When the application is reloaded, changes are reflected in the Siebel application window based on the latest version of integration workspace.

Administrators can set any integration workspace as their default workspace branch where their changes do not impact other users who are branched out from the other integration workspaces. Meanwhile, developers can work in parallel on multiple features of Siebel application without disturbing other features. Moreover, developers can verify how their latest metadata configuration appears in the Siebel application window without delivering their changes to the MAIN workspace.

To set up an integration workspace in the environment as the default workspace branch, administrators must update the parameter *WorkspaceBranchName* at the component level as shown in the following example. Notice that administrators must run this parameter by connecting to the server manager:

Command:

```
change param WorkspaceBranchName=<workspace_name> for comp <Obj Mgr_I ang>
```

Example:

```
change param WorkspaceBranchName=int_ip2017rel for comp SCCObj Mgr_enu
```

To reflect the changes, administrators must restart the services of the Siebel server.

Setting System Preferences for Parallel-Development Workspaces

This section describes the important system preferences in the Parallel Development feature for Workspaces in Siebel Tools.

Figure 7 illustrates the System Preference Administration window.

Description	System Preference Name	System Preference Value
Enable Integration Workspaces	Enable Integration Workspaces	Y
Enable Tools Workspace Deliver	Enable Tools Workspace Deliver	Y
Integration Workspace Prefix	Integration Workspace Prefix	int
Workspace Admin Responsibility	Workspace Admin Responsibility	Workspace Administrator
Workspace Prefix	Workspace Prefix	dev
AIA2.5 : SWIProductImportIO, SWIAttributeImport	Workspace Release Flag	N
Workspaces Tools Min Version	Workspaces Tools Min Version	R4g×wd99yybWdL4hocPZkwHgUC19

Figure 7. Example of System Preference Administration window

Explanation of the System Preference Administration window

Figure 7 includes the following information:

1 Enable Integration Workspaces

To enable parallel-development workspaces, administrators must set this preference to Y (*Enable Integration Workspaces=Y*).

The default value for this preference is N. After the parallel-development workspaces merge successfully to the production environment, the default value will be set to Y.

After the administrator changes the Enable Integration Workspace value, they must restart the services to reflect the latest integration workspace preferences on the server side.

2 Enable Tools Workspace Deliver

To deliver and publish the workspace changes from Siebel Tools, administrators must set this preference to Y (*Enable Tools Workspace Deliver=Y*).

The default value for this Preference is Y. If administrators do not want to deliver the workspaces from Siebel Tools, they can set this preference to N (*Enable Tools Workspace Deliver=N*).

After the administrator changes the Enable Tools Workspace Deliver value, they must restart the services to reflect the latest integration workspace preferences on the server side.

3 Integration Workspace Prefix

Integration workspaces start with the prefix value of *int*, which is derived from the preference Integration Workspace Prefix.

If an administrator wants to set up a new prefix value, they must change the Integration Workspace Prefix preference value accordingly.

After the administrator changes the integration workspace prefix value, they must restart the services to reflect the latest integration workspace preferences on the server side.

4 Workspace Admin Responsibility

The Workspace Administrator Responsibility is granted only to Workspace administrative users who will have the full privileges to manage the integration/MAIN workspace.

For more information about the detailed tasks of Workspace Administrators, see [“Workspaces Administration” on page 77](#).

5 Workspace Prefix

Development workspaces start with prefix value of *dev*, which is derived from the preference Workspace Prefix.

If an administrator wants to set up a new prefix value, he or she must change Workspace Prefix preference value accordingly.

After administrators change the workspace prefix value, they must restart the services to reflect the latest workspace preferences on the server side.

Launching the Web Client with a Specified Workspace

To launch the web client with a specified workspace on Siebel Tools, the administrator must update the *[Workspace]* and *[IngraObjMgr]* sections in the configuration file of the Siebel application.

For example, in the Siebel Financial application, they must update the *[Workspace]* sections in the *fins.cfg* file, which is the same for other applications. Administrators must update these sections in the respective applications' configuration files.

```
[Workspace]
Name = int_ip2017
Version = Latest
[InfraObjMgr]
RepositoryType = RUNTIME
```

Using the Parallel Development Tools for Workspaces

Users with the authority of a developer can perform the following tasks:

- [“Applying Parallel Development Using Workspaces Hierarchy” on page 115](#).
- [“Understanding Development Workspaces in Siebel Tools” on page 115](#).
- [“Creating Development Workspaces in the Workspaces Dashboard” on page 116](#).
- [“Creating Development Workspaces Using Siebel Tools” on page 117](#).

Applying Parallel Development Using Workspaces Hierarchy

After the administrators enable the Parallel Development feature in a working environment, developers can create the *N levels of hierarchy* for the parallel-development workspaces.

NOTE: Only administrators can enable the Parallel Development Using Workspaces feature for all users in the development team. For more information about the administrative actions of Parallel Development Using Workspaces and how to enable this feature, see [“Administering Parallel Development using Workspaces” on page 107](#) and [“Enabling the Parallel Development Feature for Siebel Tools Workspaces” on page 108](#).

You can apply a parallel-development hierarchy to create integration workspaces. The hierarchy of workspaces will be in the following combinations:

- *Integration workspace, integration workspace, followed by another integration workspace.*
This combination is applied to the N levels and you can have multiple integration or development workspaces at one level that are parallel under one parent integration workspace.
- *Integration workspace, development workspace, followed by another development workspace.*
This combination is applied to the N levels and you can have multiple development workspaces at one level that are parallel under one parent (integration or developer) workspace.

NOTE: You cannot create the integration workspaces using this combination: *Integration workspace, development workspace, followed by another integration workspace.*

For more information about the integration workspaces, see [“Understanding Integration Workspaces in Siebel Tools” on page 109](#).

For more information about the development workspaces, see [“Understanding Development Workspaces in Siebel Tools” on page 115](#).

Understanding Development Workspaces in Siebel Tools

You can create the development workspaces by branching from either an integration workspace or a development workspace at the immediate parent level.

After opening the development workspaces, you can directly edit the workspace-supported repository object and complete the development configurations.

The names of all development workspaces start with the *prefix dev_<userid>_*.

Working on the development workspaces, note that:

- Administrators can perform these workspace operations on the development workspaces: open, edit workspaces that support repository objects and complete their development configuration, submit for delivery, and rebase.

- All developers can create development workspaces branching from integration workspaces. If you are a new developer, confirm with the administrators to ensure that you have the permissions of a Composer administrator in the production database.
- While creating development workspaces on a specific release, development members should communicate with the administrators to select an integration workspace and define it as the parent workspace.
- Users with the developer permissions can create the parallel-development workspaces by selecting one of their parent release integration workspaces as the parent workspace.

For example, if the parent release integration workspace is IP20xx_apps, the developer's environment displays the parallel-development workspace trees as the following:

```
i p20xx_apps, dev_sadmi n_l 1, dev_ccheng_l 2, fo ll owed by dev_cmorri s_l 3
```

- Development teams can edit repository metadata or change the repository configuration only through the development workspaces.
- Users with the developer permissions can open the development workspaces that they own to edit the repository configuration, checkpoint the version, revert the version changes, rebase the workspaces, change the state of the workspaces to *Submitted to Delivery*, and delete the workspaces (that is in the editable state if that workspace does not have a child workspace).
- Users or developers can preview the development workspace in the Siebel application by clicking the Inspect button for the repository changes they made to that version. However, they cannot perform the Inspect task on integration workspaces.
- Workspace developers cannot deliver their workspace changes directly to the integration workspaces after they complete the repository configurations on their development workspaces. In this case, they must change the status of the workspaces to *Submitted for Delivery*, and then the Siebel administrators approve and perform the deliver process.

NOTE: Delivering the changes to the parent development workspace delivers the changes and also publishes the changes.

Creating Development Workspaces in the Workspaces Dashboard

This topic describes how to create the development workspaces in the Workspaces dashboard.

To create the development workspace in the Workspaces dashboard

- 1 Log into the Siebel application that is set in the Composer mode.
- 2 Click the Cube icon to navigate to the Workspace dashboard.
- 3 Select the parent integration workspace or any parent development workspaces.
- 4 Click the Create button.

The Create Workspace window appears displaying the following elements:

- Name of the selected parent workspace.

- Name field.
 - Comment field.
 - Cancel and Create Workspace buttons.
- 5 Append the workspace name text to the prefix *dev_ccheng_*.
For example, *dev_ccheng_bug12345*.
 - 6 Optionally, enter a comment in the Comment field.
 - 7 Click the Create Workspace button to complete the development workspace creation.

Creating Development Workspaces Using Siebel Tools

This topic describes how to create development workspaces using Siebel Tools.

To create the development workspaces using Siebel Tools

- 1 Log into the Siebel Tools environment that is set in the Composer mode.
- 2 Navigate to the Workspace Menu bar and select the Create option.
The Create Workspace window appears displaying the following elements:
 - Enter Workspace Name field.
 - Parent Workspace drop-down list. The available options are MAIN, int_1, and int_2.
 - Parent Workspace Version drop-down list.
 - Add Description field.
 - OK and Cancel buttons.
- 3 In the Enter Workspace Name field, enter a name for the new development workspace.
- 4 In the Parent Workspace drop-down list, select an integration workspace or a development workspace as the parent workspace.
NOTE: You cannot select development workspaces when they are read-only, and these read-only workspaces are not listed in Parent Workspace drop-down list.
- 5 Select a parent version.
- 6 Click the OK button to create the development workspace.

7

Using Siebel Script Editors

This chapter describes how to use the Siebel Script Editors. It includes the following topics:

- [Using the Siebel Script Editor on page 119](#)
- [Using the ST eScript Engine on page 123](#)
- [Using the Siebel Debugger on page 132](#)
- [Using the Script Profiler on page 138](#)

Using the Siebel Script Editor

This topic describes how to use the Siebel Script Editor. It includes the following information:

- [Overview of Using Siebel Script Editor on page 119](#)
- [Setting Options for the Siebel Script Editor on page 121](#)
- [Opening the Siebel Script Editor on page 122](#)

Overview of Using Siebel Script Editor

The *Siebel Script Editor* is an editor that allows you to create and maintain scripts that use Siebel VB, Siebel eScript, or Browser Script. If you cannot use a declarative configuration to implement the functionality you require, then you can use the following editors to add a script to a Siebel object:

- **Server Script Editor.** Allows you to create and modify Siebel eScript or Siebel VB.
- **Browser Script Editor.** Allows you to create and modify Browser Script that runs in the Siebel client.

For more information, including a description of scriptable events and callable methods on browser objects, see *Siebel Object Interfaces Reference*, *Siebel eScript Language Reference* and *Siebel VB Language Reference*.

About the Siebel Script Editor

You can use the Siebel Script Editor to edit Siebel VB script or Siebel eScript script. It allows you to do the following:

- Cut, copy, and paste the text from one location to another location in the Siebel Script Editor. You can also cut and paste text into the Siebel Script Editor.
- Import or export a Siebel script.
- Associate a given Siebel script with a predefined object event, such as a PreSetFieldValue event for a business component. For more information, see [“About Predefined Objects” on page 23](#).

- Debug custom script. For more information, see [“Using the Siebel Debugger” on page 132](#).
- Compile custom script. For more information on how to deliver workspaces, see [“Delivering Workspaces” on page 69](#).

The Siebel Script Editor uses a window that is similar to the Windows Notepad editor. It includes the following elements:

- Title bar
- Drop-down list that you can use to specify an object
- Drop-down list that you can use to specify an event
- Text entry window
- Vertical and horizontal scroll bars that you can use to navigate.

Object Types You Can Script

You can use the Siebel Script Editor to add a script to the following object types:

- Application
- Applet
- Business Component

An object definition for each of these object types includes a Scripted property. If this property includes a check mark, then this object includes a script.

Menus in the Siebel Script Editor

The following items are available from the File menu:

- **Import.** Imports a Siebel script.
- **Export.** Exports a Siebel script.
- **Save.** Saves a Siebel script. Make sure you save your script before you exit the editor.
- **Exit.** Closes the Siebel Script Editor window.

The following items are available from the Edit menu:

- **Cut.** Deletes the selection and saves it to the clipboard.
- **Copy.** Copies the selection to the clipboard.
- **Paste.** Copies the contents of the clipboard to the chosen area.
- **Delete.** Deletes the selection.
- **Select All.** Selects the entire script.
- **Find.** Displays the Find in Script dialog box. You can search for text or spaces.
- **Replace.** Displays the Replace in Script dialog box. You can search and replace text or spaces.

Guidelines for Using Siebel Script Editor

If you use the Siebel script editor, then it is recommended that you use the following guidelines:

- Check out or lock the project that the object definition that you modify references. If the project is not locked, then you cannot add text in the Editor window.
- To verify the format of your Siebel VB or Siebel eScript script, click the Debug menu, and then click Check Syntax. The Siebel Compiler displays any format errors it finds and indicates the lines where these errors occur.
- Save your work frequently and before you close the Siebel Script Editor. Click the File menu, and then click Save to save your work. If you close the Siebel Script Editor without saving your work, then Siebel Tools discards any modifications you make since the last save.
- You must deliver the objects that you modify in the workspace before you run a Siebel application. For more information on how to deliver a workspace, see [“Delivering Workspaces” on page 69](#).
- When pasting text into the Siebel Script Editor, avoid using two code blocks that use the same name. To do this, do the following depending on the type of script you use:
 - **Siebel eScript.** Place the code between `function Name {` and `}`. You must enclose all function code with curly braces.
 - **Siebel VB.** Place the code between `Sub Name` and `End Sub`.

Guidelines for Testing a Script

If you test a script, then it is recommended that you use the following guidelines:

- To test a script, you can click the Debug menu, and then click Start. Siebel CRM runs with the new modifications incorporated. You can also click the Start button in the Debug toolbar.
- To display an error when you test your script, you can start Siebel CRM in debug mode. To use debug mode, use the `/H` option on the start-up command line. If it encounters an error in your script, then it displays a dialog box that describes the error. To troubleshoot the error, you can open the Script Editor, click the Debug menu, and then click Check Syntax. For more information, see [“Validating Script Syntax” on page 134](#).

If a run-time error occurs in the Siebel client, or if Siebel CRM does not run in Debug mode, then it displays an error message that includes an error code. It returns control to the location in the predefined Siebel code just before the error occurred.

Setting Options for the Siebel Script Editor

You can set options for working in the Siebel Script Editor, including setting a default scripting language, specifying a location for compiling browser scripts, and defining options for debugging. Script Assist options are available only if the ST eScript Engine is enabled. For more information, see [“Overview of Using the ST eScript Engine” on page 123](#).

To set options for the Siebel Script Editor

- 1 In Siebel Tools, click the View menu, and then click Options.
- 2 In the Development Tools Options dialog box, click the Scripting tab.
- 3 Set the options.

For more information, see [“Development Options for Scripting” on page 270](#).

Suppressing Error Messages

This topic describes how to configure Siebel Tools to not display the SBL-EXL-00151 error code and error text in the pop-up error message that the RaiseErrorText method creates.

To suppress error messages

- 1 Open Siebel Tools.
- 2 Click the Screens menu, System Administration, and then click System Preferences.
NOTE: In Web Tools, click the Tools menu and then click the System Preferences menu item.
- 3 In the System Preferences list, set the Suppress Scripting Error Code system preference using values in the following table.

System Preference Name	System Preference Value
Suppress Scripting Error Code	TRUE

Opening the Siebel Script Editor

This topic describes how to open the Siebel Script Editor.

To open the Siebel Script Editor

- 1 In the Object Explorer, click one of the following object types:
 - Application
 - Applet
 - Business Component
- 2 In the Object List Editor, locate the object you want to modify.
For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
- 3 Do one of the following:
 - Right-click the object you located in [Step 2](#), and then click Edit Server Script or Edit Browser script.
 - Click the View menu, Editors, and then click Server Script Editor or Browser Script Editor.

Using the ST eScript Engine

This topic describes how to use the ST eScript Engine. It includes the following information:

- [Overview of Using the ST eScript Engine on page 123](#)
- [Enabling the ST eScript Engine on page 127](#)
- [Setting Options for the ST eScript Engine on page 127](#)
- [Opening the Script Assist Window on page 128](#)
- [Using Fix and Go on page 128](#)
- [Using Running Tool Tip on page 129](#)
- [Using Script Libraries with the ST eScript Engine on page 130](#)

Overview of Using the ST eScript Engine

The following versions of the Siebel eScript Engine are available:

- **ST eScript Engine.** Available with Siebel CRM version 7.8 and higher. The ST eScript Engine is the default Siebel eScript Engine in version 8.0.
- **T eScript Engine.** Available before Siebel CRM version 7.8.

It is recommended that you use the ST eScript Engine. Starting with Siebel CRM version 8.0, you can use only the ST eScript Engine.

Except for a few differences, the ST eScript Engine is compatible with script that you create with the T eScript Engine. This document refers to each engine by name only if it describes a difference in functionality between the ST eScript Engine and the T eScript Engine.

The ST eScript Engine is compliant with the ECMAScript Edition 4 standard. ECMAScript is the implementation of JavaScript as defined by the ECMA -262 standard. The ST eScript Engine includes the following capabilities:

- **Performance.** Provides higher throughput with a lower CPU and memory footprint in situations where you use a significant amount of script. The result is improved performance and lower maintenance on heavily scripted events.
- **Scalability.** If many users concurrently run scripts, then the performance of the ST eScript Engine is superior to the performance of the T eScript Engine.
- **Strong typing.** Supports strong typing that is compliant with the ECMAScript Edition 4 standard. Strongly typed objects result in scripts that are more functional and improved performance. The T eScript Engine does not support strong typing.
- **Functionality.** Adds new functionality, such as Script Assist, script libraries, favorites, and Fix and Go. For more information, see [“About Script Assist” on page 125](#) and [“Using Fix and Go” on page 128](#).

For more information, see *Siebel eScript Language Reference*.

About ST eScript Engine Warnings

The ST eScript Engine can display warnings that notify you of the following potential problems that might occur if you deliver your custom script:

- Referencing a method or property that is not predefined
- Referencing an undeclared identifier
- Using a variable before it initializes this variable
- Using a redundant declaration of an untyped variable
- Calling a function that includes an insufficient number of arguments

These errors might cause a run-time failure. You can use these warnings to help fix errors before you deliver your script. To enable or disable the Enable Warnings option, see [“Setting Options for the ST eScript Engine” on page 127](#).

Example of a Warning Message

The following code is an example of a compile warning message that Siebel Tools creates after a run-time failure:

```
function foo(a)
{
  var oApp: Application;
  oApp.myMethod ();
  return;
}
foo ();
Semantic Warning around line 5: Variable oApp might not be initialized.
Semantic Warning around line 5: No such method myMethod
Semantic Warning around line 10: Calling function foo with insufficient number of
arguments.
Unhandled Exception: Function expected
```

How the ST eScript Engine Determines the Type of Variables That a Script Uses

Type deduction is a feature of the ST eScript Engine that determines the type of local variables that a script uses. It scans the assignments that the script makes to each local variable. If the ST eScript Engine can successfully determine the type for all local variables, then the compiler performs strict type checks and creates statically bound code that runs faster and uses less memory. This configuration might introduce more compile warnings as a result of performing these type checks. It cannot determine the type in all situations. It is recommended that you type your script. To enable or disable type deduction, see [“Setting Options for the ST eScript Engine” on page 127](#).

Example of a Script That Deduces the Local Variable Type

The following script deduces the type of the oDate local variable to the Date. It then creates a warning about the MyMethod method. This method is not defined. This script fails at run time:

```
function goo()
{
  var oDate;
  oDate = new Date ();
  oDate.myMethod ();
  return;
}
goo ()
Semantic Warning around line 19: No such method myMethod
Unhandled Exception: 'myMethod' is not defined
```

About Script Assist

Script Assist is part of the ST eScript Engine. To help you develop a script, it inspects object definitions, and then makes information about these object definitions available to you. It includes the following functionality:

- **Syntax highlight.** Uses color to highlight reserved words, data types, operators, and other syntax in Siebel VB and Siebel eScript script. You cannot modify these colors. The following table lists the colors that it uses:

Item In the Code	Color
Reserved word or Siebel VB statement.	Blue
Data type.	OrangeRed
Operator.	Navy
String literal.	SteelBlue
Delimiter. For Siebel eScript only.	Brown
Function. For Siebel VB only.	Magenta

- **Method list.** Displays a list of methods and properties that are available for an object. For more information, see [“Icons That the Script Assist Window Contains” on page 126](#).
- **Repository inspection.** Inspects objects and object types in the repository without requiring you to type a string literal. This functionality results in fewer mistakes in your script. It also understands predefined constants for a business component method.
- **Favorites.** Uses italics to indicate the most frequently used object, method, or property name in the Script Assist window. Favorites exist for only a single Siebel Tools session. When you log out of Siebel Tools, it clears these favorites. If you create a new function, then you must add it to the declarations, and then save the script modifications. If you do not do this, then Siebel Tools does not display the function as a favorite.
- **Script libraries.** Allows you to call a business service function after you declare the business service. You do not declare property sets or make an InvokeMethod call. A script library helps you develop code that is reusable and modular. For more information, see [“Using Script Libraries with the ST eScript Engine” on page 130](#).

- **Auto complete.** Automatically completes an entry after you enter a minimum number of unique characters in the Script Assist window. For example, if you enter Bus, then Siebel Tools automatically enters the word BusComp.
- **Auto indent.** Maintains a running indent. If you press the Return key or the Enter key, then it inserts spaces and tabs that left-justifies the code.
- **Tool tips.** Allows you to view descriptions of the method arguments that you use.
- **Includes child scripts.** Script Assist can parse a script that you write on a business component, applet, or business service. You can use the Application drop-down list to choose the Siebel application that includes the child script. Script Assist displays the scripts that are written on this application object in the Script Assist window.
- **Includes scripts you write in the general section.** A script that you write in the general section of the script explorer window is available in the Script Assist window. For example, if you write a helper function named Helper in the general section of the current script, and if you start Script Assist, then it includes Helper in a pop-up window.






Using Script Assist with a Custom Siebel eScript Method

If a script references a custom Siebel eScript method on a business component, and if this script does not reside on the same business component that includes this custom method, then Script Assist does not recognize the custom method and cannot display data from it. For example, you create a business component method named MyMethod on the Account business component. You then create a script on the Contact business component that references the MyMethod method. In this example, Script Assist does not include any information about your custom MyMethod method.

Icons That the Script Assist Window Contains

Table 8 describes the methods and properties that the Script Assist window shows when you choose an object. For more information, see [“Opening the Script Assist Window” on page 128](#).

Table 8. Icons That the Script Assist Window Contains

Icon	Description
	Read-only property
	Modifiable property
	Method
	Class object
	Primitive

Enabling the ST eScript Engine

If you want to use the T eScript Engine, then you can disable the ST eScript Engine in Siebel Tools. If you disable the ST eScript Engine, you can enable it later.

To enable the ST eScript Engine

- 1 Open Siebel Tools.
- 2 Click the Screens menu, System Administration, and then click System Preferences.
 - NOTE:** In Web Tools, click the Tools menu and then click the System Preferences menu item.
- 3 In the System Preferences list, set the Enable ST Script Engine system preference to one of the following values:
 - **TRUE.** Enables the ST eScript Engine.
 - **FALSE.** Disables the ST eScript Engine.

CAUTION: Disabling the ST eScript Engine is not recommended. For help with disabling this engine, create a service request (SR) on My Oracle Support. Alternatively, you can phone Global Customer Support directly to create a service request or get a status update on your current SR.

If you want to revert to the T eScript Engine after you use the ST eScript Engine, and if you use strongly typed code while you use the ST eScript Engine, then you must undo this strongly typed code before you revert to the T eScript Engine. For more information about setting system preferences, see *Siebel Applications Administration Guide*.

- 4 Deliver your scripted objects.
 - For more information on how to deliver your workspaces, see [“Delivering Workspaces” on page 69](#).
- 5 To use the Siebel eScript Engine, exit Siebel Tools, and then reopen it.

Enabling the ST eScript Engine in the Siebel Application

If you enable the ST eScript Engine in the development environment, then make sure you also enable it in the Siebel application. You must use the same Siebel eScript Engine setting when you compile code in these environments, and you must use the same engine setting for these environments that you use to compile the code.

Setting Options for the ST eScript Engine

This topic describes how to set options for the ST eScript Engine.

To set options for the ST eScript Engine

- 1 Make sure the ST eScript Engine is enabled.
 - For more information, see [“Enabling the ST eScript Engine” on page 127](#).
- 2 In Siebel Tools, click the View menu, and then click Options.

- 3 In the Development Tools Options dialog box, click the Scripting tab.
- 4 To define the options in the Engine Settings section, use the information in the following table. It is recommended that you enable all of these options.

Option	Description
Enable Warnings	If this check box includes a check mark, then Siebel Tools displays script compile warning messages. For information, see “About ST eScript Engine Warnings” on page 124 .
Deduce Types	If this check box includes a check mark, then Siebel Tools deduces the type of local variables. For more information, see “How the ST eScript Engine Determines the Type of Variables That a Script Uses” on page 124 .
Fix and Go	If this check box includes a check mark, then Siebel Tools allows you to test and debug your script without compiling. For more information, see “Using Fix and Go” on page 128 .

Opening the Script Assist Window

You can open the Script Assist window.

To open the Script Assist window

- 1 In the Script Editor Explorer window, choose the object you want to modify.
- 2 Press CTRL+SPACE.

Siebel Tools displays the Script Assist window. This window lists all methods and properties that are available for the object you choose.

Using Fix and Go

If you enable Fix and Go, then you can edit a script in a Siebel Tools session, and then test your modifications in the Siebel client without closing this client or recompiling the script. This feature allows you to save time in script development, testing, and debugging. You can use Fix and Go only with a server script and the ST eScript Engine.

To use Fix and Go

- 1 Enable Fix and Go in the Development Tools Options window.
For more information, see [“Setting Options for the ST eScript Engine” on page 127](#).
- 2 Create a server script in the Siebel Script Editor, save it, and then deliver the repository.
If you attempt to save a script that includes a format error, then Siebel Tools displays a script error message that prompts you to fix the line or the lines that cause the error. For more information on how to deliver your workspaces, see [“Delivering Workspaces” on page 69](#).

- 3 Use the Siebel Debugger to run the script.
- 4 If an error occurs, then stop the script.
- 5 Make modifications, save your script, and then run it again.

You must save and deliver all script modifications before you exit Siebel Tools. If you do not save your modifications, then they are lost.

Using Running Tool Tip

Running Tool Tip is a help feature in the Siebel Script Editor that you can use to write a script. If you enter a method name in the Running Tool Tip window, then it displays the method arguments that you can use with this method. If you enable Script Assist, then Siebel Tools enables Running Tool Tip, by default. For more information, see [“About Script Assist” on page 125](#).

To use Running Tool Tip

- 1 Open the Siebel Script Editor.

For more information, see [“Opening the Siebel Script Editor” on page 122](#).

- 2 In the Script Editor window, enter a method name followed by an open parenthesis.

Siebel Tools displays the Running Tool Tip window. This window displays the method name and the method arguments. It uses italicized text to suggest an argument. Siebel Tools hides this window after you enter all the required method arguments.

Figure 8 includes the Running Tool Tip window.

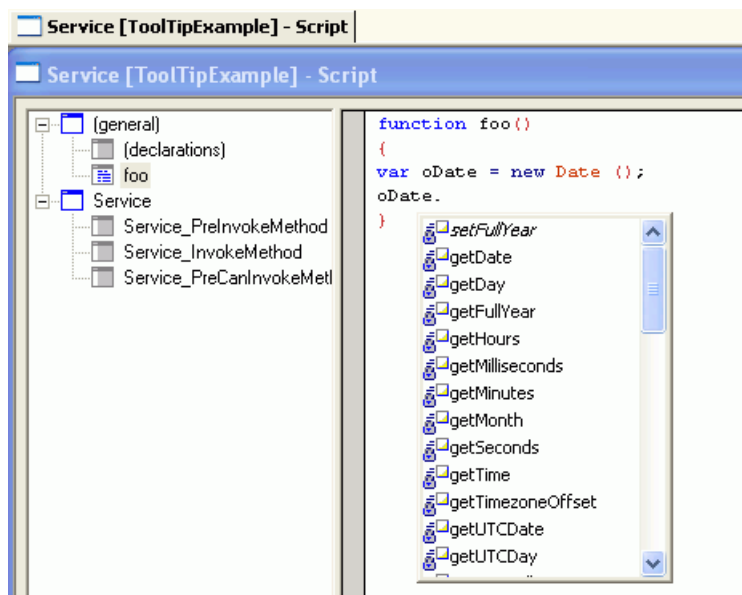


Figure 8. Running Tool Tip Window in the Siebel Script Editor

How Running Tool Tip Differs from Tool Tips in Script Assist

If you enter a function name followed by a left parenthesis in the Script Editor, then Script Assist or Running Tool Tip shows depending on the type of function you enter.

Running Tool Tip Window

If you enter a simple call expression, then Siebel Tools displays the Running Tool Tip window. A call expression allows you to send anything to the function according to the type of argument. For example, if you use the following code to enter a data object function, then Siebel CRM can send a value that you choose to this function:

```
Date.SetFullYear
```

For example, you enter the following code:

```
var oDate = new Date();
oDate.SetFullYear(
```

In this example, Siebel Tools displays the following code in the Running Tool Tip window:

```
oDate.SetFullYear(year, month, date)
```

Script Assist Window

If you enter a collection function, then Siebel Tools displays the Script Assist window. This window includes a list of methods and properties that you can choose for an object. A *collection function* is a type of function that includes a finite set of values. You can send values to a collection function. For example, if you enter the following code, then Siebel Tools displays fields that are defined only for this business component:

```
BusComp.GetFieldValue
```

The following code is an example:

```
var bo = TheApplication().GetBusinessObject(
```

For more information about using functions and expressions in Siebel eScript, see *Siebel eScript Language Reference*.

Using Script Libraries with the ST eScript Engine

A *script library* is a part of the ST eScript Engine that allows you to call a business service method from a script. This topic describes how to make a custom business service method available to a business service script library and how to call this method in the script library. Using a script library is optional. Siebel CRM supports code you write before Siebel CRM version 8.0. For more information about script libraries, see *Siebel eScript Language Reference*.

Making a Custom Business Service Method Available in a Script Library

This topic describes how to make a custom business service method available in a script library.

To make a custom business service method available in a script library

- 1 Create a script for a business service method.
- 2 Validate the script.
For more information, see [“Validating Script Syntax” on page 134](#).
- 3 Save the script.
- 4 Make sure the External Use property of the business service contains a check mark.
Siebel Tools adds the custom business service method to the script library.

Using a Script Library to Call a Custom Business Service Method

After you make a business service available for use in Siebel Tools, the following items apply:

- You can call a business service method of this business service from another script.
- The Script Assist window displays these business service methods. For more information, see [“About Script Assist” on page 125](#).

To use a script library to call a custom business service method

- 1 Make sure the following options are enabled:
 - Enable Method Listing
 - Enable Auto Complete
 For more information about setting these options, see [“Setting Options for the Siebel Script Editor” on page 121](#).
- 2 In the Siebel Script Editor, type the name of a business service object followed by a period (.).
Siebel Tools displays the business service methods that are available for the business service.
- 3 Choose the business service method that you want to add to the script.
- 4 Validate the script.
For more information, see [“Validating Script Syntax” on page 134](#).

Example of Using a Script Library

In this example, the mathService business service is enabled for use in Siebel Tools. It uses a business service method named square:

```
function square (x)
{
  return (x * x);
}
```

If you use a script library to call an argument, then the compiler examines the argument types to make sure the argument that your script calls is valid and is compatible. The following code calls a business service method named square:

```
var oBS: Service = TheApplication().GetService ("mathService");
var value = 10;
var square_value = oBS.square (value);
```

Figure 9 displays a list of the business service methods that are available for the mathService script library. This list displays the following code:

```
var square_value = oBS
```

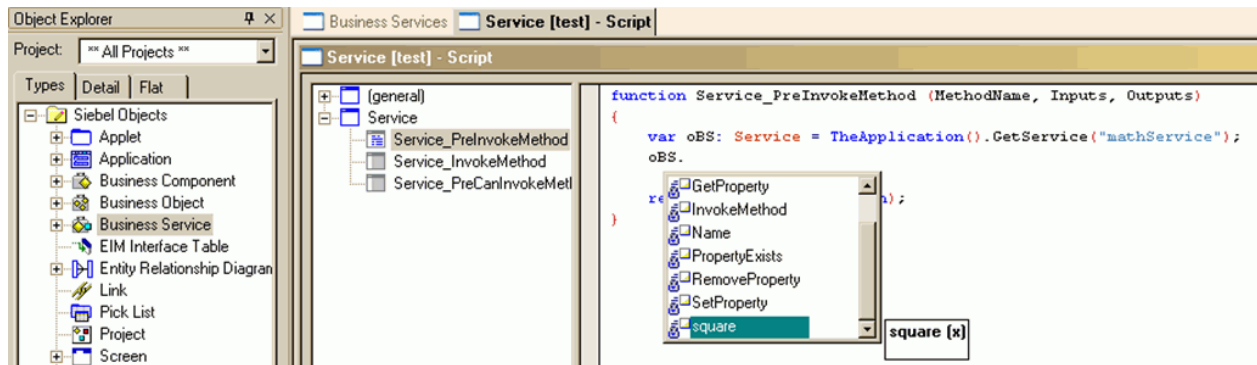


Figure 9. Example of a Script Assist Window That Displays the Business Service Methods That Are Available for the MathService Script Library

Using the Siebel Debugger

This topic describes how to use the Siebel Debugger. It includes the following information:

- [Overview of Using the Siebel Debugger on page 132](#)
- [Setting Debug Options on page 133](#)
- [Accessing the Siebel Debugger on page 134](#)
- [Validating Script Syntax on page 134](#)
- [Using Breakpoints on page 135](#)
- [Using the Calls Window on page 135](#)
- [Using the Watch Window While it Monitors a Script on page 136](#)
- [Tracing a Script on page 136](#)

Overview of Using the Siebel Debugger

The Siebel Debugger can help you edit a script and remove an error from a script that you write in Siebel VB or Siebel eScript. It displays variables and their values in the Siebel Script Editor window and a diagnostic window. You can use it to slow down or suspend a script from running so that you can monitor the logical flow of your script and to examine the contents of variables. You can use the Siebel Debugger to do the following:

- **Set a breakpoint.** Allows you to use the Debugger so that you can examine the state of the program at this line. For more information, see [“Using Breakpoints” on page 135](#).
- **Step over a line of code.** For example, if the current code line calls a subroutine, then the Debugger skips this line and stops at the next line. It does not call the subroutine.
- **Step into a subroutine.** Runs one line of code according to the following:
 - **The current line calls a subroutine or function.** The Debugger stops at the first line of the subroutine or function that the current line calls.
 - **The current line does not call a subroutine or function.** The Debugger stops at the next code line.
- **Examine the value of a variable.** The Siebel Debugger includes a Watch window that displays variables and their values. You can examine these values while a script runs.

Setting Debug Options

This topic describes how to set debug options.

To set debug options

- 1 In Siebel Tools, click the View menu, and then click Options.
- 2 In the Development Tools Options dialog box, click the Debug tab, and then set the options.

The following table includes example settings for the Debug tab. For more information, see [“Development Options for Debugging” on page 272](#).

Option	Value
Executable	siebel.exe
CFG File	c:\siebel\bin\w32u\enu\uagent.cfg
Working Directory	c:\siebel\bin\w32u
Arguments	<p>You can use the following arguments:</p> <ul style="list-style-type: none"> ■ /h. Enables local debugging of server scripts. ■ /l. Sets the language that Siebel Tools uses for the debug session. <p>You can include these arguments on the same line and in any order. For example, you can use one of the following for ENU:</p> <ul style="list-style-type: none"> ■ /h /l ENU ■ /l ENU /h
User Name	sadmin
Password	*****

Option	Value
Data Source	Sample
Enable Profiler	Contains a check mark.

- 3 Click the Scripting tab.
- 4 In the Debugging section, set the options, and then click OK.

For more information, see [“Development Options for Visualization Views” on page 270](#).

Accessing the Siebel Debugger

You can access the Siebel Debugger.

To access the Siebel Debugger

- Do one of the following:
 - Open the Siebel Script Editor, set a breakpoint in your script, and then click the Start button.
For more information, see [“Using Breakpoints” on page 135](#).
 - Run a script that includes a run-time error.
If a script includes a run-time error, then Siebel Tools suspends the script, starts the Debugger, and highlights the line that includes the error. An unhandled Siebel VB or eScript error is an example of a run-time error.

Validating Script Syntax

The debugger includes a syntax checker that you can use to make sure your script compiles properly.

CAUTION: The Check Syntax feature identifies only syntax errors and errors that occur if an object or variable is not initialized. It does not identify other types of errors and it cannot trap an error that might cause a run-time error. It examines only script that is attached to the current active object. If an error exists in another script, then you cannot compile the repository.

To validate script syntax

- 1 Open the Siebel Script Editor, click the Debug menu, and then click Check Syntax.
Siebel Tools does a test compile, and then does one of the following:
 - **It finds no error.** It displays nothing.
 - **It finds an error.** It displays a dialog box that describes the error.
- 2 If the script includes an error, then click Go to Line in the dialog box.
Siebel Tools positions the cursor on the script line that causes the error and highlights this line.

- 3 Correct the code and repeat [Step 1](#).

If the syntax of the line that you modified is correct, then the dialog box displays the next error it finds, if any.

- 4 Repeat [Step 1](#) and [Step 2](#) until you correct all errors.
- 5 Click the File menu, and then click Save.
- 6 Deliver your changes to the repository.

For more information on how to deliver your workspaces, see [“Delivering Workspaces” on page 69](#).

Using Breakpoints

A *breakpoint* is a marker in a line of code that stops code from running at this line. It allows you to use the Debugger so that you can examine the state of the program at this line.

To use breakpoints

- Do one of the following:
 - **You are editing a script.** Place the cursor on the line of code where you want to set a breakpoint, and then press F9.
 - **You are debugging script.** Click a line of code.

If a breakpoint does not already exist on this line, then Siebel Tools adds a breakpoint. If a breakpoint already exists on this line, then Siebel Tools removes it.

Using the Calls Window

The Calls window includes a list of subroutine and function calls that Siebel Tools runs prior to the current code line.

To use the Calls window

- 1 Open the Siebel Script Editor.
- 2 Click the Calls button in the Debugger toolbar while you run the Debugger.

Each line in the Calls window represents a subroutine that Siebel Tools entered but did not complete. If you choose an entry in this list, then the following occurs:

- The Debugger window displays the line of code that makes this call.
- The Variable window displays the variables that are associated with the procedure that makes the call.

Using the Watch Window While it Monitors a Script

The Watch window displays variables and variable values. You can use it to monitor these values while a script or workflow process runs. The Watch window supports the following variable types:

- Local variables
- Global variables
- Shared global variables
- Application variables
- Persistent profile properties
- Dynamic profile properties

For a detailed example that uses the Watch window, see the information about defining a workflow process that closes obsolete service requests in *Siebel Business Process Framework: Workflow Guide*.

To use the Watch window while it monitors a script

- 1 Attach a script to an object, and then deliver object's changes.
For more information on how to deliver your workspaces, see ["Delivering Workspaces" on page 69](#).
- 2 Start the Siebel client from the Siebel Debugger.
You can press F5 or click the Start icon on the Debug menu. Make sure the /h argument is set in the Debug options. For more information, see ["Setting Debug Options" on page 133](#).
- 3 Display the Watch window.
You can press SHIFT+F9 or click the Watch button (glasses icon) on the Debug toolbar.

Tracing a Script

You can run a trace on an allocation, event, or SQL command. You can start a trace for a user account, such as your development team. The Siebel Server saves the trace information to a log file. Tracing a script is not the same as tracing a file. For more information about tracing a file, see the information that describe the Trace, TraceOn, and TraceOff methods in *Siebel Object Interfaces Reference*.

To trace a script

- 1 Log in to the Siebel client, navigate to the Administration - Server Management screen, and then the Components view.
- 2 Choose the component that you want to trace.
- 3 Navigate to the Component Event Configuration view.

4 Locate the Object Manager Extension Language Log event.

If this event does not exist, then the component you chose in [Step 2](#) does not support logging. Most components support logging, but some do not. If necessary, repeat [Step 2](#) but choose another component.

5 Set the Log Level to 1.

To disable logging when you are done, you can set the Log Level to 0 (zero).

6 Navigate to the Component Parameters view.

7 (Optional) To display only the script tracing parameters, enter a query using the values from the following table.

Field	Value
Parameter Alias	Trace*
Subsystem	Object Manager

8 Set one or more tracing parameters using the values from the following table.

Object to Trace	Alias	Description
Allocations	TraceAlloc	Enter 0 (zero) to disable logging. Enter 1 to enable logging.
Events	TraceEvents	
SQL Commands	TraceSql	
Users	TraceUser	Enter a comma-separated list of user names. Do not use spaces. For example, you can enter the following: sadmi n, mmasters, hki m, cconnors

To set tracing parameters, enter a value depending on when the modification must take affect:

- **Immediately.** You modify values in the Current Value column.
- **After a restart.** You modify values in the Value on Restart column.

Example of a Trace

The following code is an example of a trace that Siebel Tools creates after the Business Service Simulator runs:

```
ObjMgrExtLangLogObjMgrExtLangLog000000080475f1578: 02007-12-12 07: 33: 45[User: ] SQLBIND, 90, 1, Y.
ObjMgrExtLangLogObjMgrExtLangLog000000080475f1578: 02007-12-12 07: 33: 45[User: ] COMPILE, END, <unknown>.
ObjMgrExtLangLogObjMgrExtLangLog000000080475f1578: 02007-12-12 07: 33: 45[User: ] EVENT, BEGIN, Service
[sam], Service_PrelInvokeMethod. ObjMgrExtLangLogObjMgrExtLangLog000000080475f1578: 02007-12-12 07: 33: 45
[User: ] EVENT, END, Service [sam], Service_PrelInvokeMethod. ObjMgrExtLangLogObjMgrExtLangLog0
00000080475f1578: 02007-12-12 07: 34: 39[User: ] SQLSTMT, 91,
```

Example of a Detailed Trace

The following code is an example of a trace that Siebel Tools creates if the log level is set to high. For brevity, only part of this log file is included:

```
2021 2007-12-12 07:46:29 2007-12-12 07:56:46 -0700 000003fd 001 003f 0001 09 SCCObjMgr_enu 11534365 5452
780 d:\21022\ses\siebsrvr\log\SCCObjMgr_enu_0011_11534365.log 8.1 [21022] ENU
```

```
ObjMgrLogInfo3000000b9475f1578:02007-12-12 07:46:29(modpref.cpp (949)) SBL-DAT-50803: SharedFileReader:
D:\fs\userpref\SADMIN&Siebel Universal Agent.spf_SDCHS21N625_5452_780: File successfully opened.
```

```
ObjMgrLogInfo3000000b9475f1578:02007-12-12 07:46:29(modpref.cpp (949)) SBL-DAT-50804: LoadPreferences:
D:\fs\userpref\SADMIN&Siebel Universal Agent.spf: Preferences successfully loaded.
```

```
ObjMgrExtLangLogObjMgrExtLangLog0000000b9475f1578:02007-12-12 07:46:29[User: ] SQLSTMT, 1, SELECT, SELECT
```

```
T1.CONFLICT_ID,
T1.DB_LAST_UPD_SRC,
CONVERT (VARCHAR (10),T1.DB_LAST_UPD, 101) + ' ' + CONVERT (VARCHAR (10),T1.DB_LAST_UPD, 8),
CONVERT (VARCHAR (10),T1.LAST_UPD, 101) + ' ' + CONVERT (VARCHAR (10),T1.LAST_UPD, 8),
CONVERT (VARCHAR (10),T1.CREATED, 101) + ' ' + CONVERT (VARCHAR (10),T1.CREATED, 8),
T1.LAST_UPD_BY,
T1.CREATED_BY,
T1.MODIFICATION_NUM,
T1.ROW_ID,
T1.BUILD_NUMBER,
T1.CURR_WEBFILE_VER,
T1.ENTERPRISE_NAME,
T1.PREV_WEBFILE_VER,
T1.RECORD_INFO_TEXT,
T1.RECIDENTIFIER
FROM
    dbo.S_UIF_WEB_FILE T1
WHERE
    (T1.RECIDENTIFIER = ?).
```

```
ObjMgrExtLangLogObjMgrExtLangLog0000000b9475f1578:02007-12-12 07:46:29[User: ] SQLBIND, 1, 1,
16:sdchs21n625:4330siebel.
```

Declaring Functions and Procedures in Siebel VB

The Siebel Compiler compiles Siebel VB functions and procedures in alphabetical order for each object definition. If a function or procedure calls another function or procedure that is not defined, then the compiler creates an error message that is similar to the following:

```
function_name Is An Unknown Function
```

To avoid this error, you must use the Declare statement to declare the function or procedure in the general declarations section. Siebel eScript does not require you to declare these functions. For more information, see *Siebel VB Language Reference*.

Using the Script Profiler

This topic describes how to use the Script Profiler. It includes the following information:

- [Overview of the Script Profiler on page 139](#)
- [Enabling the Script Profiler and Line Profiler on page 141](#)
- [Running the Script Profiler on page 142](#)

Overview of the Script Profiler

The *Siebel Script Performance Profiler* is a part of the ST eScript Engine that allows you to observe and monitor the performance of a script. For brevity, this guide refers to the Siebel Script Performance Profiler as the Script Profiler.

If you start Siebel CRM in debug mode from Siebel Tools, then the Script Profiler gathers and displays data for all executable scripts. It displays data in the Script Performance Profiler window in Siebel Tools and updates this data if a script runs. You can use this data to monitor script performance, identify performance bottlenecks, and compare performance with previous script runs. The Script Profiler allows you to do the following:

- **Use the Call Tree view.** Displays profile data as a tree of function calls. For more information, see [“Example of the Script Performance Profiler Window” on page 139](#).
- **View function profile and line profile.** Line profile data includes the call count and total time spent for each line that runs in a function. Line profile information is available only for a compiled script and is not available for a line that does not run.
- **Save profile data to file.** You can save profile data that the Script Performance Profiler window shows to a text file.
- **Use the Siebel Script Debugger and the Siebel Script Profiler.** You can use the Script Profiler and the Script Debugger at the same time. Profile data is consistent even if the function uses Debugger functionality, such as a breakpoint.
- **View the script source.** Siebel Tools opens the objects that a script references in the Script Editor window. You can double-click a function name or line number to view the script from the Script Performance Profiler window. The View Source option is available only for a compiled script. It is not available for a runtime script.

Example of the Script Performance Profiler Window

[Figure 10](#) displays an example of the Script Performance Profiler window. It allows you to examine script performance. It displays a line number for each code line, total time spent to run that line, the number of times Siebel CRM runs the line, and so on.

Function / Source Line	Call Count	Total T...	Max T...	Min T...	Total Self ...	Max ...	Min ...
[-] Root	1	47	0	0	0	0	0
[-] Service[Test1]::Service_PreInvokeM...	2	47	31	16	0	0	0
... 4:a=TheApplication().GetBusObj...	1	0	0	0			
... 6:b=a.GetBusComp("Account");	1	16	0	0			
... 7:axis(a,b);	1	0	0	0			
... 8:return (ContinueOperation);	1	0	0	0			
... Application::TheApplication	2	0	0	0	0	0	0
... Application::GetBusObject	2	0	0	0	0	0	0
... BusObject::GetBusComp	2	47	31	16	47	31	16
[-] Service[Test1]::axis	2	0	0	0	0	0	0
... 3:var isRecord,count=1,det...	1	0	0	0			
... 4:date1=new Date();	1	0	0	0			
... 5:isRecord=true;	1	0	0	0			
... 6:while (isRecord)	2	0	0	0			
... 8:isRecord= false;	1	0	0	0			
... 9:count++;	1	0	0	0			
... 10:}	1	0	0	0			
... 11:date2=new Date();	1	0	0	0			
... 13:return (CancelOperation);	1	0	0	0			
... global::_operatorNew	4	0	0	0	0	0	0
... global::ToBoolean	4	0	0	0	0	0	0

Figure 10. Example of the Script Performance Profiler Window

Description of the Columns in the Script Performance Profiler Window

Table 9 describes the columns that the Performance Profiler window includes. For the Source Line, Call Count, and Total Time columns, if line profiling is enabled for a function, then the line profile displays as child nodes of a function in the Call Tree view and in the Flat Profile view.

Table 9. Description of Columns in the Script Performance Profiler Window

Column	Description
Function/Source Line	Function name and the name of the object that contains the function. For example: Service [ATP Check]::service_preinvokeMethod and BusComp[Account]::foo
Call Count	The number of times Siebel CRM calls a function. You can use one of the following views: <ul style="list-style-type: none"> ■ Flat Profile view. Displays the total number of times Siebel CRM calls the function. ■ Call Tree view. Displays the number of times Siebel CRM calls the function in the current position in the call tree. It displays profile data as a tree of function calls. A node represents each function in a call sequence. You can drill down into each node to examine a subtree. You can use the Expand All option in the Profiler toolbar to expand all nodes.
Total Time	Total milliseconds spent in this function and in nested functions.
Max Time	Maximum milliseconds spent in this function and in nested functions.
Min Time	Minimum milliseconds spent in this function and in nested functions.
Total Self Time	The total milliseconds spent in the current function, not including time spent in the subtree of this function.
Max Self Time	The maximum milliseconds spent in the current function, not including time spent in the subtree of this function.
Min Self Time	The minimum milliseconds spent in the current function, not including time spent in the subtree of this function.

Enabling the Script Profiler and Line Profiler

The Script Profiler and the line profiler are not enabled after you install Siebel Tools. You must enable them.

To enable the Script Profiler

- 1 Make sure your Siebel Tools environment is currently connected to the database that a Siebel application uses, and that this application is opened in debug mode.

You can use the Script Profiler only if your Siebel Tools environment is currently connected to the database that a Siebel application uses, and if this application is opened in debug mode.

- 2 Make sure the ST eScript Engine is enabled.

You can use the Script Profiler only if the ST eScript Engine is enabled. For more information, see [“Enabling the ST eScript Engine” on page 127](#).

- 3 Set the debug options on the Debug tab of the Development Tools Options dialog box.

Make sure you complete options in the Run-time Start Up Information section and the Login Information section. For more information, see [“Setting Debug Options” on page 133](#).

- 4 Make sure the Enable Profiler check box on the Debug tab of the Development Tools Options dialog box contains a check mark.

The Enable Profiler option is available only if the ST eScript Engine is enabled. Siebel Tools enables the Script Profiler the next time you use the debug mode in Siebel Tools to open a Siebel application.

- 5 Enable line profiling:

- a Click Set Line Profile Rules.

The Set Line Profile Rules button resides in the Profiler Start Up Options section on the Debug tab of the Development Tools Options dialog box.

- b In the Line Profile Rules dialog box, add a rule using information from the following table, and then Click Add.

Item	Description
Object Type	Choose the object type. Only scriptable objects are available. For more information, see “Object Types You Can Script” on page 120 .
Object Name	Enter the name of an object. You can include the entire object name or use wildcard operators.
Function	Enter the name of a function. You can enter the entire function name or use wildcard operators. For example, you can enter <code>Servi ce_PreI nvokeMethod</code> or <code>Servi ce*</code> .

- c Repeat [Step b](#) for each rule you want to add.

- d Click Ok to exit the Line Profile Rules dialog box.

- 6 Click OK to exit the Development Tools Options dialog box.

The settings for the line profile rule remain active only for the current Siebel Tools session.

To disable line profiling, make sure the Enable Profiler check box on the Debug tab of the Development Tools Options dialog box does not contain a check mark.

Running the Script Profiler

This topic describes how to run the Script Profiler. For more information, see [“Overview of the Script Profiler” on page 139](#).

To run the Script Profiler

1 Log in to Siebel Tools.

2 Make sure the Script Profiler is enabled.

For more information, see [“Enabling the Script Profiler and Line Profiler” on page 141](#).

3 Open the Siebel Script Editor for a scriptable object, and then deliver the changes to the script to a repository.

Make sure you deliver the scripted objects or the projects that these objects reference to the repository that the Siebel client uses. If you do not deliver these objects, then the Script Profiler does not display them. For more information, see [“Object Types You Can Script” on page 120](#) and [“Delivering Workspaces” on page 69](#).

The Script Profiler is now enabled for runtime sessions and you can open the Siebel application from Siebel Tools in debug mode.

4 In Siebel Tools, click the Debug menu, and then click Start.

This step opens the Siebel application in debug mode. If Siebel CRM prompts you for a login, then enter the user name and password, and then click Run. For example, to run a script for the Business Service Simulator, you query for the Service Name and Method Name, and then click Run.

5 In Siebel Tools, click the View menu, Profiler, and then click Call Tree.

Siebel Tools displays the functions that Siebel CRM calls when it runs a script as a hierarchy in the Script Performance Profiler window. It displays the functions that it calls from other functions as child objects. For an example, see [“Example of the Script Performance Profiler Window” on page 139](#).

6 In the Script Performance Profiler window, you can do the following:

- Navigate between nodes to view the different parameters for a function.
- Right-click a node, and then choose a menu item. For more information, see [“Navigating in the Script Performance Profiler Window” on page 144](#).

7 (Optional) To set or reset line profile rules for a function, do one of the following:

- Right-click a function node, and then click Enable Line Profiling or Disable Line Profiling. If you modify a line profile rule, then this modification applies to future calls to the function that this rule references. For more information, see [“Enabling the Script Profiler and Line Profiler” on page 141](#).
- Use the Debug tab in the Development Tools Options dialog box. For more information, see [“Setting Debug Options” on page 133](#).

8 Continue monitoring your script.

For example, you want to test custom script on a business service. In the Siebel client, you can navigate to the Business Service Simulator to run a business service, and then navigate back to the Profiler window to examine the profile data that Siebel CRM logs while the script runs.

Navigating in the Script Performance Profiler Window

You can right-click in the Script Performance Profiler window, and then click one of the following menu items. The menu items that Siebel Tools enables depends on the script. Some of these items might not be available for all scripts:

- **Expand Node.** Expands the function node.
- **Expand All.** Expands the entire tree of a function node.
- **Export.** Exports profile data to a text file that you specify.

You can use Expand All and Export in the Profiler toolbar. For more information about the Profiler toolbar, see [“Example of the Script Performance Profiler Window” on page 139](#).

- **View Source.** Navigates to the current function in the script. If a Script Editor window is not open for this function, then Siebel Tools opens a new editor window and places the cursor at the beginning of this function.
- **Enable Line Profiler.** Enables a function for line profiling. If a function is not in the list of functions chosen for line profiling, then you can click Enable Line Profiler to enable it.
- **Disable Line Profiler.** Disables a function for line profiling.

8

Customizing Objects

This chapter describes how to customize objects. It includes the following topics:

- [Overview of Customizing Objects on page 145](#)
- [Creating, Modifying, Copying, Validating, and Deleting Objects on page 145](#)
- [Examining Objects on page 150](#)

Overview of Customizing Objects

You can edit workspace-enabled repository objects directly without locking any project. Note that only workspace-enabled repository objects can be edited inside workspaces. By default, all workspace-enabled objects are locked at the database level. Therefore, you are not required to lock projects when you edit the objects or perform any type of CRUD (create, read, update, delete) operation.

However, locking a project is still required before you edit the non-workspace objects. The non-workspace objects are Table, Task, Repository, Type, EIM Table, Projects, Dock Objects, Schema Maintenance, and Server Components.

Creating, Modifying, Copying, Validating, and Deleting Objects

This topic describes how to create, modify, copy, or delete an object. It includes the following information:

- [Creating Objects on page 146](#)
- [Modifying Objects on page 147](#)
- [Copying Objects on page 147](#)
- [Validating Objects on page 148](#)
- [Deleting Objects on page 149](#)

This task is a step in [“Roadmap for Setting Up and Using Siebel Tools” on page 16](#).

Creating Objects

You can use a wizard to create an object, or you can create it manually. It is recommended that you use a wizard. For example, to create a new business component, you use the Business Component Wizard. A wizard guides you while you configure a new object. It prompts you for the required property values and automatically configures any required child objects. If a wizard is not available for the object type that you want to create, then you can create it manually in the Object List Editor. For more information about using wizards and creating objects, see *Configuring Siebel Business Applications*.

Using a Wizard to Create Objects

This topic describes how to use a wizard to create an object.

To use a wizard to create objects

- 1 Click the File menu, and then click New Object.
- 2 Choose the appropriate wizard to create the new object.
- 3 Follow the instructions that the wizard shows.

NOTE: The creation of applets and views through a wizard in Siebel Tools Innovation Pack 2017 remains almost the same as in the previous releases. In the previous releases, the web layout editor is opened at the end of the wizard creation. However, starting from Siebel Tools Innovation Pack 2017, after the view and applet are created using wizard, the place holders (item identifiers) that are assigned to the fields are in random order and you must map the fields using the ODH editor. Hence, web layout must be checked in the ODH editor in Web Tools and placeholders must be assigned to fields appropriately.

Using the Object List Editor to Create New Objects

This topic describes how to use the Object List Editor to create a new object. For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).

To use the Object List Editor to create new objects

- 1 In the Object Explorer, click the object type that you want to create.
- 2 Right-click in the Object List Editor, and then click New Record.
- 3 Enter property values in the new row in the Object List Editor.

You must enter values in the Name and Project properties. Siebel Tools might require that you set other properties, depending on the object type.

- 4 To save your modifications, click anywhere outside of the new row.

Modifying Objects

You can use the Object List Editor or the Properties window to modify an object. For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#). For guidelines about when to modify an object or when to create a new object, see *Configuring Siebel Business Applications*.

Using the Properties Window to Modify Objects

This topic describes how to use the Properties window to modify an object.

To use the Properties window to modify an object

- 1 Create or open a workspace.
- 2 In the Object List Editor, locate the object that you want to modify.
For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
- 3 Click the View menu, Windows, and then click Properties.
- 4 In the Properties window, enter a new value for a property.
- 5 To save your modifications, choose another property or click anywhere outside the Properties window.

Siebel Tools saves your modifications and places a check mark in the Changed property.

Renaming Objects

It is recommended that you copy an object, and then rename the copy instead of only renaming the original object. If you rename an object, then Siebel Tools might display an error message that is similar to the following:

Modifying the name of a checked out or locked object causes "unique constraint" error during check-in. To avoid this error, modify the name of the object back to the original name. Do you want to continue?

For more information, see [“Copying Objects” on page 147](#).

Copying Objects

This topic describes how to copy an object. For guidelines about copying an object, see *Configuring Siebel Business Applications*.

To copy an object

- 1 Create or open a workspace.

- 2 In the Object List Editor, locate the object that you want to copy.

For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).

- 3 Click the Edit menu, and then click Copy Record.

Siebel Tools inserts a new record preceding the record that you copied. This new record includes all of the same values as the record that you copied, except the Name property is empty and the Changed property contains a check mark. Siebel Tools also creates a copy of any child objects for the record you copied. For example, if you copy the Account business component, then it creates a new business component that includes several hundred business component fields.

- 4 Enter a new value for the Name property.
- 5 If necessary, modify other properties and child objects.
- 6 To save your modifications, click anywhere outside of the new row.

Validating Objects

The *Validate Tool* is an error correction tool that you can use to make sure each object that you customize does not contain a configuration error. For example, you can use it to make sure an error workflow process does not itself contain an error workflow process. When you validate a workflow process, Siebel Tools displays warnings that describe errors that the workflow process contains. The validation dialog box allows you to correct these errors. This dialog box is without a mode. You can keep it open to view the error messages while you correct the problems that the dialog box reports, or you can proceed with deployment without fixing these validation errors.

Validation uses a set of rules that help make sure your configuration modifications are logically consistent with other objects. If you validate an object, then Siebel Tools also validates the child objects of this object. For example, if you validate the Account business component, then Siebel Tools also validates the child objects of the Account business component, such as fields, joins, and so on.

Using the Object List Editor to Validate Objects

This topic describes how to use the Object List Editor to validate an object.

To use the Object List Editor to validate objects

- 1 In the Object List Editor, locate the object that you want to validate.

For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).

- 2 Click the Tools menu, and then click Validate Object.

Siebel Tools displays the Validate dialog box. For more information, see [“Elements of the Validate Dialog Box” on page 266](#).

- 3 Click Options.

- 4 In the Validation Options dialog box, in the Rules section, choose the validation rules that Siebel Tools must enforce.
For more information, see ["Elements of the Validation Options Dialog Box" on page 267](#).
- 5 (Optional) In the Time Filter section, choose one of the following options:
 - **Last Validated.** Validates objects that you modified since the last time you ran validation.
 - **Custom.** Validates objects that you modified since the date and time that you enter. Enter a date and time to use this option.
- 6 (Optional) In the Action section, choose the following options:
 - **Do Not Report Warnings.** Siebel Tools reports only errors, not warnings. It sets the Enforce field for warnings to No.
 - **Abort Validation After.** You enter a number in the window that this section shows. If Siebel Tools identifies the number of errors that you specify in this window, then it stops validating the object, and then displays the Error dialog box. For example, if you enter 8, then Siebel Tools stops validating after it identifies 8 errors.
- 7 Click OK.
- 8 In the Validate dialog box, click Start.
Siebel Tools validates the object, and then displays any errors that it finds in the Validate dialog box.

Using the Command Line to Validate Objects

You can use the command line to validate objects.

To use the command line to validate objects

- 1 Open a command line, and then navigate to the `SIEBEL_TOOLS_ROOT\BIN` folder.
- 2 Enter the following command:

```
si ebdev.exe /bv
```

The `/bv` switch runs all validation rules for the entire repository.

Deleting Objects

This topic describes how to delete an object.

To delete objects

- 1 In Siebel Tools, create or open a workspace.

NOTE: In case of workspace-enabled objects, you cannot delete any predefined object. If you try to delete the object, it will throw an error indicating that deletion is possible only for object created in the current version and that you have to inactivate the record to proceed further. For more information on making an object inactive, see [“Using the Inactive Property” on page 30](#).

- 2 In case of non-workspace enabled objects, in the Object List Editor, locate the object that you want to delete.

For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).

- 3 Ensure that either the project or the object is locked.
- 4 From the Edit menu, choose Delete Record.
- 5 Check out the project that you examined in [Step 3](#).
- 6 Click the Edit menu, and then click Delete Record.

CAUTION: It is recommended that you do not delete a predefined object. Other objects might reference this predefined object. Instead, you can make this object inactive. For more information, see [“Using the Inactive Property” on page 30](#).

Siebel Tools deletes the object, and it also deletes any child objects of this object. For example, you create a business component named My Account Business Component. If you delete this business component, then Siebel Tools deletes all child objects of this business component, such as fields. It does not delete objects that only reference the object that you delete. For example, if you delete a view, then it does not delete the applets that reference this view.

Examining Objects

This topic describes how to examine objects. It includes the following information:

- [Searching the Repository on page 150](#)
- [Viewing Object Relationships on page 152](#)
- [Comparing and Synchronizing Objects Between Repositories and Archives on page 153](#)
- [Comparing Different Versions of a Workflow Process or Task UI on page 156](#)
- [Determining When Siebel CRM Created or Updated a Record on page 157](#)

Searching the Repository

This topic describes how to locate object definitions for multiple object types. For information about how to locate object definitions for a single object type, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).

If you know that one or more properties includes a value, then you can search the repository across all properties of multiple object types to locate the objects that include this value.

To search the repository

- 1 Click the Tools menu, and then click Search Repository.

Searching the repository can consume a significant amount of time. You can click Cancel in the Search Repository dialog box to cancel a search at any time while the search runs.

- 2 In the Search Repository dialog box, in the Parameters section, enter the search criteria in the Search value window.
- 3 To limit the search to values that are case-sensitive, make sure the Case Sensitive check box contains a check mark.

For example, select this check box to locate objects that include a value of Account and not account.

- 4 To limit the search to values that match the entire search string that you enter in [Step 2](#), make sure the Exact Match check box contains a check mark.
- 5 In the Types to Search list, choose the object type that Siebel Tools must search.

Note the following:

- Siebel Tools searches all object types, by default.
- You can click the object type to choose a single object type.
- You can use multiselect to choose multiple object types.

For more information, see [“Choosing More Than One Record in the Object List Editor” on page 43](#).

- For improved performance, search only the minimum number of object types.
- 6 Click Search Now.

Siebel Tools runs the search and lists the results in the window at the end of the Search Repository dialog box. This window lists all the objects that match the search criteria. It includes the following columns:

Column	Description
Type	Object type of the object that the search returns.
Name	Name of the object that the search returns.
Property	Property name of the object that the search returns.
Value	Property value of the object that the search returns.

- 7 To display the object definition of an item that the search returns, double-click this item in the results window.

Siebel Tools displays the object definition of the item in the Object List Editor.

- 8 To export the search results to a file, click Export.

Viewing Object Relationships

You can use a visualization view to examine how objects relate to each other.

To view object relationships

- 1 Open Siebel Tools.
- 2 Set options for the visualization views:
 - a Click the View menu, and then click Options.
 - b In the Development Tools Options dialog box, click the Visualization tab, and then set the options.

For more information, see [“Development Options for Visualization Views” on page 270](#).

- 3 To open a visualization view, do one of the following:
 - Click the View menu, click Visualize, and then click one of the following menu items:
 - View Details
 - View Relationships
 - View Descendents
 - View Web Hierarchy
 - Right-click an object in the Object List Editor, and then choose a Visualization view.

Table 10 describes the Visualization views.

Table 10. Description of Visualization Views

View	Description
Details	<p>Displays a diagram that illustrates relationships depending on the following object type that you choose in the Object List Editor:</p> <ul style="list-style-type: none"> ■ Business Component. Illustrates the relationship between this business component and tables that this business component references. ■ Business Object. Illustrates the relationship between this business object and the links and business components that this business object references.
Relationships	<p>Displays a diagram that illustrates relationships depending on the following object type that you choose in the Object List Editor:</p> <ul style="list-style-type: none"> ■ Business Component. Illustrates the relationship between this business component and links to other business components. ■ Table. Illustrates the relationship between this table and other tables.

Table 10. Description of Visualization Views

View	Description
Descendents	Displays a dialog box that lists all objects that reference the current object in their Upgrade Ancestor property. For example, if you right-click the Proposal Template Section business component, and then choose View Descendents, then Siebel Tools lists the Proposal Template Section Hierarchy business component. This is the only business component that includes Proposal Template Section in the Upgrade Ancestor property.
Web Hierarchy	<p>Displays a diagram that illustrates a Web hierarchy depending on the following object type that you choose in the Object List Editor:</p> <ul style="list-style-type: none"> ■ Applet ■ Application ■ Business Component ■ Screen ■ View <p>The Web hierarchy displays the parent-child relationships between the object you choose and the parent and child objects of this object throughout the hierarchy.</p>

Comparing and Synchronizing Objects Between Repositories and Archives

You can compare two objects that are of the same object type. Siebel Tools uses color-coded icons to highlight differences. You can choose and copy properties and individual child objects from one object to another object. You can use this feature to propagate a modification that you make to an ancestor object to the descendents of this object or to other objects that are of a similar type. You can assess and adjust differences between objects. You can also compare properties of checked out objects with their counterparts on the Siebel Server. For more information about ancestor objects, see *Configuring Siebel Business Applications*.

You can compare two objects that are of the same type. The Object Comparison dialog box displays a line-by-line comparison between the two objects. You can compare the top-level objects that are defined in the following items:

- Current repository
- Different repositories
- Archive files

NOTE: You can compare and synchronize objects between repositories and archives if the Workspace feature is not enabled (a flattened repository) and the concept of Workspace is not applicable.

For information about top-level object types, see [“Displaying Object Types in the Object Explorer” on page 25](#).

Comparing Two Objects in the Same Repository

You can compare two objects that reside in the same repository.

To compare two objects in the same repository

- 1 In the Object Explorer, click an object type.
- 2 In the Object List Editor, choose two top-level object types.
For more information, see [“Choosing More Than One Record in the Object List Editor”](#) on page 43.
- 3 Click the Tools menu, Compare Objects, and then click Selected.
- 4 Examine the results in the Compare Objects dialog box.
For more information, see [“Elements of the Compare Objects Dialog Box”](#) on page 268.

Comparing Two Objects in Different Repositories

You can compare an object in the current repository to an object in a different repository.

NOTE: The objects that you compare must have the same name in both repositories. A workflow name is appended with the version number of the workflow. Consequently, to compare two workflows in different repositories, the workflows must have the same name and the same version number.

To compare two objects in different repositories

- 1 In the Object Explorer, click an object type.
- 2 In the Object List Editor, choose one top-level object type.
- 3 Click the Tools menu, Compare Objects, and then click Selected vs. Repository.
- 4 In the Open Repository dialog box, choose the repository that includes the object you want to compare, and then click Open.
- 5 Examine the results in the Compare Objects dialog box.

This dialog box displays the following:

- The object in the current working repository in the one window
- The corresponding object in the other repository in the other window

You can update the current working repository or the other repository from the Object Comparison dialog box. To do an update, you must make sure the project that the object references is locked. For more information, see [“Elements of the Compare Objects Dialog Box”](#) on page 268.

Comparing an Object in the Current Repository to an Object in an Archive

You can compare an object in the current repository to an object in an archive.

To compare an object in the current repository to an object in an archive

- 1 In the Object Explorer, click an object type.
- 2 In the Object List Editor, choose one top-level object type.
- 3 Click the Tools menu, Compare Objects, and then click Selected vs. Archive.
- 4 In the Select Archive File to Compare Against dialog box, choose the archive file that includes the object you want to compare.

Siebel Tools saves an archive file as a SIF file. For more information, see [“Overview of Archiving Objects” on page 221](#).

- 5 Click Open.

Siebel Tools starts the comparison at the project level. It does the following:

- If it finds a corresponding object in the archive, then it displays the Compare Objects dialog box.
- If it does not find a corresponding object in the archive, then it does nothing.

For more information, see [“Elements of the Compare Objects Dialog Box” on page 268](#).

Comparing Objects in Two Different Archives

You can compare an object in an archive to an object in another archive.

To compare objects in two different archives

- 1 In the Object Explorer, click an object type.
- 2 In the Object List Editor, choose one top-level object type.
- 3 Click the Tools menu, Compare Objects, and then click Archive vs. Archive.
- 4 In the Select Archive File for Left Side of Comparison dialog box, choose an archive file, and then click Open.
- 5 In the Select Archive File for Right Side of Comparison dialog box, choose an archive file, and then click Open.

Siebel Tools displays the Object Comparison dialog box. It populates with the contents of the archives you choose in [Step 4](#) and [Step 5](#). During the comparison, these archives are read-only. For more information, see [“Elements of the Compare Objects Dialog Box” on page 268](#).

Synchronizing Objects Between Repositories

You can use the Compare Objects dialog box to synchronize objects.

To synchronize objects between repositories

- 1 Lock the projects that the objects that you want to synchronize reference.

- 2 In the Object List Editor, choose two top-level object types of the same object type.
For more information, see [“Choosing More Than One Record in the Object List Editor” on page 43](#).
- 3 Click the Tools menu, click Compare Objects, and then click Selected.
- 4 In the Compare Objects dialog box, choose an object in the First Selection box.
For more information, see [“Elements of the Compare Objects Dialog Box” on page 268](#).
- 5 Click the right arrow button.

Siebel Tools does one of the following, depending on whether or not a corresponding object exists in the repository that the Second Selection section represents:

- **Corresponding object exists.** Siebel Tools modifies the properties of the object in the repository that the Second Selection section represents. It modifies these properties so that they are equal to the object properties of the object that the First Selection section shows.
- **Corresponding object does not exist.** Siebel Tools creates a corresponding object in the repository that the Second Selection section represents.

If you synchronize an object from one repository to another repository, then Siebel Tools synchronizes this object and any child objects that this object includes.

Comparing Different Versions of a Workflow Process or Task UI

Siebel Tools allows you to compare two different versions of the same workflow process or task UI. It uses two separate windows to display these differences.

To compare different versions of a workflow process or task UI

- 1 Log in to Siebel Tools.
- 2 In the Object Explorer, click Workflow Process.
Alternatively, you can compare two versions of a task UI.
- 3 In the Workflow Processes list, choose two versions of the same workflow process.
Make sure you choose two versions of the same workflow process and that these versions include different values in the Version property. Siebel Tools allows you to choose two entirely different workflow processes, but the comparison is not meaningful. You must click Revise to create a new version of the same workflow process.
For more information about Revise, see [“WF/Task Editor Toolbar” on page 263](#). For more information about choosing two workflow versions, see [“Choosing More Than One Record in the Object List Editor” on page 43](#).

- 4 Click the Tools menu, Compare Objects, and then click Selected.

Siebel Tools displays each workflow process version in a separate window. It displays the first record that you choose in [Step 3](#) in the first window and the second record that you choose in [Step 3](#) in the second window. It compares the first record to the second record. It considers the first record as the newer record. It uses the following colors to indicate differences between these records:

- **Yellow.** You modified the object. For example, you added a process property to a business service step.
- **Red.** You deleted an object from the old version.
- **Green.** You added an object to the new version.

Note the following behavior:

- If you click an object that is not red or green in one window, then Siebel Tools chooses the corresponding object in the other window. This functionality works only if a corresponding object exists. For example, if you click a red or green object, then Siebel Tools does not choose any object in the other window because no corresponding object exists.
- You can double-click the object to view details about your modifications.
- You can repeat [Step 3](#) and [Step 4](#) to compare multiple sets of workflows. Siebel Tools displays each set in a separate tab. You can click each tab to navigate between sets.
- To compare the workflow process properties of the two versions, you can double-click the background of either version, and then Siebel Tools displays an object comparison dialog box that allows you to view the differences for process properties instead of the differences between process steps. If a difference exists between the process properties of the two versions, then Siebel Tools sets the background color of one of the versions to yellow.

Determining When Siebel CRM Created or Updated a Record

You can determine the last time Siebel CRM created or updated a record and who made the modification or update.

To determine when Siebel CRM created or updated a record

- 1 In Siebel Tools, create or open a workspace.
- 2 In the Object List Editor, locate the object you want to modify.
For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
- 3 Click the Help menu, and then click About Record.
- 4 In The Siebel Tools dialog box, click Details to display more information about the record.

9

Managing Repositories

This chapter describes how to manage repositories. It includes the following topics:

- [Viewing, Renaming, Comparing, and Configuring Repositories on page 159](#)
- [Exporting and Importing Repositories on page 169](#)
- [Upgrading Repositories on page 173](#)
- [Process of Migrating Repositories on page 176](#)
- [Deleting Repositories on page 186](#)

Viewing, Renaming, Comparing, and Configuring Repositories

This topic describes how to view, rename, and compare repositories. It includes the following information:

- [Overview of Managing Repositories on page 159](#)
- [Viewing Information About the Current Repository on page 160](#)
- [Renaming Repositories on page 162](#)
- [Configuring Siebel CRM to Read Data from a Single Repository on page 163](#)

Overview of Managing Repositories

A *Siebel Repository* is a set of tables that includes Siebel objects and server scripts. These objects and scripts define a Siebel application, such as Siebel Call Center. This repository provides the metadata that Siebel CRM requires to interact with enterprise data and to interact with the people who use the Siebel client. The Siebel application stores these tables in the Siebel runtime repository, and then reads the repository at run time. You can use Siebel Tools to view and modify the information that this repository contains. For more information, see *Configuring Siebel Business Applications*.

CAUTION: You must use only one repository in a production environment, and you must synchronize the compiled repository and repository table data. If you configure Siebel CRM to use multiple Siebel repositories, then unpredictable behavior might result.

Siebel CRM includes *Incremental Repository Merge*, which is a feature that allows you to merge multiple repositories to a single repository during an incremental upgrade. It automatically does some of this upgrade work for you, such as importing SIF files and seed data, applying schema modifications, and deliver. For more information, see *Siebel Database Upgrade Guide*.

Files That You Use to Manage Repositories

A *Siebel Archive File* (SIF) is a type of file that you can use to import object definitions into or export objects from the Siebel runtime repository. You can use Siebel Tools to create an SIF file, and you can use SIF files to move your modifications from a source environment to a destination environment. Siebel Tools writes SIF files in XML format. A SIF file uses the following hierarchy, and it can include property values and scripts:

- 1 Repository
- 2 Project
- 3 Object
- 4 Child Objects

For example, the following code is part of an SIF file that Siebel Tools creates when it adds objects to a hotfix. It includes a definition for the Account Assoc Applet:

```
<REPOSITORY
  NAME="Siebel Repository"
  ... >
  <PROJECT
    ...
    NAME="Account (SSE)"... >
    <APPLET
      ASSOCIATE_APPLET="Account Assoc Applet"
      BUSINESS_COMPONENT="Account"
      CLASS="CSSFrameListBase"
      ...
      NAME="Account List Applet"
      ... >
      <APPLET_METHOD_MENU_ITEM... >
      </APPLET_METHOD_MENU_ITEM>...
    </APPLET>
    <BUSINESS_COMPONENT
      CACHE_DATA="N"
      CLASS="CSSBusComp"
      ... >
    </BUSINESS_COMPONENT>
    ...
  </PROJECT>
</REPOSITORY>
```

For more information about:

- Using Siebel Tools to import or export an SIF file as an archive file, see [Chapter 12, "Archiving Objects."](#)

Viewing Information About the Current Repository

In Siebel Tools, only one repository is available for all users working on workspaces in the Siebel environment. Siebel Tools opens this repository, by default.

In some situations, multiple repositories might reside on the Siebel Server (for example, if you upgrade to a new version of Siebel CRM).

To view information about the current repository

- 1 In Siebel Tools, click the File menu, and then click Open Repository.
 Siebel Tools displays the Open Repository dialog box. This dialog box lists the repositories in the database that Siebel Tools uses. Siebel Tools highlights the repository that it is currently using.
- 2 Click the Help menu, and then click About SRF.
- 3 Examine the information that the About Repository File dialog box shows using the information from the following table.

Field	Description
Internal Version	Version number that Oracle maintains. Siebel CRM modifies this value only if the internal format of the repository is modified (for example, during an upgrade).
User Version	Reserved for use by Oracle’s Siebel Anywhere. It maintains this number when Oracle creates kits that upgrade the repository. Siebel CRM reads this value when it does a version check.
Full Compile	Choose this option to display information about the most recent full compile.
Last Incremental Compile	Choose this option to display information about the most recent incremental compile. If no incremental compile has occurred since the last full compile, then this option is not available.
When	Date of the last compile.
Machine Name	Name of the computer that Siebel CRM uses to compile the repository.
Language	Language code.
User Name	The Microsoft Windows logon name of the user who compiled the repository.
Repository	Repository name of the repository that was current when Siebel CRM performed the compile. This value is typically Siebel Repository.
Tools Version	The version number and build number of the Siebel Tools software that performed the compile. Global Customer Support can use this information to help you resolve a problem.
Schema Version	Database schema version of the Siebel database that Siebel CRM uses to compile the repository.

Renaming Repositories

It is recommended that you use Siebel Repository as the name of the repository that you use in your production environment. If you want to rename the repository, then it is recommended that you use the procedure that this topic describes.

To rename repositories

- 1 Make sure all developers check in all projects that are currently checked out from the repository that you want to rename.
- 2 Log in to Siebel Tools, connected to the server database.
- 3 Display the Repository object type.
For more information, see [“Displaying Object Types in the Object Explorer” on page 25](#).
- 4 In the Object Explorer, click Repository.
- 5 In the Repositories list, locate the repository that you want to rename.
For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
- 6 Enter the new name in the Name property, and then step off the record to save your modifications.
For more information, see [“Guidelines for Naming Repositories” on page 162](#).
- 7 Deliver your modifications.
For more information on how to deliver your workspaces, see [“Delivering Workspaces” on page 69](#).
- 8 Communicate the name of the new repository to your developers and request that each developer get all projects from the server repository.
- 9 Modify the value of the Siebel Repository enterprise parameter to the new name of the repository.
For information about how to modify an enterprise parameter, see *Siebel System Administration Guide*.
- 10 Modify the Application Main Repository Name parameter in the Object Manager.

Guidelines for Naming Repositories

If you name a repository, then it is recommended that you use the following guidelines:

- Use a naming convention for all repositories that your Siebel CRM implementation uses. Siebel Servers reference a repository by name. The procedure that you use to upgrade to a new version of Siebel CRM depends on the repository name. A consistent naming convention promotes successful configuration and testing and minimizes the work required to migrate a new repository or to do an upgrade.
- Use the default Siebel Repository name, where possible. You can modify this name only if it is absolutely necessary. The default configuration that Siebel CRM uses assumes that the repository name is Siebel Repository.

- Use the same repository name in your test environment that you use in your production environment. Using the same name simplifies the process of migrating a repository from development to test and from test to production. It also eliminates the need for you to modify the client configuration or server configuration during a migration. For more information, see [“Process of Migrating Repositories” on page 176](#).
- If your development environment uses multiple repositories, then use a unique and descriptive name for each repository. For example, you might use Siebel v8.2 Original as the name of the repository when you install Siebel CRM. You can use another descriptive name for the repository that you use during an upgrade and for a repository from a prior custom configuration.

Configuring Siebel CRM to Read Data from a Single Repository

The example in this topic describes how to configure Siebel CRM to read data from a single repository when the Siebel database contains multiple repositories.

To configure Siebel CRM to read data from a single repository

- 1 Log in to the Siebel client with administrative privileges, and complete the following steps:
 - a Navigate to the Administration - Order Management screen, and then the Message Types view.
 - b Navigate to the Payload view, and then create a new record.
 - c In the new record, access the drop-down list for the Response Field.
If the Response Field drop-down list:
 - ❑ **Does not display duplicate values.** The Siebel database does not contain multiple repositories, and you can exit this task.
 - ❑ **Displays duplicate values.** The Siebel database contains multiple repositories, and you must configure Siebel CRM to read data only from a single repository. Continue to [Step 2](#).
 - d Delete the new record that you created.
- 2 Configure Siebel CRM to read data only from a single repository:
 - a Log in to Siebel Tools.
 - b In the Object Explorer, click Pick List.
 - c In the Pick Lists list, query the Name property for the following value:
UMS Pi ckLi st Response Fi el d

- d Modify the Search Specification property using the value from the following table.

Property	Value
Search Specification	[Buscomp] = 'UMS Response' and [Repository Id] = RepositoryId()

Siebel CRM displays the UMS Type Variables List Applet in [Step 2](#). This applet references the UMS Type Variable business component. It displays the Response Field Name field and uses the UMS PickList Response Field picklist for this field. This picklist uses the following search specification when it references the UMS Pick List Field business component:

[Buscomp] = ' UMS Response'

The UMS Pick List Field business component references the S_FIELD table. It does not include a search specification, so Siebel CRM does not filter the records that it gets according to the specifications that the repository contains. If the Siebel database contains multiple repositories, then Siebel CRM gets all the field names from all repositories, and then displays them in the picklist. This configuration might result in Siebel CRM displaying duplicate values that you cannot choose in the picklist. To correct this situation, you create a search specification that configures Siebel CRM to read data only from the current repository.

- 3 Deliver your modifications.
- 4 Repeat [Step 1](#) to make sure the Response Field drop-down list does not display duplicate values.

Tagging Objects When Using Siebel Remote

This topic describes how to configure your environment so that you can use object tagging with Siebel Remote. Siebel Tools does not synchronize any modifications that you make when you use Siebel Remote. It communicates only with the repository on the Siebel Server through project Check Out, project Check In, or a Get. You do not use Siebel Tools to synchronize the modifications that you make through object tagging in a Siebel Remote client. Instead, you must use the synchronization features that Siebel Remote provides. For more information, see *Siebel Remote and Replication Manager Administration Guide*.

To tag objects when using Siebel Remote

- 1 Make sure every developer who uses the Siebel Remote client uses the MOBILE CLIENT - STANDARD routing model to synchronize client changes with the server database.

Developers must not use the MOBILE CLIENT - EXTRACT ONLY routing model because it only extracts the local database. It does not synchronize the local database. For more information about how to use these routing models and Siebel Remote, *Siebel Remote and Replication Manager Administration Guide*.
- 2 Make sure the tools.cfg configuration file and the Local ODBC data source that the Siebel Web Client uses each reference the same database file, such as sse_data.dbf.
- 3 Make sure each developer synchronizes any modifications that the developer makes through object tagging with the Siebel Server, and that each developer periodically synchronizes their local database with the Siebel Server so that their client includes the object tagging modifications that other developers make.

Restricting the Objects That Developers Can Modify

In some situations, it might be helpful to restrict the objects that developers can modify, according to a date that you specify. If you enable this feature, then Siebel Tools applies these restrictions when the developer does the following work:

- Import or export an SIF or SDF file. For more information, see [“Files That You Use to Manage Repositories” on page 160](#).
- Directly modify the Siebel Database.

To restrict the objects that developers can modify

- 1 Enable the system preferences:
 - a Log in to Siebel Tools, connected to the Siebel Server.

You must make sure that you log in to Siebel Tools, connected to the Siebel Server. If the values that you set for the system preferences in this procedure do not match between the Siebel Server and the local computer, then Siebel Tools does not allow you to check in your modifications.

- b Click the Screens menu, click System Administration, and then click System Preferences.
- NOTE:** In Web Tools, click the Tools menu and then click the System Preferences menu item.
- c In the System Preferences list, set values for the following system preferences.

System Preference Name	System Preference Value
EnablePerfFieldModification	Set the value to TRUE. This system preference applies edit and copy restrictions to objects.
Enable Object Version Control	Set the value to TRUE.
ServerEditRecordTimeStamp	Enter the time when Siebel Tools must start using this configuration. For more information, see “Setting the Server Edit Record Time Stamp Parameter” on page 166.

- 2 Log out of Siebel Tools.

Setting the Server Edit Record Time Stamp Parameter

When you set the ServerEditRecordTimeStamp parameter, you enter the time when Siebel Tools must start using this configuration. Use the following format:

MM/DD/YYYY

You can specify only the month, day, and year. You cannot specify minutes and seconds.

For example, to start using the configuration on October 9, 2014, enter the following value:

10/09/2014

In this example, Siebel Tools does the following:

- Allows you to modify any object that includes a timestamp that occurs after October 9, 2014.
- Does not allow you to modify any object that includes a timestamp that occurs before October 9, 2014. Developers can view or copy these objects, but not modify them.
- Allows you to create new records, and then to modify these new records.
- Allows you to copy any record, regardless of the timestamp.

If you set a property value in the source record, if this property affects database performance, and if you connect Siebel Tools to the:

- **Siebel Server**, then Siebel Tools resets the property value in the copied record, and you cannot reset this value.
- **Local database**, then you cannot check in this record.

For more information, see [“Properties That Affect Database Performance”](#) on page 167.

Restricting Access to an Object

This topic describes how to restrict access to an object.

To restrict access to an object

- 1 Determine the date that Siebel Tools most recently updated the object:
 - a In the Object List Editor, choose the record where you want to restrict access.
 - b Click Help, and then click About Record.
 - c In the Siebel Tools dialog box, in the Updated section, note the date that the On window shows.
- 2 Set the value for ServerEditRecordTimeStamp to the date that you noted in [Step c](#).

Properties That Affect Database Performance

[Table 11](#) lists the properties that affect database performance.

Table 11. Properties That Affect Database Performance

Object Type	Property
business component	The following properties affect database performance: <ul style="list-style-type: none"> ■ Force Active
field	The following properties affect database performance: <ul style="list-style-type: none"> ■ Force Active ■ Link Specification ■ Immediate Post Changes
multi value link	The following properties affect database performance: <ul style="list-style-type: none"> ■ Check No Match
multi value field	The following properties affect database performance: <ul style="list-style-type: none"> ■ Force Active ■ Link Specification
single value field	The following properties affect database performance: <ul style="list-style-type: none"> ■ Force Active ■ Link Specification

Overriding Restrictions That Prevent Developers from Modifying Objects

This topic describes how to log in to Siebel Tools so that the developer can modify any object, regardless of the timestamp restrictions that you place in [Step 1 on page 165](#).

To override restrictions that prevent developers from modifying objects

- 1 In Microsoft Windows, click Start, All Programs, and then click the Siebel Tools installation folder.
- 2 Right-click the Siebel Tools icon, and then click Properties.
- 3 In the Siebel Tools Properties window, click the Shortcut tab, add the EditPerfFields switch to the Target field, and then click OK.

For example:

```
C: \Siebel \8.1\Tools_2\BIN\si ebdev. exe /c  
"C: \Siebel \8.1\Tools_2\bin\enu\tool s. cfg" /u SADMI N /p SADMI N /d Sample /  
Edi tPerfFI el ds
```

Bold font indicates the switch that you add.

- 4 Log in to Siebel Tools.

Adding Identification Numbers to Repository Modifications

In some development environments, you might find it useful to associate an identification number with each modification that a developer makes to the repository. You can use this feature to help manage and maintain development work, particularly when your organization includes multiple developers who work across multiple development initiatives. For example, you can use this identification number to track all the modifications that your development team makes for a feature, or to help track the resolution of a bug that uses a unique identification number.

To add identification numbers to repository modifications

- 1 Enable the system preferences. Do [Step 1 on page 165](#).
- 2 Log out of, and then log back into Siebel Tools.
- 3 Modify an existing object, or create a new one.
- 4 Check in the project or object:
 - a Right-click the object that you modified in [Step 3](#), and then click Check In.
 - b In the Check In dialog box, click Check In.
 - c In the Repository Object Check-in Description dialog box, in the Enter Feature No/Bug No/RF No field, enter a number.

You can specify any alphanumeric number up to a maximum length of 15 characters. Siebel Tools assigns this number to all objects that you check in.

- 5 (Optional) Locate all the modifications that are associated with an identification number:
 - a Start SQLPlus, connected to the Siebel Database on the Siebel Server.
 - b Run the following SQL query:


```
SELECT column REL_FEATURE_TXT FROM S_TAG_OBJECT WHERE
REL_FEATURE_TXT = ' i d e n t i f i c a t i o n _ n u m b e r '
```

where:

- *identification_number* is the number that you entered in [Step 4](#). You must enclose this number with single quotes.
- S_TAG_OBJECT is the Siebel CRM table that contains all of the identification numbers.
- REL_FEATURE_TXT is the column of the S_TAG_OBJECT table that contains the identification numbers.

For example, if you specify an identification number of myObjects123 in [Step 4](#), then you run the following SQL query:

```
SELECT column REL_FEATURE_TXT FROM S_TAG_OBJECT
WHERE REL_FEATURE_TXT = 'myObj e c t s 1 2 3'
```

Siebel Tools adds the identification number to the REL_FEATURE_TXT column the first time that you check in the object. For each subsequent checkin of this object, it saves a unique number in the same column. To do this, it appends the subsequent identification number to the existing identification number. It uses two vertical bars (||) to separate these numbers.

To locate all identification numbers, run the following SQL query:

```
SELECT * FROM S_TAG_OBJECT
```

You must run this query against the Siebel database on the Siebel Server.

For more information about SQLPlus, see the SQL*Plus® User's Guide and Reference on the Oracle Help Center at http://docs.oracle.com/cd/B19306_01/server.102/b14357/toc.htm.

Exporting and Importing Repositories

This topic describes how to export and import repositories. It includes the following information:

- [Overview of Exporting and Importing Repositories on page 169](#)
- [Importing Repositories in Windows Environments on page 171](#)
- [Importing Repositories in UNIX Environments on page 172](#)
- [Exporting Repositories on page 173](#)
- [Importing or Exporting Repositories at a Later Time on page 173](#)

Overview of Exporting and Importing Repositories

You can use the Database Configuration Wizard to export or import a repository. You can use this utility to do the following work:

- Back up or restore a repository.

- Move all repository objects to another environment that uses the same physical database schema that the source environment uses.
- Export objects to an archive so that you can export or import only some objects. For more information, see [“Exporting Objects to an Archive” on page 222](#).

For more information about how to use the Database Configuration Wizard, see *Siebel Database Upgrade Guide* and the *Siebel Installation Guide* for the operating system you are using.

Guidelines for Exporting and Importing Repositories

If you use the Database Configuration Wizard to import or export a repository, then it is recommended that you use the following guidelines:

- If you import a custom repository, then the Database Configuration Wizard restores all the languages that are part of the predefined repository when you do the import. For example, if you archive repositories weekly, and if your development repository includes ENU and DEU, then the wizard includes ENU and DEU when it imports one of the archived repositories. For more information, see [“About Predefined Objects” on page 23](#).
- If you modify the repository, then make sure you perform Full Publish. Make a backup copy of the repository file in case you want to compare it to the contents of the updated repository. If necessary, you can use this comparison to verify that the import is identical to the backup.
- You can use the database utilities that your RDBMS vendor provides to back up the entire contents of the Siebel database.
- If you customize the source repository, then you can use the Migrate option of the Database Configuration Wizard. For more information, see [“Process of Migrating Repositories” on page 176](#).

Character Encoding That Siebel CRM Supports to Import or Export Repositories

[Table 13](#) describes the character encoding that Siebel CRM supports when it imports or exports a repository. These databases must use the same Siebel version.

Table 13. Character Encoding That Siebel CRM Supports to Import or Export a Repository

Source Database	Target Database
Code Page	Code Page
Unicode	Unicode
Code Page	Unicode

Where the Database Configuration Wizard Saves Log Files

If you export a repository in a Windows or UNIX environment, and if you use the Export Repository option of the Database Configuration Wizard, then this wizard saves log files in following directories:

- `SI EBSRVR_ROOT\log\exprep\output`

- `SI EBSRVR_ROOT\log\exprep\state`

Exprep is the default process name for the exprep utility. You can modify this value.

Importing Repositories in Windows Environments

This topic describes how to import repositories in a Windows environment.

To import repositories in Windows environments

- 1 Make sure Siebel CRM supports the databases that you intend to use.
For more information, see [“Character Encoding That Siebel CRM Supports to Import or Export Repositories”](#) on page 170.
- 2 In Microsoft Windows, navigate to Start Programs menu, Settings, Control Panel, and then click Services.
- 3 Stop all Siebel Servers.
For more information, see *Siebel System Administration Guide* and the *Siebel Installation Guide* for the operating system you are using.
- 4 Click the Start Programs menu, Programs, Siebel Enterprise Server Configuration 8.0, and then click Database Server Configuration.
- 5 In the Database Configuration Wizard, enter information when this wizard prompts you, and then click Next to continue.
- 6 Choose Import Repository when the wizard prompts you to specify a database operation.
For important caution information, see [“Caution About Migrating a Repository”](#) on page 180.
- 7 Specify the following items:
 - To import your custom 8.x repository
 - The location of where the custom CustRep.dat file resides
- 8 When the wizard displays the Configuration is Complete window, choose one of the following options, and then click Next:
 - **Yes Apply Configuration Changes Now.** The wizard saves the configuration information that you entered, and you can open the Siebel Upgrade Wizard in [Step 10](#).
 - **No I Will Apply Configuration Changes Later.** The wizard saves the configuration information that you entered, but you cannot open the Siebel Upgrade Wizard in [Step 10](#).
- 9 In the Configuration Parameter Review window, review the configuration that you entered. To modify a value, click Back to return to the window that includes the parameter you want to modify, modify the parameter, and then click Next.
- 10 When the wizard prompts you to run the configuration, click one of the following:
 - **No.** The wizard does not save the configuration information that you entered. You must enter the database configuration parameters again.

- **Yes.** The wizard saves the configuration information that you entered.

11 Click OK.

The Database Configuration Wizard does one of the following, depending on the choice that you make in [Step 8](#):

- **Apply now.** Opens the Siebel Upgrade Wizard, and then calls the SQL generator to create the SQL scripts.
- **Apply later.** Saves the configuration information, but does not open the Siebel Upgrade Wizard. You can restart the configuration, and then run the Upgrade Wizard later. For more information, see [“Importing or Exporting Repositories at a Later Time” on page 173](#).

Importing Repositories in UNIX Environments

This topic describes how to import repositories in a UNIX environment.

To import repositories in UNIX environments

- 1 Do [Step 1 on page 171](#) and [Step 3 on page 171](#).
- 2 Make sure \$SIEBEL_ROOT is the current folder.
- 3 Run a script, depending on the following UNIX shell that you are using:
 - **Korn shell.** Run siebenv.sh.
 - **C shell.** Run siebenv.csh.
- 4 Make sure that the following environment variables use the following values:
 - **SIEBEL_ROOT.** This path must end in siebsrvr (for example, /usr/siebel/siebsrvr).
 - **LANGUAGE.** Determines the language that the Database Configuration Wizard uses. The value of this variable is a text string that identifies the language (for example, enu for English).
- 5 Run the following command to start the Database Configuration Wizard:

```
$i nstal l _path/confi g/confi g -mode dbsrvr
```
- 6 In the Database Configuration Wizard, do the following:
 - Enter information when the wizard prompts you.
 - Use the Next and Back button to navigate between dialog boxes.
 - Choose Import Repository when the wizard prompts you to choose a database operation.
 - Specify to import your 8.x repository.
 - Identify the location of where the custom CustRep.dat file resides.
- 7 After you enter all the requested information, the wizard displays a message that is similar to the following. Click Next to continue:

Configuration is complete: configuration parameters will be saved to *\$Masterfile* file when the wizard completes. Please run the following command line after you exit from this configuration wizard. This command will deploy the process you configured to the database.

```
$SIEBEL_ROOT/siebsvr/bin/srvrupgwi z /m $SIEBEL_ROOT/siebsvr/bin/$Masterfile
```

- 8 The wizard displays the values that you entered in the Parameter Review window. To modify a value, click Back to return to the appropriate window.
- 9 The wizard prompts you to click one of the following values:
 - **Yes.** The wizard saves the configuration in a master file in the *\$SIEBEL_ROOT/bin* folder. It does not start the Upgrade Wizard. For information about how to start the Upgrade Wizard, see *Siebel Database Upgrade Guide*.
 - **No.** The wizard does not save the configuration that you entered.

Exporting Repositories

To export a repository when you use:

- **Microsoft Windows.** Use the same procedure that you use to import a repository, but choose Export Repository in [Step 6 on page 171](#).
- **UNIX.** Use the same procedure that you use to import a repository, but choose Export Repository in [Step 6 on page 172](#).

Importing or Exporting Repositories at a Later Time

If you use the Database Configuration Wizard to export or import a repository, then it saves the values that you specify to the *master_exprep.ucf* file in the *SI EBSRVR_ROOT* folder. After the wizard finishes collecting information from you, it prompts you to export or to not export. If you choose not to export or not to import, then you can run the following command to do the export or import at a later time:

```
siebug.exe /m master_exprep.ucfs
```

Upgrading Repositories

The *Siebel Application Upgrader* is a utility that you can use to get new features from the latest software release while preserving the custom configuration that you created in the current repository. It allows you to do the following:

- Compare your custom configuration to the modifications that a new Siebel CRM release contains.
- Receive notifications about conflicts between the customizations that you make and the new release.
- Merge any differences between objects.

- Choose the modifications to apply and manually override modifications.
- Merge objects with versions, including task UI objects and workflow processes. It copies version 1 through version n from the prior custom repository to the new custom repository. It merges version 0 from the prior repository with the new custom repository. This configuration results in version $n + 1$ in the new custom repository.
- Reduce the time required to upgrade Siebel CRM.

You can use the Application Upgrader to merge an entire custom repository with another repository. To merge only part of a repository, you must import the repository. For more information, see [“Exporting and Importing Repositories” on page 169](#). For more information about the Application Upgrader, see *Siebel Database Upgrade Guide*.

To upgrade repositories

- 1 Log in to Siebel Tools.
- 2 Click the Tools menu, Upgrade, and then click Upgrade Application.
- 3 In the Merge Repositories dialog box, choose the repositories to merge, and then click Merge.

Siebel Tools does the following:

- Starts the upgrade.
- Displays any differences between objects and properties.
- Displays any differences between objects and properties for different versions of task UIs and workflow processes.

Identifying Conflicts That Occur During Upgrades

This topic describes how to identify merge conflicts between objects that Siebel Tools added or modified during a repository merge. A *merge conflict* is a scenario in which a customer changed an object in the current customer repository, and Siebel CRM changed that same object to a different value in a new version of the Siebel repository. Consequently, the attribute for the object is different when the following three repositories are compared: the current customer repository, the current Siebel repository, and the new version of the Siebel repository.

To identify conflicts that occur during upgrades

- 1 Make sure you finish upgrading repositories.
For more information, see [“Upgrading Repositories” on page 173](#).
- 2 In Siebel Tools, click the Screens menu, Application Upgrader, and then click Application Upgrade Object List.
- 3 In the Application Upgrades list, right-click the record of the merge that you want to analyze, and then click Hierarchy Reports.

Siebel Tools displays the hierarchy report, which includes the objects that Siebel Tools added or modified during a merge. The report includes the following types of objects:

- **Objects that include a valid value for each field value.** Siebel Tools modified these objects during the merge. An N or a Y in a binary field is an example of a valid value.

For these objects, the Status field designates added objects or objects with modified attributes. For added objects, an N appears in the Attributes field because none of the attributes are modified. For objects with modified attributes, a Y appears in the Attributes field, and the modified attributes appear in the Attributes pane.

- **Objects that include an asterisk (*) for each field value.** Siebel Tools modified children of these objects during the merge, or Siebel Tools modified dependent objects of these objects during the merge.

4 To view information about merge details, complete the following steps:

- a Expand and navigate through the hierarchical tree in the Object Types pane to select a parent or child object type.

The starting List of Objects pane shows the objects for the parent object type that you select, and the next List of Objects pane shows the objects for the child object type that you select.

- b Click an object in a List of Objects pane.

In the Attributes pane, Siebel Tools displays any merge conflicts for the object.

5 (Optional) To filter the objects in the report, select one of the following values in the Filter drop-down list, and then click Go:

- **Siebel and Customer Modified.** Siebel Tools displays only objects that Oracle or you modified. These objects have a Y in the In Prior Standard, In Prior Customized, and In New Standard fields.
- **Siebel Modified.** Siebel Tools displays only objects that Oracle modified or added. These objects have an N in the In Prior Standard and In Prior Customized field and a Y in the In New Standard field.
- **All.** Siebel Tools displays all modified or added objects.

6 (Optional) To filter the merge conflicts in the Attributes pane, select or clear the Critical Only check box as follows, and then click Go:

- Select the check box to show only critical merge conflicts.
- Clear the check box to show all merge conflicts.

7 (Optional) To display the dependencies for an object, complete the following steps:

- a Click an object in a List of Objects pane.

If you want to display the dependencies for an additional object, then hold down the CTRL key, and click the additional object in the List of Objects pane.

- b Click Show Dependencies.

If the object has no dependencies, then the Show Dependencies button is disabled.

Siebel Tools displays the dependent objects that it modified and the attributes for these objects.

- c Click Back to return to the hierarchy report.

Process of Migrating Repositories

To migrate a repository, do the following tasks:

- 1 [Preparing to Migrate Repositories on page 176](#)
- 2 [Using the Database Configuration Wizard to Migrate Repositories on page 177](#)
- 3 (Optional) [Updating Siebel Remote Databases on page 180](#)
- 4 [Migrating Non-repository Configurations and Data on page 181](#)

This topic also includes the following information:

- [Using the Repository Import and Export Utility on page 183](#)
- [Guidelines for Migrating Repositories on page 185](#)

It is sometimes necessary to migrate the Siebel Repository and any application customizations between databases before you deploy modifications to a development, test, or production environment. Doing this migration makes sure that the database schema for the user data, the business objects, and the user interface remain synchronized.

Preparing to Migrate Repositories

In this topic, you prepare to migrate the repository.

This task is a step in ["Process of Migrating Repositories" on page 176](#).

To prepare to migrate repositories

- 1 Make sure you complete testing.
You must verify that any customizations you make work correctly, and that they meet your business requirements.
- 2 Make sure the target database configuration meets the database requirements.
For more information, see the *Siebel Installation Guide* for the operating system you are using and *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.
NOTE: For Siebel CRM product releases 8.1.1.9 and later and for 8.2.2.2 and later, the system requirements and supported platform certifications are available from the Certifications tab on My Oracle Support. For information about Certifications, see article 1492194.1 (Article ID) on My Oracle Support.
- 3 Check in all projects in the source database and in the target database.
If a project is checked out, and if you migrate a database, then the migration works but Siebel CRM does not lock the project in the target database.
- 4 If your environment includes Remote users, then make sure all these users do a full synchronization.
- 5 Do a full backup of the target database.

- 6 Delete old repositories from the target database.

For more information, see [“Deleting Repositories” on page 186](#).

- 7 Update database statistics, if necessary.

- 8 Export the source repository to a file.

It is recommended that you use the Database Configuration Wizard to export the source repository to a file. You use this file later during a migration. For more information, see [“Exporting Repositories” on page 173](#).

You can also migrate directly from the source database. If you do this, then you do not need to export the source repository to a file.

- 9 Stop all Siebel Server tasks and disconnect all database access.

If you do not need to minimize downtime during a migration, then it is recommended that you stop all Siebel Server tasks and disconnect all database access until the migration finishes. This downtime occurs in the Siebel Enterprise where the target repository resides.

Using the Database Configuration Wizard to Migrate Repositories

In this topic, you use the Database Configuration Wizard to migrate repositories.

This task is a step in [“Process of Migrating Repositories” on page 176](#).

To use the Database Configuration Wizard to migrate repositories

- 1 Make sure all users, including the database administrator, disconnect from the Siebel application.
- 2 Start the Database Configuration Wizard according to the following operating system that you are using:

- **Windows.** Do [Step 1 on page 171](#) through [Step 5 on page 171](#).

- **UNIX.** Do [Step 1 on page 172](#) through [Step 5 on page 172](#).

- 3 In the Database Configuration Wizard, click **Migrate Repository** when the wizard prompts you to choose a database operation, and then click **Next**.

For important caution information, see [“Caution About Migrating a Repository” on page 180](#).

- 4 Choose to get the source repository from one of the following items, and then click **Next**:

- **Get from a file.** If you choose this option, then the wizard displays the **Repository File Selection** window. Enter the path and the file name of the source repository file that you created in [Step 8 on page 177](#), and then click **Next**.

- **Get from the database.**

- 5 Specify whether or not the target enterprise is online or offline, and then click **Next**:

- **Online.**

- Offline.

If you choose the online option, then Siebel CRM keeps the Siebel Enterprise where the target repository resides online during the migration until it must shut down this environment. When it must do this shut down, it displays a dialog box that informs you that it will shut down the target environment during the remainder of the migration. This option allows you to minimize the amount of downtime during a migration. If you run this migration unattended, then do not choose this online option. Running this migration from a script is an example of running it unattended.

6 Choose one of the options, and then click Next:

- **There are new schema changes to be applied.** Choose this option if the source repository includes any physical or logical schema modifications. Examples of schema modifications include adding an extension column to an existing table, or adding a new extension table. If you are not sure whether or not any schema modifications exist, then it is recommended that you choose this option.
- **There are no new schema changes.** Choose this option if the source repository does not include any physical or logical schema modifications. If you choose this option, then the Database Configuration Wizard requires less time to finish the migration.

7 If in [Step 4 on page 177](#) you choose to get the source repository:

- **From a file.** The wizard displays the Target Repository Name window. Enter the target repository name, and then click Next.
- **From the database.** The wizard displays the series of dialog boxes that require you to define the source database. Use information from the following table to complete each of these dialog boxes.

Dialog Box	Description
ODBC Data Source Name	Enter the ODBC data source name.
Database User Name	Enter the user name and password for the source database.
Database Table Owner	Enter the table owner and table owner password for the source database.
Source Database Repository Name	Enter the source database repository name and the target database repository name.

- 8 Complete the information in the following windows, and then click Next.

The sequence that the wizard uses to display these windows and the information that you enter might vary slightly depending on whether or not your environment uses Windows or UNIX, or the type of database that your environment uses. The wizard does not display any windows for Microsoft SQL Server.

Window	Description
RDBMS Platform	For UNIX only. Choose an RDBMS platform.
Target Database Encoding	Choose whether or not the target database is Unicode. Consult with your DBA regarding how to set up the target database.
Target Database Repository Name	Enter the repository name of the target database.
Target RDBMS Platform	Choose the target RDBMS platform.
Target Database ODBC Datasource	Enter the ODBC data source for the target database.
Target Database User Name	Enter the user name and password that the server components that reside on the target database use.
Target Database Table Owner	Enter the table owner and the table owner password for the target database.
Index Table Space Name	If you choose IBM DB2 for the target database platform, then the wizard displays the Index Table Space Name dialog box. Enter the index tablespace name and 4KB tablespace name.
16K Table Space Name	If you choose IBM DB2 for the target database platform, then the wizard displays the 16K Table Space Name dialog box. Enter the 16KB tablespace name and the 32KB tablespace name.
Index Table Space Name	If you choose Oracle Database for the target database platform, then the wizard displays the Index Table Space Name dialog box.

- 9 If the wizard displays a Language Selection window, then choose the appropriate language, and then click Next.

The wizard migrates strings from the repository that uses the language that you choose. If you migrate a repository in a multilingual environment, and if the source repository includes duplicate strings in multiple languages, then the wizard copies all of these strings to the target repository.

- 10 Finish the migration, depending on the following operating system that your environment uses:
- **Windows.** Do [Step 8 on page 171](#) through [Step 11 on page 172](#).
 - **UNIX.** Do [Step 7 on page 172](#) through [Step 9 on page 173](#).

11 Wait for the wizard to finish the migration.

The Database Configuration Wizard does the following work:

- Exports the source repository from a file. You can also export the repository for the database.
- Imports the repository into the target database.
- If you choose to apply new schema modifications in [Step 6](#), then the wizard does the following work:
 - Exports the logical schema definition from the source repository to a control file.
 - Synchronizes the physical schema of the target database with this logical schema definition.

Caution About Migrating a Repository

If you migrate a custom repository, then you must use the Migrate Repository option in the Database Configuration Wizard.

CAUTION: If you migrate a custom repository and schema from one environment to another environment, such as migrating from a development environment to a test environment, then do not use the Export/Import option in the Database Configuration Wizard. Instead, you must use the Migrate Repository option.

This Migrate Repository option calls the Repository Migration Utility (dev2prod). If you use the Repository Migration Utility directly instead of calling it from the Database Configuration Wizard, then you must reset the Locked and Allow Object Locking columns after you finish the migration.

Updating Siebel Remote Databases

If your environment includes Remote users, then you must update Siebel Remote databases. For more information about performing the steps in this topic, see *Siebel Remote and Replication Manager Administration Guide*.

This task is a step in [“Process of Migrating Repositories” on page 176](#).

To update Siebel Remote databases

- 1 Stop and restart Siebel Remote server components.
- 2 Wait for the Transaction Processor (TxnProc) and the Transaction Router (TxnRoute) server components to finish processing all pending transactions.
- 3 Regenerate the local database templates.

Use the Generate New Database server component (GenNewDb) to regenerate the local database template file. This step updates this template file so that it uses the same schema and version that the server database uses.

- 4 Reextract the local database for all Remote users.

If you do not use Siebel Anywhere to update Remote clients, then you must use the Database Extract (DbXtract) server component to extract the local database for all remote users.

If you do not extract these databases, then Remote users can synchronize, but Siebel CRM does not create any error message even though these databases might not include the migration modifications. This configuration allows users who use Siebel Anywhere to upgrade their Remote databases and continue working.

Using Siebel Anywhere to Update Remote Databases

You can use Siebel Anywhere to update some Remote databases. Importing a repository and then synchronizing the schema definition in a target environment is equivalent to migrating the repository. The Remote users must synchronize the next time they log in to their local database after the migration finishes. Synchronizing downloads new schema modifications from the Siebel Server to the local database that the Remote user uses. Before synchronization occurs, you must rename the older repository to a temporary name, and rename the imported repository to the correct name. If a Remote user does not synchronize, then the data that the local database contains does not match the data that the server database contains. For more information, see *Siebel Anywhere Administration Guide*.

Migrating Non-repository Configurations and Data

The Database Configuration Wizard does not migrate non-repository configurations and data. This topic describes how to migrate this information.

This task is a step in [“Process of Migrating Repositories” on page 176](#).

To migrate non-repository configurations and data

- 1 Migrate Web templates, image files, and style sheets.

For more information, see [“Migrating Image Files and Style Sheets” on page 182](#).

- 2 Migrate any customizations that you have made to the Siebel application or to runtime data.

You must migrate any customizations that your Siebel Enterprise data contains, such as custom views, responsibilities, assignment rules, and so on. For more information, see *Siebel Application Deployment Manager Guide* and *Siebel Assignment Manager Administration Guide*.

- 3 Migrate custom configurations, such as parameter values and server component definitions.

For more information, see [“Migrating Custom Configurations” on page 182](#).

- 4 Migrate database triggers.

Siebel CRM does not migrate custom database triggers. If your Siebel base tables include custom database triggers, then you must disable them before you migrate the repository, and then recreate them on the target repository after the migration finishes.

- 5 Update database statistics.

Migrating Image Files and Style Sheets

A Siebel Server installation includes the following files that determine how Siebel CRM displays the Siebel client in a Web browser:

- Image files (GIF, JPG, and PNG files)
- Cascading style sheet files (CSS files)

If you modify any of these files in the source environment, then you must copy them to the target environment. You can use Siebel Application Deployment Manager (ADM) to automate this work. For more information, see *Siebel Application Deployment Manager Guide*.

For information about how to specify the Siebel Enterprise Security Token and how to migrate content from the Siebel Server to the Web server, see the *Siebel Installation Guide* for the operating system you are using and *Siebel Security Guide*.

To migrate image files and style sheets

- Copy any new or modified files from the source Siebel Server to the target Siebel Server, and use values from the following table to determine where to copy files.

File Type	File Location
Images	<i>SI EBSRVR_ROOT</i> \webmaster\images\ <i>language_code</i>
Cascading style sheets	<i>SI EBSRVR_ROOT</i> \webmaster\files\ <i>language_code</i>

Migrating Custom Configurations

If you migrate a repository, then you must also migrate any custom environment configurations in the source environment to the target environment (for example, custom server component definitions). Siebel CRM stores this information in the Siebel Gateway Registry. The Database Configuration Wizard does not migrate this information.

To migrate custom configurations

- 1 Recreate any custom environment configurations.

Use the Server Manager command-line interface (svrmgr) and scripts that provide input to svrmgr that automate this work in the target environment. It is recommended that you use scripts because you can save them and reuse them for future migrations. If you reuse a script, then make sure you edit it if the target environment is a new Siebel version (for example, if a Siebel upgrade includes a new Siebel version). New parameters or svrmgr commands might be available with this new version. For more information, see *Siebel System Administration Guide*.

You can also use the Server Manager GUI or command-line interface (svrmgr) to manually create these custom configurations in the target environment. For information, see *Siebel System Administration Guide*.

- 2 Recreate parameter values.

For more information, see *Siebel System Administration Guide*.

Using the Repository Import and Export Utility

The Repository Import and Export Utility imports or exports a repository, or creates a file dump of a repository. It can also import an INTL table. INTL tables contain language-specific information and are a part of the repository. You use this utility only if your migration requires special parameters settings that a batch file cannot access, or if you must create a file dump. In all other situations, you use the Database Configuration Wizard. For information “[Using the Database Configuration Wizard to Migrate Repositories](#)” on page 177.

To use the Repository Import and Export Utility

- 1 Open a Windows command line.
- 2 To import a repository, enter the following command:

```
repi mexp /A I /G language_codes
```

where:

- *A I* specifies to import data from a file.
- *language_codes* is a list of codes.

- 3 To export a repository, enter the following command:

```
repi mexp /A E parameters
```

where:

- *A E* specifies to export data to a file.
- *parameters* specify the parameters.

- 4 To display a complete list of command-line usage options for this utility, enter the following command:

```
repi mexp
```

You can save this output to a text file.

For more information, see “[Parameters You Can Use with the Repository Import and Export Utility](#)” on page 184.

Parameters You Can Use with the Repository Import and Export Utility

Table 14 describes the parameters that you can use with the Repository Import and Export Utility.

Table 14. Parameters You Can Use with the Repository Import and Export Utility

Parameter	Description
<p>You can use one of the following values:</p> <ul style="list-style-type: none"> ■ /a e ■ /a i ■ /a x ■ /a o ■ /a s ■ /a p ■ /a t 	<p>Required. You can use one of the following parameters:</p> <ul style="list-style-type: none"> ■ a e. Specifies to export all data from the Siebel Repository tables to a file. ■ a i. Specifies to import data from a file into the Siebel Repository tables. ■ a x. Specifies to import locale-specific repository data from a file into the INTL tables. If you use a x, then you must also use the /g option. ■ a o. Specifies to export Runtime Repository Tables and required base tables during initial, full, and incremental migration. ■ a s. Specifies to skip import of Runtime Repository tables only. ■ a p. Specifies to import Runtime Repository Tables and required base tables during initial, full, and incremental migration. ■ a t. Specifies to delete a given repository
/u <i>user_name</i>	Required. Specifies the Siebel administrator user name.
/p <i>password</i>	Required. Specifies the password for the Siebel administrator user name.
/c <i>ODBC_data_source</i>	Required. Specifies the ODBC data source. The value in the SIEBEL_DATA_SOURCE environment variable is the default value. If this value includes a space, then you must enclose the value in quotes.
/d <i>table_owner</i>	Required. Specifies the Siebel Database table owner. The value of the SIEBEL_TABLE_OWNER environment variable is the default value.
/g <i>language_codes</i>	<p>For import only.</p> <p>Required only if you use the /a x parameter to import to INTL tables.</p> <p>Specifies to import a repository that contains locale-specific data for one or more languages (for example, FRA, DEU, ITA). You can specify ALL to import all the languages that the repository file contains.</p> <p>To import locale objects, you must use the /g parameter, and you must specify at least one language code. If you do not do this, then the utility does not import any locale objects, and the Siebel client does not include any text after you compile the imported repository.</p> <p>If you export repository data to a file, then this utility includes locale-specific data for all languages.</p>

Table 14. Parameters You Can Use with the Repository Import and Export Utility

Parameter	Description
<i>/r repository_name</i>	Required. Specifies the name of the repository that you are importing or exporting. The default value is Siebel Repository. If this value includes a space, then you must enclose the value in quotes.
<i>/f data_file</i>	Required. Specifies the data file for the repository, including the path to this file. Specify the file to which you are exporting, or from which you are importing, repository data. The path must not include spaces.
<i>/y <watermark string></i>	Specifies watermark information needed for migration.
You can use one of the following values: <ul style="list-style-type: none"> ■ <i>/v y</i> ■ <i>/v n</i> 	Specifies to verify data: <ul style="list-style-type: none"> ■ The default value for export is n. ■ The default value for import is y.
You can use one of the following values: <ul style="list-style-type: none"> ■ <i>/n 0</i> ■ <i>/n 1</i> ■ <i>/n 2</i> 	For export only. Specifies to modify information about creating and updating records. You can use one of the following values: <ul style="list-style-type: none"> ■ n 0. Make no modifications. ■ n 1. Update the CREATED_BY, UPDATED_BY, and OWNER_BRANCH columns. The default value is 1. ■ n 2. Update the CREATED_BY, UPDATED_BY, and OWNER_BRANCH columns, and update the date columns.

Guidelines for Migrating Repositories

If you migrate a repository, then it is recommended that you use the following guidelines:

- Isolate development and test environments from the production environment.
- Do not migrate a repository between two databases that use different releases or patch levels. To avoid an inconsistent environment, do not migrate repositories between different versions of Siebel applications.
- If your migration requires special parameter settings that batch files cannot access, or if you must do a file dump, then use the Repository Import and Export Utility. For more information, see [“Using the Repository Import and Export Utility” on page 183](#).
- Use a consistent naming convention for the source repository and for the target repository. You can rename the target repository to indicate that you replaced it during a migration. For more information, see [“Guidelines for Naming Repositories” on page 162](#).

- Do not modify ODBC parameters or settings for the data sources that you created when you configured the Siebel Server when you installed Siebel CRM. A repository migration references one ODBC data source for the source database, and another ODBC data source for the target database. This requirement applies to all servers and databases. For more information about how to verify the ODBC data source, see the *Siebel Installation Guide* for the operating system you are using.
- Consider the time required to export over a WAN. It is more efficient to export the file on the source environment, and then copy the exported file to the target environment. If you migrate a repository over a wide area network (WAN), and if you run the Database Configuration Wizard from the target environment, then Siebel CRM exports only the source repository to a flat file. All other processing occurs on the local area network (LAN) of the target environment.
- If you defined a custom tablespace on IBM DB2 for z/OS, then this tablespace might affect the Database Configuration Wizard. For more information, see *Implementing Siebel Business Applications on DB2 for z/OS*.

Deleting Repositories

You can delete repositories either using the repimexp utility or the RRCleanup utility. You use repimexp utility by using the option /a t to delete the existing repository.

You use the RRCleanup utility to delete the existing repository or the orphan records from the Siebel database. Using this utility, you can delete:

- Orphan records.

This option does not require you to pass a repository name as an argument to the RRCleanup utility because orphan records do not have the valid repository in Siebel system and the repository does not exist in the S_REPOSITORY table.

- Repository data.

This option passes a valid repository as an argument to the RRCleanup utility to delete all repository data from all repository tables.

- Only runtime repository data.

This option passes a valid repository as an argument to the RRCleanup utility to delete all runtime repository data from all run-time tables.

In Siebel Tools, the RRCleanup utility is located at this location:

```
<Siebel_Home>\BIN\RRCleanup.exe
```

In the Siebel Server, the RRCleanup utility is located at this location:

```
<Siebel_Home>\ses\si_ebsrvr\BIN\RRCleanup.exe
```

To run the RRCleanup utility

- 1 Open the Command Prompt window from your computer by clicking the Start button and then selecting the Run option.
The Run window appears.
- 2 Enter the value CMD in the Open field.
The Command Prompt window appears.
- 3 Change the directory in the Command Prompt window using one of the following commands:

```
cd <Siebel _Home>\BIN\RRCleanup.exe
```

 or

```
cd <Siebel _Home>\ses\si esrvr\BIN\RRCleanup.exe
```
- 4 The Command Prompt window displays the following arguments and parameters, which you can use to run the RRCleanup utility:
 - -t Siebel Table Owner (required)
 - -u Username (required)
 - -p Password (required)
 - -o ODBC Data Source (required)
 - -l Log File Name (default: RRCleanup.log)
 - -r Repository Name (delete Runtime Repository data for repository name pass. e.g. "Siebel Repository")
 - -a Orphan Flag (Pass Y if required to delete only orphan records)
 - -s Server Tool (Pass that contains odbcsql.exe usually <tools>\bin)
 - -b RepositoryType ((R)unTime, (D)esignTime) (required)
- 5 Use the following command to delete a particular repository data from repository tables.

```
RRCleanup.exe -t "Table Owner" -u "Siebel User Name" -p "Siebel User Name Password" -r "Repository Name" -o "ODBC Data source" -s "Path that contains odbcsql command" -b "Repository Type"
```
- 6 View the output stages that represent the progress of the tasks performed by the utility to ensure that the RRCleanup utility is run successfully.

10 Localizing Strings and Locale Data

This chapter describes how to localize strings and locale data. It includes the following topics:

- [Configuring Symbolic Strings on page 189](#)
- [Converting Symbolic Strings on page 197](#)
- [Configuring Locale Data on page 205](#)
- [Using the Locale Management Utility on page 209](#)

Configuring Symbolic Strings

This topic describes how to configure symbolic strings. It includes the following information:

- [Overview of Configuring Symbolic Strings on page 189](#)
- [Modifying the Configuration File to Support Symbolic Strings on page 191](#)
- [Creating a Project for Your Symbolic Strings on page 191](#)
- [Modifying a Predefined Symbolic String on page 192](#)
- [Creating a New Symbolic String on page 192](#)
- [Modifying a Symbolic String to Globally Update Display Values on page 193](#)
- [Setting a Symbolic String Reference on page 194](#)
- [Entering a String Override on page 196](#)

Overview of Configuring Symbolic Strings

A *symbolic string* is an object that you can use to store the value of a string. It allows you to define a string one time, and then reference it from multiple objects. An object can reference this symbolic string to get a literal value, and then display it as text in the Siebel client. For example, an applet can reference a symbolic string to get the text it shows in a control caption, a list column display name, or an applet title.

A *translatable string* is a type of string that Siebel CRM can translate to another language. For example, it can translate the text string that defines a control label from English to French.

The symbolic string centralizes all the strings in the repository. It includes strings in English and all other languages. It includes the following advantages:

- Reduces redundancy because many objects can reference one symbolic string
- Results in a more consistent user interface
- Simplifies maintenance because you are required to maintain only one string for a word or phrase

- Eliminates duplicate translations of the same word
- Reduces translation costs

Siebel CRM cannot translate a nontranslatable property to another language. For example, the HTML Sequence property, HTML Height property, and HTML Width property are nontranslatable properties.

If you deliver to the repository, then Siebel Tools uses the current language mode that you set for Siebel Tools. It uses this language mode to choose from one of several translations for a set of symbolic string locale records. For more information on how to deliver your workspaces, see [“Delivering Workspaces” on page 83](#).

Symbolic Strings in the Object Hierarchy

The Symbolic String object type is a top-level object type. It includes the child Symbolic String Locale object type. Each symbolic string represents a word or phrase that is language independent. For example:

Account

Siebel CRM stores all translations of this word or phrase, including English, as a symbolic string locale. For information about top-level object types, see [“Displaying Object Types in the Object Explorer” on page 25](#).

Siebel Tools stores data for a symbolic string in the S_SYM_STR (symbolic strings) table. It stores data for a symbolic string locale in the S_SYM_STR_INTL (Repository Symbolic String Locale) table. Objects that reference symbolic strings store foreign key references to these strings in the S_SYM_STR table.

Symbolic strings do not include other types of strings that Siebel CRM typically includes as seed data, such as LOVs, error messages, or predefined queries. For more information about localizing these types of strings, see [“Fixing Orphaned String References After an Upgrade” on page 208](#).

How Siebel Tools Determines a Translatable String

Siebel Tools does the following to compile an object property that displays a translatable string, such as the Title property of an applet:

- If a value exists in the string language that the compile uses, then it compiles this string override value.
- If the string override field for the current language mode that you set in Siebel Tools does not include a string value, then Siebel Tools uses the current language mode and the String Value property of the associated symbolic string locale to determine the value.

In most situations, a string override does not exist. For more information, see [“Entering a String Override” on page 196](#).

How Some Early Releases Store Translatable Strings

Some early releases of Siebel Tools store translatable strings in the locale objects of a parent object type. For example, each applet includes a set of child locale records that define the text for the applet title that Siebel CRM shows in the Siebel client.

Modifying the Configuration File to Support Symbolic Strings

This topic describes how to modify the Siebel Tools configuration file to support symbolic strings.

To modify the configuration file to support symbolic strings

- 1 Open the tools.cfg file.
- 2 In the Siebel section, set the EnableToolsConstrain parameter to one of the following values:
 - **TRUE.** Constrained mode. Siebel Tools requires you to choose a translatable string from the list of available string references. You cannot override the string reference. For example, you cannot enter a value for a string override field. You cannot create a new symbolic string reference.
 - **FALSE.** Unconstrained mode. Siebel Tools does not require you to choose a translatable string from the list of string references. You can override the string reference. For example, you can enter a value in a string override field. You can create a new symbolic string reference.
- 3 (Optional) Set the SymStrPrefix parameter to a custom value.

To distinguish between a predefined symbolic string and a custom symbolic string that you create, Siebel Tools sets the prefix for any new symbolic string you create to X_, by default. You can modify this value. For example, if you set this parameter to SymStrPrefix=X_NewString, then any new symbolic string you create includes the X_NewString prefix. For more information, see [“Modifying a Predefined Symbolic String” on page 192](#) and [“About Predefined Objects” on page 23](#).

Creating a Project for Your Symbolic Strings

The Symbolic String project is very large. Checking in or checking out the entire project might consume a significant amount of time. It is recommended that you do not check out the entire Symbolic String project. Instead, you can create a new project and set the Project property for any new symbolic string you create to reference this project.

To create a project for your symbolic strings

- 1 Log in to Siebel Tools.
- 2 In the Object Explorer, click Project.
- 3 In the Projects list, create a new project.
For example, create a new project named CompanyXYZ New Symbolic String.
- 4 Make sure the Locked property contains a check mark.

Modifying a Predefined Symbolic String

You can modify a *predefined symbolic string*, which is a string that comes predefined with Siebel CRM. It includes a SBL_ prefix in the Name property. For more information, see [“About Predefined Objects” on page 23](#).

CAUTION: It is recommended that you do not modify a predefined symbolic string. Instead, it is recommended that you create a new symbolic string that includes the text you require. It is recommended that you do not modify the display value of a predefined symbolic string. Siebel CRM might use this display value throughout the Siebel client. For a monolingual deployment, you risk modifying text in the Siebel client that you do not intend to modify. For a multilingual deployment, you risk breaking the relationships between display values for different languages. For more information, see [“Creating a New Symbolic String” on page 192](#).

To modify a predefined symbolic string

- 1 Log in to Siebel Tools and display the Symbolic String object type.
For more information, see [“Displaying Object Types in the Object Explorer” on page 25](#).
- 2 In the Object Explorer, click Symbolic String.
- 3 In the Object List Editor, locate the object you want to modify.
For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
- 4 Set the Project property of the string you locate in [Step 3](#) to the project you create in [“Creating a Project for Your Symbolic Strings” on page 191](#).
To modify multiple records, see [“Using the Properties Window to Modify Objects” on page 147](#).

Creating a New Symbolic String

You create a new symbolic string.

To create a new symbolic string

- 1 Make sure the EnableToolsConstrain parameter allows you to create a new symbolic string.
For more information, see [“Modifying the Configuration File to Support Symbolic Strings” on page 191](#).
- 2 Log in to Siebel Tools and display the Symbolic String object type.
For more information, see [“Displaying Object Types in the Object Explorer” on page 25](#).
- 3 In the Object Explorer, click Symbolic String.

- 4 In the Symbolic Strings list, create a new record using the values from the following table.

Property	Description
Name	Enter a unique name for this symbolic string. For more information about the prefix that Siebel Tools adds to the name of any custom symbolic string you create, see "Modifying the Configuration File to Support Symbolic Strings" on page 191 .
String Value	Enter the text that this symbolic string shows. In some situations, this value might be a value that Siebel Tools determines according to the current language mode and the String Value property of the child symbolic string locale object. For more information, see "Setting the Language Mode" on page 18 . Siebel Tools truncates any trailing spaces you enter, including full width spaces in Japanese.
Definition	Enter text that describes the purpose of this symbolic string.
Project	Set this property to the project you create in "Creating a Project for Your Symbolic Strings" on page 191 .

Modifying a Symbolic String to Globally Update Display Values

You can configure Siebel CRM to make global modifications to the values it shows in the Siebel client. For example, this feature can be useful in the following situations:

- You want to modify all instances of the word Account that Siebel CRM shows in the Siebel client to Customer.
- You want to deploy Siebel CRM specific to an industry in a locale other than English. The text strings that it shows in the Siebel client might not be appropriate for this industry.

To modify a symbolic string to globally update display values

- 1 Set the language mode.
For more information, see ["Enabling Language Override" on page 19](#).
- 2 In the Object Explorer, click Symbolic String.
- 3 In the Object List Editor, locate the object you want to modify.
For more information, see ["Locating and Modifying Object Definitions in the Object List Editor" on page 23](#).
- 4 Modify the value of the String Value property.
- 5 Deliver the projects that include the objects that reference this symbolic string.

Setting a Symbolic String Reference

A *symbolic string reference* is a reference to a symbolic string. It allows you to set the text that Siebel CRM shows in the Siebel client for an object property, such as an applet title or application display name. This topic describes two different ways that you can set this text.

Using the String Reference Property to Set a Symbolic String Reference

This topic describes how to use the String Reference property to set a symbolic string reference.

To use the String Reference property to set a symbolic string reference

- 1 Make sure the EnableToolsConstrain parameter is set to FALSE.
For more information, see [“Modifying the Configuration File to Support Symbolic Strings” on page 191](#).
- 2 In the Object List Editor, locate the object you want to modify.
For example, locate the Account List Applet. For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
- 3 In the Title - String Reference property, click the drop-down arrow.
This property varies according to object type. For more information, see [“Properties That Store a Symbolic String” on page 194](#).
- 4 In the Title - String Reference dialog box, locate the string reference you require, and then click Pick.
Siebel Tools enters a value in the Title property according to the current language mode and the value in the String Value property of the child symbolic string locale.
- 5 If no existing symbolic string meets your requirements, then you can enter a string override.
For more information, see [“Entering a String Override” on page 196](#).

Properties That Store a Symbolic String

[Table 15](#) describes the properties that store a symbolic string. The property varies according to object type. Siebel Tools displays these properties only in the Object List Editor. It does not display them in the Properties window.

Table 15. Properties That Store a Symbolic String

Object Type	Label for the String Reference Property
Web Page	Title - String Reference
Applet	Title - String Reference
Screen	Viewbar Text - String Reference

Table 15. Properties That Store a Symbolic String

Object Type	Label for the String Reference Property
View	A view includes the following properties that reference a symbolic string: <ul style="list-style-type: none"> ■ Title - String Reference ■ Thread Title - String Reference
Application	Display Name - String Reference

Entering a Value to Set a Symbolic String Reference

This topic describes how to enter a value to set a symbolic string reference.

To enter a value to set a symbolic string reference

- 1 Make sure the EnableToolsConstrain parameter is set to FALSE.
For more information, see [“Modifying the Configuration File to Support Symbolic Strings” on page 191](#).
- 2 In the Object List Editor, locate the object you want to modify.
For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
For example, locate the Account List Applet.
- 3 In the Title property, enter the text that Siebel CRM must show in the Siebel client.
This property varies according to object type. For more information, see [“Properties That Store the Display Text” on page 195](#).
- 4 Tab out of the field.
Siebel Tools searches for a string reference that includes a value in the String Value property that matches the value you enter, and then does one of the following depending on the result of the search:
 - **A unique match exists.** Siebel Tools enters the symbolic string reference it finds into the String Reference property. It enters this value according to the current language mode and the value that the String Value property of the child symbolic string locale contains.
 - **Multiple matches exist or no match exists.** Siebel Tools displays an error dialog box. You can click OK to close the dialog box, and then use the String Reference Property. For more information, see [“Using the String Reference Property to Set a Symbolic String Reference” on page 194](#).

Properties That Store the Display Text

[Table 16](#) describes the properties that store the display text. The property varies according to object type. Siebel Tools displays these properties in the Object List Editor and in the Properties window.

Table 16. Properties That Store the Display Text

Object Type	Property That Stores the Display Text
Web Page	Title
Applet	Title
Screen	Viewbar Text
View	A view includes the following properties that include display text: <ul style="list-style-type: none"> ■ Title ■ Thread Title
Application	Display Name

Entering a String Override

If you cannot find a symbolic string that meets your requirements, then you can override the symbolic string that Siebel CRM uses. [Table 16 on page 196](#) lists the properties that store a translatable string. Siebel CRM includes a corresponding String Override property for each of these properties that store a translatable string. For example, the Title – String Override property is the corresponding property for the Title property of an applet. You can use this override property to enter a custom string.

To enter a string override

- 1 Make sure the EnableToolsConstrain parameter is set to FALSE.
For more information, see [“Modifying the Configuration File to Support Symbolic Strings” on page 191](#).
- 2 In the Object List Editor, locate the object you want to modify.
For example, locate the Account List Applet. For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).
- 3 In the Title-String Override property, enter the string you require, and then tab out of this field.
This property varies according to object type. For more information, see [“Properties That Store the String Override” on page 197](#).
For more information, see [“How Siebel Tools Stores the Override Value You Enter” on page 197](#).

How Siebel Tools Stores the Override Value You Enter

If you modify the value in the Title-String Override property, then Siebel Tools also modifies the value in the Title property to the custom value you enter. It stores this value in a locale object that is a child of the parent you modify. For example, if you use the Title-String Override property to modify the display name for the Account List Applet to My Big Accounts, then Siebel Tools adds a child locale object to the Account List Applet. The Title property of this locale object includes the custom value you enter, such as My Big Accounts. To view this locale object, you must display it in the Object Explorer. For more information, see [“Displaying Object Types in the Object Explorer” on page 25](#).

To set this value, Siebel Tools uses the current language mode and stores it as a language-dependent value. This value does not affect other references to symbolic strings.

Properties That Store the String Override

[Table 17](#) describes the properties that store the string override. This property varies according to object type. Siebel Tools displays these properties in the Object List Editor and in the Properties window.

Table 17. Properties That Store the String Override

Object Type	Property That Stores the String Override
Web Page	Title - String Override
Applet	Title - String Override
Screen	Viewbar Text - String Override
View	A view includes the following properties that store a string override: <ul style="list-style-type: none"> ■ Title - String Override ■ Thread Title - String Override
Application	Display Name - String Override

Converting Symbolic Strings

This topic describes how to convert symbolic strings. It includes the following information:

- [Consolidating Symbolic Strings on page 202](#)
- [Using a Batch File to Convert Strings on page 204](#)

You can use Siebel Tools to convert a translatable string to a symbolic string. Siebel CRM stores this translatable string as a child locale record of a top-level object type. It stores symbolic strings in a single table. Converting a translatable string might be useful in the following situations:

- You upgrade to a new Siebel CRM version and you want to convert custom translatable strings to symbolic strings.
- You use string overrides to store text strings and periodically want to convert them to symbolic strings.

Converting to symbolic strings reduces the size of the repository, simplifies translations, and helps to make sure that the text that Siebel CRM shows in the Siebel client is consistent. It also requires that development work is finished.

To convert symbolic strings

- 1 Review the caution information described in [“Caution About Converting and Consolidating Symbolic Strings” on page 200](#).
- 2 Prepare to do the conversion:
 - a Back up the Siebel database and repository.
 - b Verify the Siebel Tools configuration file:
 - Make sure the DataSource parameter references the correct Siebel database. The conversion utility uses this Siebel database.
 - Make sure the EnableToolsConstrain parameter is set to FALSE and that the SymStrPrefix parameter is set to the appropriate prefix.For more information, see [“Modifying the Configuration File to Support Symbolic Strings” on page 191](#).
 - c Make sure all projects are unlocked.
 - d Inform other developers not to log on to the development environment while the conversion runs.
- 3 Log in to Siebel Tools and display the Attribute object type.
For more information, see [“Displaying Object Types in the Object Explorer” on page 25](#).
- 4 Identify the object types you want to convert:
 - a In the Object Explorer, click the Flat Tab, and then click Attribute.
 - b In the Attributes list, query the Name property for the following string:
String Reference

The Parent Type property displays the complete set of object types to convert. If an object type includes more than one property that references a symbolic string, then you can run the conversion for this object type only one time.
- 5 Convert locale strings to symbolic strings. You must convert the locale strings for the object types you identify in [Step 4](#). You can use the conversion utility to do this conversion:
 - a Open a command line, and navigate to the `SIEBEL_TOOLS_ROOT\BIN` folder.
 - b Enter the following command:

```
consol eapp config_file app_lang user_Id password "business_service"
"business_service_method: parameters"
```

Use the values from the following table. For more information, see [“Running the Console Application Executable” on page 200](#).

Parameter	Description
config_file	Specify the Siebel Tools configuration file, such as tools.cfg. Use the default data source. For more information, see “Modifying the Configuration File to Support Symbolic Strings” on page 191 .
app_lang	Specify the application language, such as ENU.
user_Id	Specify the user Id.
password	Specify the password.
business_service	Specify the String Conversion business service.
business_service_method	Specify the name of the business service method.
parameters	Specify the input parameters to the business service method. For more information, see “Separating Conversion Files into Smaller Files” on page 201 .

- Export candidates for one of the object types you identified in [Step 4](#). Enter the following command:

```
consol eapp "ConversionExport: Filename=name.txt, Repository=repository_name,
Object=object_type, LogFile=object_typeExport.log, Language=language_code, MatchMin=an_ineger"
```

For example, the following command exports the control object type:

```
consol eapp "ConversionExport: Filename=Control.txt, Repository=Siebel Repository,
ObjFile=Control, LogFile=ControlExport.log, Language=ENU, MatchMin=1"
```

For information about setting parameters in this command, see [“Parameters You Use with the Conversion Export Utility” on page 274](#).

For more information, see [“How the Conversion Export Utility Converts Strings” on page 201](#).

- Repeat [Step 6](#) for each object type that you identified in [Step 4](#).
- Import the symbolic strings that you converted in [Step 6](#). Enter the following command:

```
consol eapp "ConversionImport: Filename=name.txt, Repository=repository_name,
LogFile=name.log, UnlockProjects=false, SkipParentUpdates=true, Project=Symbolic
Strings"
```

For example:

```
consol eapp "ConversionImport: Filename=Control.txt, Repository=Siebel  
Repository, LogFile=ConversionImport.log, UnlockProjects=false, SkipParentUpdates=  
true, Project=Symbolic Strings"
```

For information about setting parameters in this command, see [“Parameters That You Can Use with the Conversion Import Utility” on page 275](#).

For more information, see [“How the Conversion Import Utility Converts Strings” on page 201](#).

9 Consolidate strings.

For more information, see [“Consolidating Symbolic Strings” on page 202](#).

Caution About Converting and Consolidating Symbolic Strings

Converting or consolidating symbolic strings might consume computer processing resources.

CAUTION: Converting or consolidating symbolic strings might consume a significant amount of computer processing resources. For information about the requirements for computer processing speed, see [Siebel System Requirements and Supported Platforms on Oracle Technology Network](#).

NOTE: For Siebel CRM product releases 8.1.1.9 and later and for 8.2.2.2 and later, the system requirements and supported platform certifications are available from the Certifications tab on My Oracle Support. For information about Certifications, see article 1492194.1 (Article ID) on My Oracle Support.

File and Object command-line parameters are case sensitive.

CAUTION: File and Object command-line parameters that you use when you convert or consolidate a symbolic string are case sensitive. Other command-line parameters are not case sensitive.

Running the Console Application Executable

You can use `consol eapp.exe` (console application executable) to run various utilities that you use to manage symbolic strings.

To run the console application executable

1 On the computer where you installed Siebel Tools, navigate to the `SIEBEL_TOOLS_ROOT\BIN` folder.

2 Enter the following command:

```
consol eapp parameter_1 parameter_n
```

This command runs the console application executable.

Separating Conversion Files into Smaller Files

The command that this topic describes separates an export file that the conversion utility creates into smaller, more manageable files according to object type. For example, the Siebel runtime repository might include up to 130,000 controls. To improve performance, you can simultaneously import multiple consolidation files of the same object type or of different object types. An average desktop PC can typically run only 10 simultaneous conversion import processes.

To separate conversion files into smaller files

- Add the following parameter to the command you use in [Step 5 on page 198](#):

```
"Spl i tFi l e: Fi l e n a m e=C o n t r o l . t x t , L i n e s=2000"
```

Use the values from the following table.

Parameter	Description
Filename	Name of the export file.
Lines	Approximate number of lines in each file. The conversion utility does not separate a set of symbolic strings, so the number of lines might not equal the value you set in this parameter.

How the Conversion Export Utility Converts Strings

The Conversion Export utility can convert any object that can include a translatable string (for example, a control, list column, or applet). The export utility does the following work:

- 1 Creates a sorted list of English (ENU) child records for each translatable string in an object.
- 2 Sequentially processes each object that includes multiple translatable strings. For example, a list column that includes a display name and prompt text includes multiple translatable strings.
- 3 Uses the list it creates in [Step 1](#) to create information about any new symbolic strings.
- 4 For records that include identical ENU translations, it compares the non-ENU records and reuses the same symbolic string for subsequent records, if possible.
- 5 Repeats [Step 1](#) through [Step 4](#) for the next object type.
- 6 Produces an output file that includes information about the new symbolic strings, including information about each language translation and replacement strings. This file is for information purposes. It is not a log file. It is not necessary to review the contents of this file.

How the Conversion Import Utility Converts Strings

To convert the records that use the new symbolic strings, you can use a utility that performs inserts, updates, and deletes on the Siebel database. For input, this utility uses the output file that the conversion export creates. This utility does the following:

- 1 Creates a new symbolic string record and child symbolic string locale records for each string according to the string values in the target objects. The export file includes information about each string, including a unique name and information about each child locale object.

- 2 Adds a reference to the new symbolic string in the relevant property of each original object. For example, 10 applets exist and the title for each of these applets is My Big Service Requests. The non-ENU values for these titles all use the same value. In this example, the export file includes information about one new symbolic string and instructions for each of the 10 applets to use this new symbolic string as the title. The conversion import does the following:
 - Creates a symbolic string with an ENU value of My Big Service Requests.
 - Sets the Title - String Reference property for each of the 10 applets to the name of this new symbolic string.
 - Clears redundant values for child locale objects. Each of the 10 applets now include a value in the String Reference property and a value in the String Override property. The value in the String Override property is redundant and the conversion import clears this value for each child locale object.
- 3 Deletes any records that the object locale records no longer reference.

For more information, see [“Converting Symbolic Strings” on page 197](#).

How Siebel CRM References Symbolic Strings at Run Time

Siebel CRM uses the same repository before and after a conversion. For example, an applet gets the value for the Title property from a child applet locale record. During conversion, Siebel Tools does the following:

- 1 Creates a symbolic string.
- 2 Places the reference for this symbolic string in the Title - String Reference property of the applet.
- 3 Removes the applet locale record.

After Siebel Tools finishes conversion it gets the applet title from the symbolic string. The display value for the title that it compiles is the same value that this title contained before the conversion. Siebel Tools compiles strings into object definitions in the repository. Siebel CRM reads symbolic strings from these object definitions. It does not read them from the S_SYM_STR table at run time.

Consolidating Symbolic Strings

Consolidating symbolic strings eliminates duplicate symbolic strings that the conversion utility might create when it converts these strings. The conversion utility converts all the symbolic strings for one object type, and then converts all the symbolic strings for the next object type. It does this until it finishes converting all the symbolic strings that you specify. It creates multiple symbolic string records for the same display value that occurs in multiple object types. Duplicate symbolic strings can include identical sets of locale records or one symbolic string might include more child locale records than another string. The strings that these objects use in common are identical. Consolidation can remove these duplicates.

To consolidate symbolic strings

- 1 Review the important caution information described in [“Caution About Converting and Consolidating Symbolic Strings” on page 200](#).

- 2 Convert symbolic strings.

For more information, see [“Converting Symbolic Strings” on page 197](#).

- 3 Run the following command to merge duplicate symbolic strings:

```
consoleapp "ConsolidationExport: Filename=ConsExp.txt, Repository=Siebel
Repository, LogFile=ConsolidationLog.txt, Language=ENU, MatchMin=2"
```

Use the values from the following table. For more information, see [“Running the Console Application Executable” on page 200](#).

Parameter	Description
Filename	The name of the export file.
Repository	The name of the repository.
LogFile	The name of the log file.
Language	Same as the Language parameter that you use to export candidates for an object type. For more information, see “Separating Conversion Files into Smaller Files” on page 201 .
MatchMin	The minimum number of matches that the utility must find in a set of matching symbolic strings before it writes them to the file. The default value is 2.
SkipSBLStrings	You can use one of the following values: <ul style="list-style-type: none"> ■ TRUE. Ignore all strings that include the SBL_ prefix in the Name property. ■ FALSE. Consolidate all strings that include the SBL_ prefix and all custom strings that you create. ■ MasterOnly. Do not consolidate strings that include the SBL_ prefix, but use them as master strings. The default value is TRUE.

After you convert locale strings to symbolic strings, you can use the consolidation utility to find duplicate symbolic strings and merge them and their references into a single symbolic string. As input, this utility uses the file that the consolidation export creates. It deletes redundant symbolic strings. It replaces all references that objects make to these strings with a reference to the master string. This consolidation might consume a significant amount of time because the object types in the repository include approximately 80 translatable string properties.

- 4 Import consolidated strings using the ConsolidationImport business service method.

For example:

```
consoleapp "ConsolidationImport: Filename=ConsExp.txt, Repository=Siebel
Repository,
LogFile=ConsolidationLog.txt, UnlockProjects=false, SkipParentUpdates=true"
```

These parameters are the same parameters that you use to import the symbolic strings in [Step 8 on page 199](#). For more information, see [“Running the Console Application Executable” on page 200](#).

- 5 (Optional) Separate a consolidation export file into smaller files.

To separate a consolidation export file into smaller files, you use the same parameters that you use to separate a conversion export file. You can then do [Step 4](#) to import each of these smaller files. For more information, see [“Separating Conversion Files into Smaller Files” on page 201](#).

Using a Batch File to Consolidate Strings

You can use a batch file to consolidate strings.

To use a batch file to consolidate strings

- 1 Navigate to the `SIEBEL_TOOLS_ROOT\BIN` folder.
- 2 Enter the following command:

```
strcons Action User_Id Password
```

You use the same parameters that you use when you use a batch file to convert strings. The only difference are:

- If you set the action parameter to Import, then the utility imports the files from the working folder that the TEST_LOCATION parameter in the batch file identifies.
- You do not specify an object type.

For more information, see [“Using a Batch File to Convert Strings” on page 204](#).

Using a Batch File to Convert Strings

You can run the conversion utility from a batch file.

CAUTION: If the Siebel Tools installation path includes a space, then you must enclose this path in quotes.

To use a batch file to convert strings

- 1 Navigate to the `SIEBEL_TOOLS_ROOT\BIN` folder.
- 2 Enter the following command:

```
strconv "object_type" action user_id password
```

For example:

```
strconv "Applet" export user_Id password
```

The following table describes the parameters you can use to run the strconv.bat (string conversion) batch file. You set other parameters in the batch file. For information about the batch file, see comments that the batch file contains.

Parameter	Description
object_type	Object type to convert. For example: <ul style="list-style-type: none"> ■ Applet ■ Control ■ List Column
action	You can use one of the following values: <ul style="list-style-type: none"> ■ Export. The utility exports all convertible locale records. ■ Import. The utility imports the file that the object_type parameter identifies.
user_Id	The user name that you use to log in to the Siebel application.
password	The user password that you use to log in to the Siebel application.

Configuring Locale Data

This topic describes how to configure locale data. It includes the following information:

- [Configuring Nontranslatable Locale Object Properties on page 205](#)
- [Displaying Controls and List Columns According to Locale on page 206](#)
- [Fixing Orphaned String References After an Upgrade on page 208](#)

Configuring Nontranslatable Locale Object Properties

A locale can be one of the following:

- **Translatable.** For example, a text string that defines a control label.
- **Nontranslatable.** For example, the HTML Sequence property, HTML Height property, or HTML Width property of a control.

User interface conventions can vary by locale. For example, one locale might require a different sequence of fields from another locale. To configure a nontranslatable object property for a locale, you can enable language override mode. This mode allows you to store a nontranslatable, locale property as a child locale record of the parent object.

In the following example, a Siebel enterprise requires Japanese (JPN) and four Western European languages. Japanese does not use middle names but Western European languages do use middle names. Japanese uses the family name first. Western European languages use the family name last.

To configure nontranslatable locale object properties

- 1 Set the language mode to JPN.

For more information, see [“Setting the Language Mode” on page 18](#).

- 2 Enable language override.

If you do not enable language override, and if you compile any of the Western European languages, then Siebel CRM uses the Japanese configuration of no middle name and family name first. For more information, see [“Enabling Language Override” on page 19](#).

- 3 In the Object List Editor, locate the object you want to modify.

For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).

- 4 To define locale values, modify the object properties:

- a To hide the middle name, remove the value from the Title-String Override property.
- b Reverse the order of the first name and last name.

You modify these locale values to meet the needs that this example requires. The locale values you must modify depend on your business requirements. You can use the Object List Editor or the layout editor.

- 5 Deploy your changes to the Siebel runtime repository.

Displaying Controls and List Columns According to Locale

You can control how Siebel CRM displays a control or list column according to the locale requirements.

CAUTION: To hide or display a control or list column, it is recommended that you modify a property. If you delete a control or list column, then Siebel Tools deletes it from all languages even if you enable language override.

The following example displays locale objects in an applet.

To display controls and list columns according to locale

- 1 Set the language mode.

For more information, see [“Setting the Language Mode” on page 18](#).

- 2 In the Object Explorer, expand the Applet tree.

- 3 Do one of the following:

- Click Control.
- Expand the List tree, and then click List Column.

- 4 In the Object List Editor, locate the object you want to modify.

For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23.](#)

- 5 Use the values from the following table to modify a property.

Object Type	Property	Description
Control	Visible	If TRUE, then Siebel CRM displays this control in the Siebel client in the parent language and in all other languages that it supports.
	Visible-Language Override	<p>If you enable language override, then you can set this property to one of the following values:</p> <ul style="list-style-type: none"> ■ FALSE. Hides the control in the Siebel client. ■ TRUE. Displays the control in the Siebel client. <p>For more information, see “Enabling Language Override” on page 19.</p>
List Column	Show in List	If TRUE, then Siebel CRM displays this column in the Siebel client in the parent language and in all other languages that it supports.
	Show in List-Language Override	<p>If you enable language override, then you can set this property to one of the following values:</p> <ul style="list-style-type: none"> ■ FALSE. Hides the column in the Siebel client. ■ TRUE. Displays the column in the Siebel client. <p>For more information, see “Enabling Language Override” on page 19.</p>

Fixing Orphaned String References After an Upgrade

An upgrade from one release of Siebel CRM to another release can result in some string references disappearing. The Fix Strings Utility allows you to locate these orphaned strings and update them with new references. This utility uses the Siebel Tools Fix String References business service and the FixStringReferences business service method.

To fix orphaned string references after an upgrade

- 1 Navigate to the `SIEBEL_TOOLS_ROOT\BIN` folder.
- 2 Open a Windows command line.
- 3 Enter the following command:

```
consol eapp config_file app_lang user_id password "Siebel Tools Fix String
References" "FixStringReferences: Repository=repository_name, PriorRepository=prior_r
epository_name, LogFile=log_file_name. Log, FixReferences=true_or_false, VerboseOutput
=true_or_false, Object=object_type"
```

Use the values from the following table.

Parameter	Description
config_file	Specify the path to and name of the Siebel Tools configuration file.
app_lang	Specify the current language, such as ENU.
user_id	Specify the user Id for the repository that this utility searches.
password	Specify the password for the repository that this utility searches.
repository_name	Required. Specify the name of the repository that includes the string references to fix.
PriorRepository	Specify the name of the repository that your deployment used before the upgrade. If you set the FixReferences parameter to TRUE, then you must include the PriorRepository parameter.
log_file_name	Required. Specify the name of the log file. This utility writes this log file to the current working folder. You can enter an explicit path for this log file.
FixReferences	You can set this parameter to one of the following values: <ul style="list-style-type: none"> ■ TRUE. Fix invalid references. ■ FALSE. Save invalid references to the log file. This is the default value.

Parameter	Description
VerboseOutput	You can set this parameter to one of the following values: <ul style="list-style-type: none"> ■ TRUE. Write progress information to the command line. ■ FALSE. Do not write progress information to the command line. This is the default value.
object_type	Specify the object type, such as Applet. The utility finds invalid string references that this object type references. If you do not include this parameter, then the utility finds invalid string references for all object types.

Example Commands That Fix Orphaned String References After an Upgrade

The following example runs the utility for the Business Service object type, writes progress output to the command line, and writes information to the fixstrings.log file:

```
consol eapp SIEBEL_TOOLS_ROOT\bin\enu\tools.cfg ENU jgolding db2 "Siebel Tools Fix String References" "FixStringReferences: Repository=Siebel Repository, LogFile=fixstrings.log, FixReferences=false, VerboseOutput=true, Object=Business Service"
```

The following example runs the utility for all object types and writes the results to the fixstrings.log file:

```
consol eapp SIEBEL_TOOLS_ROOT\bin\enu\tools.cfg ENU jgolding db2 "Siebel Tools Fix String References" "FixStringReferences: Repository=Siebel Repository, LogFile=d:\temp\fixstrings.log, FixReferences=false"
```

Using the Locale Management Utility

This topic describes how to use the Locale Management Utility. It includes the following information:

- [Finding Untranslated Text Strings on page 210](#)
- [Finding Existing Translations on page 211](#)
- [Finding Objects That the Locale Management Utility Modifies on page 212](#)
- [Finding Objects Modified Since the Last Export on page 212](#)
- [Exporting Text Strings and Locale Properties to a File on page 214](#)
- [Importing Text Strings and Locale Properties on page 215](#)
- [Modifying the Value in All Strings for a Language on page 216](#)
- [Using the Command Line to Run the Locale Management Utility on page 216](#)

The *Locale Management Utility* (LMU) is a utility in Siebel Tools that you can use to manage how you configure Siebel CRM to localize text strings, such as field labels, and other locale properties, such as the height and width of controls. You can use it to export text strings to a file. After Siebel Tools modifies or translates these strings in the file, you can use the Locale Management Utility to import them back into the repository. The Locale Management Utility also includes search and comparison tools.

Finding Untranslated Text Strings

You can use the Locale Management Utility to find text strings in the repository that Siebel CRM has not translated or to find text strings that Siebel Tools must retranslate because Siebel CRM modified the source string since the last translation. The Locale Management Utility searches or compares objects. It does not search or compare properties. If a locale object includes multiple string properties, then the search returns all strings that the locale object contains, even if only one of them is translated.

To find untranslated strings

- 1 Log in to Siebel Tools.
- 2 Click the Tools menu, Utilities, and then click Locale Management.
- 3 In the Locale Management dialog box, in the Options tab, in the Languages section, choose the source language and the target language.
- 4 In the Objects section, choose the applications or projects that you want to localize.
- 5 Click the Untranslated Strings tab.

The Untranslated Strings tab includes an Attribute column. Siebel CRM uses the terms attribute and property. These terms have the same meaning.

- 6 (Optional) To display strings that are marked as Redo, make sure the following check box contains a check mark:

Report String Attributes of Objects Marked With 'Redo' Flag

For more information, see [“How the Locale Management Utility Indicates That Siebel CRM Modified a Record Since the Last Export” on page 213](#).

- 7 Click Find Strings.

The Locale Management Utility searches the properties of the string objects. It searches the objects only in the application or project you choose. It displays object definitions that include strings that are not translated. If you choose to display Redo objects in [Step 6](#), then it also displays the strings that you must retranslate.

- 8 (Optional) To do more, you can click the following buttons:
 - **Find Views.** Find the views that reference the untranslated strings.
 - **Go To.** You choose an object in the Results list, and then click Go To. The Object Explorer displays the object definition that includes the string.

- **Export.** Export all untranslated strings to a file.
- 9 (Optional) Determine if an existing translation exists that you can use for the untranslated strings that the Locale Management Utility shows in [Step 7](#).

For more information, see [“Finding Existing Translations” on page 211](#).

Finding Existing Translations

To find an existing translation that you can use for an untranslated string, you can use the Locale Management Utility to search the objects in the repository. You can reuse an existing translation for an object that you create or modify.

To find existing translations

- 1 Click the Tools menu, Utilities, and then click Locale Management.
- 2 In the Locale Management dialog box, in the Options tab, in the Languages section, choose the source language and the target language.
- 3 In the Objects section, choose the application or project that you want to localize.
- 4 Click the Untranslated Strings tab.
- 5 Click the Find Translations button.

After a few moments, the Locale Management Utility displays the results. For more information, see [“How the Locale Management Utility Finds Existing Translations” on page 211](#).

How the Locale Management Utility Finds Existing Translations

The Locale Management Utility compares an untranslated string to the properties that might contain this string in other objects in the repository. If it finds an object that includes the same string, then it searches for a translation in the language that you choose as the target language. If it finds a translation, then it displays the best candidates for translation in the Results list and allows you to export it to a file.

For example, you choose English-American as the source language and Spanish as the target language. You create an applet with a title of Customer that is not translated. If you click Find Translation in the Locale Management dialog box, then the Locale Management Utility searches the repository for other objects that include a property that includes the Customer text string. If it finds a match, then it searches for a Spanish translation of this string. If a translation already exists, then it displays this translation and you can export it to a file.

How the Locale Management Utility Chooses Among Multiple Translations

If the Locale Management Utility finds more than one translation for a source string, then it does the following:

- If the source string resides in the property of an object that is related to a business component, such as Control Caption or List Column Display Name, then the Locale Management Utility examines translations from the same business component first, and then does the following:
 - If multiple translations exist in the same business component, then it chooses the string that occurs most frequently.
 - If no translations exist in the same business component, then it chooses the translation that occurs most frequently in all business components.

For example, Applet A references the Account business component. Applet A includes a control caption with the value of Account, and this value is translated to Account_FRA for French. You create a new applet, Applet B, that also references the Account business component, and Applet B also includes a control caption with the value of Account. If you run Find Translations, then the Locale Management Utility finds Account_FRA as an existing translation and chooses it as the best candidate for this string.

- If the source string is not a property related to a business component, such as Menu Item Caption, then the Locale Management Utility chooses the translation that occurs most frequently.

Finding Objects That the Locale Management Utility Modifies

This topic describes how to find objects that the Locale Management Utility modifies.

To find objects that the Locale Management Utility modifies

- 1 Click the Tools menu, Utilities, and then click Locale Management.
- 2 In the Locale Management dialog box, in the Options tab, in the Languages section, choose the source language and the target language.
- 3 In the Objects section, choose the application or project that you want to localize.
- 4 Click the Modified Objects tab.
- 5 In the Search Criteria section, make sure the Changed Since check box includes a check mark.
- 6 Click the Changed Since drop-down arrow, and then choose the date that the Locale Management Utility must use as the starting date for the search.
- 7 Click Start.

Finding Objects Modified Since the Last Export

You can use the Locale Management Utility to find objects in the repository that were modified since the last time you exported strings. This search might be useful if your development and localization efforts occur simultaneously. It helps you keep strings in the repository synchronized with the strings that you export to a file for localization.

To find objects modified since the last export

- 1 Click the Tools menu, Utilities, and then click Locale Management.
- 2 In the Locale Management dialog box, in the Options tab, in the Languages section, choose the source language and the target language.
- 3 Click the Modified Objects tab.
- 4 (Optional) In the Search Criteria section, set the Locale Management Utility to search from a date:
 - Make sure the Changed Since check box includes a check mark.
 - Click the Changed Since drop-down arrow, and then choose the date that the Locale Management Utility must use as the starting date for the search.

If you use the Changed Since option, and if you click Start, then the Locale Management Utility marks all records it returns for a modified project as Redo regardless of whether Siebel CRM modified a locale property. It does this because it searches for modifications that reside at the object level, not at the property level. For more information, see [“How the Locale Management Utility Indicates That Siebel CRM Modified a Record Since the Last Export” on page 213](#).
- 5 (Optional) In the Search Criteria section, set the Locale Management Utility to compare objects in the repository to objects in a source file:
 - Make sure the Different From File check box includes a check mark.
 - Click Browse, and then choose the source file.
- 6 Click one of the following buttons:
 - **Start.** Finds records that match the search criteria, flags the records that it returns as Redo, and then displays the results.
 - **Preview.** Finds records that match the search criteria and displays the results. It does not flag records as Redo.
- 7 (Optional) To do more, you can click the following buttons:
 - **Save.** Saves a result set in a log file.
 - **Go To.** The Object Explorer displays the parent object of the string or property.
 - **Load.** Imports a result set from a previously saved file. After the Locale Management Utility displays this result set in the Results list, you can use Go To to examine each record.

How the Locale Management Utility Indicates That Siebel CRM Modified a Record Since the Last Export

If Siebel CRM modified a record in the repository since the last export, then it might require another translation. The Locale Management Utility uses the Redo flag to mark this record.

If you use the Locale Management Utility to import records, then it compares the source language records in the repository with the source language records in the import file. If Siebel CRM modified the records in the repository since the export occurred, then this utility uses the Redo flag to mark the target language records. This configuration helps you identify records that might require another translation. For more information, see [“Importing Text Strings and Locale Properties” on page 215](#).

Exporting Text Strings and Locale Properties to a File

You can use the Locale Management Utility to export strings and other locale properties to a file. This file can use any one of the following file types:

- .slf
- .txt
- .xlf

You cannot use the Locale Management Utility to export to a Microsoft Excel.xls file.

To export text strings and locale properties to a file

- 1 In Siebel Tools, click the Tools menu, Utilities, and then click Locale Management.
- 2 In the Locale Management dialog box, in the Options tab, in the Languages section, choose the source language and the target language.

To export text strings and locale properties, the language mode and the Locale Management Utility source language must use the same language. For more information, see [“Setting the Language Mode” on page 18](#).

- 3 In the Objects section, choose the application or project that the Locale Management Utility must export.
- 4 Click the Export Tab.
- 5 Click String Attributes Only or All Localizable Attributes.

A localizable property can include translatable strings and other locale properties, such as the width and height of controls. Each locale property might contain a different value that meets the need for a specific locale. Siebel CRM uses the terms attribute and property. These terms have the same meaning.

- 6 Click Export.
- 7 In the Save As dialog box, choose the folder where the Locale Management Utility must export the file.

For example, choose `SIEBEL_TOOLS_ROOT\OBJECTS\language_code`, where `language_code` is the target language you choose in [Step 2](#).

- 8 Enter a file name, choose a file type, and then click Save.
 - If you click String Attributes Only in [Step 5](#), then the available file types are .txt or .xlf.
 - If you click All Localizable Attributes in [Step 5](#), then the available file type is .slf.

Importing Text Strings and Locale Properties

You can use the Locale Management Utility to import translated strings and other locale properties into the repository. You can preview the import before the utility actually does the import.

To import text strings and locale properties

- 1 Click the Tools menu, Utilities, and then click Locale Management.
- 2 In the Locale Management dialog box, in the Options tab, in the Languages section, choose the source language and the target language.
- 3 Click the Import tab.
- 4 Enter the name of the file that the utility must import.

You can use the Browse button to choose the file. The default file name is one of the following:

- Resul ts. txt if the file includes strings only
- Resul ts. sl f if the file includes all locale properties

If you import an XML Localization Interchange Field file (.xlf), then make sure a working Internet connection exists during the import.

- 5 (Optional) To preview the import, do the following:
 - a Enter the path and name of the file where the utility must store the results for previewing.

The default file name is `previ ew. l og`.

- b Click Preview.

The Locale Management Utility writes the results of the import to the log file. It does not write the results to the repository. It does not mark modified records with a Redo flag during preview.

- 6 (Optional) To mark records that Siebel CRM modified since the export occurred, make sure the following check box contains a check mark:

Mark Changed Records with Redo Flag

For more information, see [“How the Locale Management Utility Indicates That Siebel CRM Modified a Record Since the Last Export” on page 213](#).

- 7 Click Import.

The utility imports the locale properties into the repository. It creates the following log file in the `SIEBEL_TOOLS_ROOT\OBJECTS` folder:

`LMUI mportTruncati on. l og`

This file includes detailed information and error messages about records that the utility did not import.

Modifying the Value in All Strings for a Language

You can use the Locale Management Utility to replace strings in bulk. For example, you want to configure Siebel CRM to modify all text strings that include Accounts to Companies for the English locale. You can use the Locale Management Utility to export the strings to a file, modify the file so that it includes only Companies instead of Accounts, and then import these strings into the repository.

Using the Locale Management Utility to replace strings is most useful for strings that Siebel CRM stores in string-override fields. For information about modifying symbolic strings, see [“Modifying a Symbolic String to Globally Update Display Values” on page 193](#).

To modify the value in all strings for a language

- 1 Export the strings you want to modify to a file.

The source language and the target language cannot be the same language. For more information, see [“Exporting Text Strings and Locale Properties to a File” on page 214](#).

- 2 Use a text editor to modify the strings in the file that you created in [Step 1](#):

- Remove strings that you do not want to replace from the file.
- In the Target String column of the file, enter the new string that replaces the old value.

- 3 Import the file.

For more information, see [“Importing Text Strings and Locale Properties” on page 215](#).

Using the Command Line to Run the Locale Management Utility

You can use the command line to run the Locale Management Utility. The commands that this topic describes use the following format:

- `xxx`. A placeholder for a required parameter.
- `[xxx]`. A placeholder for an optional parameter.
- `xxx|yyy`. An OR condition (for example, `xxx or yyy`).

You must include the absolute path with any file name that you specify. For example, if you specify the export file as `result.txt`, then the utility creates this file in the current folder. In this example, if the installation folder is `C:\Program Files\Siebel\8.0\Tools`, then the utility creates this file in the following folder:

```
C:\Program Files\Siebel\8.0\Tools\BIN
```

It does not create it in the following folder:

```
C:\Program Files\Siebel\8.0\Tools\OBJECTS
```


Using the Command Line to Export Strings and Locale Properties

The command that this topic describes allows you to export localizable properties for all projects or for all applications. You can specify one of the following values:

- **all.** Export all translatable properties and language override properties to a file that uses an .slf extension.
- **string.** Export only string properties to a file that uses a .txt or .xlf extension. If you do not specify a file name, then the utility displays an error.

Format

This command uses the following format:

```
/lmu source_language target_language export proj|app all|string file [project_file]
```

Project Parameter

The Locale Management Utility includes the following parameter:

```
project_file
```

You can use it to identify the projects to export. This parameter identifies the name of an ASCII text file that includes a list of projects that are delimited by a line feed. If you do not include the project_file parameter, then the utility exports all projects.

You can use the proj|app parameter to identify the projects or applications that include the strings to export. If you use the project_file parameter, then you must choose proj. If you choose app, and if you include the project_file parameter, then the utility ignores this file.

Example

The following command exports properties:

Example 1:

```
Siebedev /c tools.cfg /u sadmin /p sadmin /d serverdatasrc /lmu ENU DEU export proj
string C:/lmu/lmu_1.xlf /ws "ws1" "Siebel Repository"
```

This example exports all string properties and language override properties for the projects that the following file lists:

```
C: temp\proj_to_exp.txt
```

It exports these properties to the following file:

```
C:\temp\named_my_proj_results.txt
```

The source language of the file is English-American (ENU) and the target language is French (FRA).

Using the Command Line to Import a File

The command that this topic describes allows you to import a file. If the source string in the import file is different than the source string in the repository, then it marks the target locale object as Redo.

Format

This command uses the following format:

```
/lmu source_language target_language import file
```

Example

The following command imports a file:

```
siebdev /u sadmin /p db2 /d server /lmu ENU FRA import "C:\Program  
Files\Siebel\8.0\Tools\objects\result.sif"
```

This example imports the `result.sif` file from the following folder. This folder is the installation folder for an earlier version of Siebel Tools:

```
C:\Program Files\Siebel\8.0\Tools\objects
```

The source language of the file is English-American (ENU) and the target language is French (FRA).

The file includes all localizable string properties and language override properties.

Using the Command Line to Export Strings That Require Translation

The command that this topic describes allows you to export all untranslated strings and strings marked with the Redo flag to a file. It can export all projects or all applications. The exported file includes the related view names.

Format

This command uses the following format:

```
/lmu source_language target_language todo proj|app [file]
```

Example

The following command exports strings that require translation:

```
Siebdev /c tools.cfg /u sadmin /p sadmin /d serverdatasrc /lmu ENU DEU export proj  
string C:/lmu/lmu_1.xlf /ws "ws1" "Siebel Repository"
```

This example finds all untranslated strings and redo strings for all applications. It exports the results to the following file:

```
C:\Program Files\Siebel\8.0\Tools\objects\result.txt
```

The source language is English-American (ENU) and the target language is French (FRA).

11 Testing and Troubleshooting Your Customizations

This chapter describes how to use Siebel Tools to test and troubleshoot your customizations. It includes the following topics:

- [Testing and Troubleshooting Your Modifications on page 219](#)

Testing and Troubleshooting Your Modifications

This topic includes the following information:

- [Setting Debug Options to Open the Siebel Client on page 219](#)
- [Recovering from a Failed Development Server on page 220](#)

To test your modifications, you can use a local instance of the Siebel Web Client that runs on your computer. If you compile objects and test the results locally, then consider the following:

- If you compile modifications to a repository, then any local instances of the Siebel Web Client that are open and that read this repository automatically close, reopen, and then display the modified configuration.
- You can configure Siebel Tools to automatically open the Siebel Web Client and read the most current repository.
- To display your modifications in a local instance of the Siebel Web Client, this client must read the repository that you compile.

This task is a step in [“Roadmap for Setting Up and Using Siebel Tools” on page 16](#).

For more information about installing the Siebel Web Client, see the *Siebel Installation Guide* for the operating system you are using.

For information about using debug windows, see the following information:

- [Using the Calls Window on page 135](#)
- [Using the Watch Window While it Monitors a Script on page 136](#)

Setting Debug Options to Open the Siebel Client

The debug options allow you to modify the run time settings that open an instance of the Siebel Web Client in the following situations:

- You enable Auto-start Web Client.

- You click the Debug menu, and then click Start to start an instance of the Siebel Web Client. You typically do this if you want to debug Siebel eScript or Siebel VB. For more information, see *Siebel eScript Language Reference* or *Siebel VB Language Reference*.

To set debug options to open the Siebel client

- 1 In Siebel Tools, click the View menu, and then click Options.
- 2 In the Development Tools Options dialog box, click the Debug tab.
- 3 Set the debug options.

Make sure you set the following options:

- Executable
- CFG File
- Browser
- Working Directory

For more information, see [“Development Options for Debugging” on page 272](#).

Recovering from a Failed Development Server

This topic describes how to recover from a failed development server or failed repository when you cannot access a backup copy of this server or repository. It requires that a recent image of the repository exist on the computer that a remote user uses, and that this user performed a full Get to create this repository.

To recover from a failed development server

- 1 Log in to Siebel Tools on the remote computer that includes the intact repository.
- 2 Archive the projects that this repository contains.
For more information, see [“Exporting Objects to an Archive” on page 222](#).
- 3 Create a new repository:
 - To create a repository record, you can add a new repository object in Siebel Tools.
 - To create the schema in the database for the repository, you can run `imprep.ksh` for Linux or UNIX.For more information, see [“Exporting and Importing Repositories” on page 169](#).
- 4 Import the archive you created in [Step 2](#) into the repository you created in [Step 3](#).
For more information, see [“Importing Objects from an Archive” on page 223](#).
- 5 Test the repository you created in [Step 3](#) and make sure it includes the expected functionality.
- 6 Check projects into the Siebel Server.

12 Archiving Objects

This chapter describes how to archive objects. It includes the following topics:

- [Overview of Archiving Objects on page 221](#)
- [Exporting Objects to an Archive on page 222](#)
- [Importing Objects from an Archive on page 223](#)
- [Using the Application Deployment Manager on page 230](#)

Overview of Archiving Objects

Archiving is a process where you export objects from the repository to an archive file. You can then import objects from the archive file back into the repository. You can use an archive to back up sets of objects or move them to another environment that shares the same physical database schema as the source environment. Note the following:

- An archive does not depend on a database because it includes only repository information. You can use it to exchange repository data between environments with different database platforms, including local and server databases, as long as these databases use the same schema.
- If you import objects from an archive, then you can specify conflict resolution rules for each object. You can configure Siebel Tools to ignore an imported object, replace an existing object with an imported object, or merge two objects according to object properties.
- You can archive individual objects or entire projects to an archive.
- You can use code control software to control an archive.
- If you want to back up or move the entire repository to another environment, then see [“Exporting and Importing Repositories” on page 169](#).

For more information about the files that you use when archiving objects, see [“Files That You Use to Manage Repositories” on page 160](#).

Using Export and Import for Different Versions of Siebel CRM

If you export objects from one version of Siebel CRM to another version, then do not import these objects through .sif files into a different version of Siebel CRM because Siebel CRM might modify object definitions between these versions. If you import an object that is not valid, then an invalid configuration might result. Oracle does not support an invalid configuration.

Where Siebel Tools Stores Archive Files

The Temp parameter in the Siebel section of the tools.cfg file identifies the folder where Siebel Tools stores the following items:

- Archive files
- Projects and objects that you check out or check in

Siebel Tools sets this Temp parameter to the *client_root\TEMP* folder of your Siebel Tools installation folder, by default.

Exporting Objects to an Archive

You can export top-level objects to an archive, such as business components, applets, views, and projects. Siebel Tools exports and imports child objects with their parents. For information about top-level object types, see [“Displaying Object Types in the Object Explorer” on page 25](#).

NOTE: Web Tools supports only export in Innovation Pack 2017.

Do not export the repository object to export an entire repository. If you do this, then the export file is too large and it degrades performance. For more information, see [“Exporting Repositories” on page 173](#).

This task is a step in [“Roadmap for Setting Up and Using Siebel Tools” on page 16](#).

To export objects to an archive

- 1 In the Object List Editor, locate the object you want to export.

For more information, see [“Locating and Modifying Object Definitions in the Object List Editor” on page 23](#).

- 2 Click the Tools menu, and then click Add To Archive.

Siebel Tools displays the Export to Archive File dialog box. The status bar in this dialog box indicates the child objects that Siebel Tools includes in the export. When the process finishes, this dialog box displays the top-level objects in the Objects to Archive list.

TIP: The dialog box displays as Add To Archive in Web Tools. The Objects textbox displays the name of the selected object. You can archive only one object at a time.

- 3 If you want to add more objects, then repeat [Step 1](#) through [Step 2](#) for each object. Do not close the Export to Archive File dialog box while you add these objects.

You can add objects of the same object type or you can add objects of another object type.

- 4 In the Export to Archive file dialog box, in the Archive File window, enter the path and file name of the archive file.

For example, enter C:\Siebel\8.2\Tools_1\objects\objects.sif.

- 5 Click Save.

Siebel Tools creates a SIF file in the location that you specify in [Step 4](#). Web Tools creates the same file and exports using the browser functionality.

NOTE: Use Siebel Tools to import an archive.

Using the Command Line to Export Objects to an Archive

You can use the command line to export objects to an archive.

To use the command line to export an archive

- 1 Open a command line, and then navigate to the `SIEBEL_TOOLS_ROOT\BIN` folder.
- 2 Use the following command to run the `siebdev.exe` file:

```
siebdev /c config_file /d database /u user_name /p password /ws workspace_name /
batchexport repository_name input_file_name log_file
```

The following example exports the objects that the `obj.txt` input file specifies. It logs results to the `export.log` file:

```
siebdev /c tools.cfg /d sample /u sadmin /p sadmin /ws MAIN /batchexport "siebel
repository" obj.txt export.log
```

For more information about the input file, see [“Input File That Batch Export Uses” on page 223](#).

Input File That Batch Export Uses

The batch export switch uses an input file that specifies the objects to export. This input file uses the following format. You cannot use a space before or after a comma in this file:

```
object_type, object_name, query_expression, file_name.sif
```

where:

- *object_type* identifies the object type to export, such as Business Component.
- *object_name* identifies the name of the object to export, such as Account.
- *query_expression* can include any query that Siebel Tools can use. For more information, see [“Using the Query Menu to Run a Query” on page 31](#).
- *file_name.sif* can use an absolute file path or a relative file path to the current folder.

You can include multiple lines in the input file and each of these lines can specify to export multiple objects to a different SIF file. If you specify the same SIF export file in multiple lines, then batch export uses only the last export that you specify.

For example, consider the following line from an input file. In this example, the batch export switch exports all business components where the Name property is like `*Account*` to a repository file named `export.sif`:

```
Business Component, *Account*, export.sif
```

Importing Objects from an Archive

This topic describes how to import objects from an archive. It includes the following information:

- [Preparing the Siebel Tools Environment to Import Objects from an Archive on page 224](#)

- [Importing an Archive on page 224](#)
- [Importing Multiple Archives on page 226](#)
- [Using the Command Line to Import an Archive on page 227](#)
- [Using the Review Conflicts and Actions Dialog Box of the Import Wizard on page 229](#)

Preparing the Siebel Tools Environment to Import Objects from an Archive

You must prepare the Siebel Tools environment before you import an archive.

To prepare the Siebel Tools environment to import objects from an archive

- 1 Make sure the local computer can access the import file through a network or local drive.
- 2 Make sure the repository you use to create the archive and the repository to which Siebel Tools imports the archive use the same schema version.

You can export or import an archive only among repositories that use the same schema version. To determine the schema version that a repository uses, see [“Viewing Information About the Current Repository” on page 160](#).

- 3 Make sure the repository is open in Siebel Tools and is the active repository.
For more information, see [“Viewing Information About the Current Repository” on page 160](#).
- 4 Make sure the projects that the import affects are checked out.
For more information, see [“Locking Projects Before You Import an Archive” on page 224](#).

Locking Projects Before You Import an Archive

For non-workspace enabled objects, you need to lock the project before importing an archive file. The Import wizard informs you of any unlocked projects that you must lock. It does this after it analyzes the objects in the archive and compares them to the objects that reside in the repository. For workspace-enabled objects, object locking is done at the database level. Therefore, there is no need to lock any project or object for importing.

Importing an Archive

This topic describes how to use Siebel Tools to import an archive.

To import an archive

- 1 Log in to Siebel Tools.

- 2 Prepare the Siebel Tools environment to import an archive.

For more information, see [“Preparing the Siebel Tools Environment to Import Objects from an Archive” on page 224](#).

- 3 Click the Tools menu, and then click Import From Archive.

- 4 In the Select Archive To Import dialog box, choose the archive file, and then click Open.

Siebel Tools displays the Import Wizard - Preview dialog box. It lists the projects and other top-level objects that the archive file you opened contains.

- 5 Specify the work that Siebel Tools does if it determines that the archive and the repository both contain the same top-level object. In the Conflict Resolution section, choose an option.

For more information, see [“Options You Can Choose to Resolve a Conflict” on page 226](#).

- 6 Click Next.

Siebel Tools does one of the following:

- If the objects in the archive already exist in the repository, and if Siebel Tools finds no conflicts, then it makes no modifications. It displays a dialog box that describes that it found no conflicts and made no modifications to the repository. In this situation, click OK and exit this procedure.
- If any object involved in this import is not locked, or if any project that this object references is not locked, then Siebel Tools displays a warning message. You must cancel the import, lock the objects and projects, and then restart the Import wizard.
- If the objects in the archive already exist in the repository, and if Siebel Tools finds a conflict, or if the objects do not exist in the repository, then it displays the Import Wizard - Review Conflicts and Actions dialog box. This dialog box includes information about the differences. In this situation, go to next step.

- 7 In the Import Wizard - Review Conflicts and Actions dialog box, in the Conflicting Objects section, choose an object.

If you choose an object, then Siebel Tools displays differences in the Object Differences and Attribute Differences sections. Siebel CRM uses the terms attribute and property. These terms have the same meaning. For more information, see [“Using the Review Conflicts and Actions Dialog Box of the Import Wizard” on page 229](#).

- 8 To make an adjustment, right-click a difference, and then choose an action.

- 9 Click Next.

Siebel Tools displays the Summary dialog box and starts the import.

- 10 When Siebel Tools finishes the import, you can click Finish.

Siebel Tools creates a log file named importlog.txt in the following folder:

SIEBEL_TOOLS_ROOT\TEMP

It includes the list of messages that it displayed in the Summary dialog box. To maintain a record of this import, you can modify the importlog.txt file name to the current date.

Options You Can Choose to Resolve a Conflict

A *conflict* is a situation that occurs if an object that Siebel Tools attempts to import from an archive does not match the corresponding object in the repository. Siebel Tools examines these objects to determine if they include the same properties, property values, and child objects. If these items are not the same, then Siebel Tools recognizes this situation as a conflict.

Table 18 describes options you can choose to resolve a conflict that occurs during an import. It describes the work that Siebel Tools does if it determines that the archive and the repository both contain the same top-level object.

Table 18. Options You Can Choose to Resolve a Conflict

Option	Description
Overwrite the Object in the Repository	Siebel Tools deletes the object in the repository and the child objects of this object, and then copy the object and child objects from the archive to the repository.
Merge the Object Definitions From the Archive With the Definition in the Repository	<p>Siebel Tools does the following:</p> <ul style="list-style-type: none"> ■ Replaces the properties that are different in the repository with the properties that the archive contains. ■ If the archive includes child objects that the repository does not contain, then it copies these objects from the archive to the repository. ■ Does not modify child objects in the repository that are not also in the archive. <p>When Siebel Tools finishes this merge, the top-level objects in the repository include the same properties and child objects that the object in the archive includes. It also includes any child objects that already exist in the repository.</p> <p>Siebel Tools merges objects, by default. This option is typically the safest option.</p>
Do Not Import the Object Definition From the Archive	Siebel Tools does not modify the objects in the repository.

Importing Multiple Archives

This topic describes how to import multiple archives.

To import multiple archives

- 1 Log in to Siebel Tools.

- 2 Prepare the Siebel Tools environment to import an archive.

For more information, see [“Preparing the Siebel Tools Environment to Import Objects from an Archive” on page 224](#).

- 3 Click the Tools menu, and then click Import from Archives.
- 4 In the Select Archives to Import dialog box, choose one or more SIF files to import, and then click OK.
- 5 In the Conflict Resolution section of the Import from Archives dialog box, choose an option.

You specify the work that Siebel Tools must do if it determines that the archive and the repository both contain the same top-level object. For more information, see [“Options You Can Choose to Resolve a Conflict” on page 226](#).

You can also do the following, as necessary:

- To control the order that Siebel Tools uses to import the archive, choose a row, and then click the up arrow or down arrow.
 - To add more archive files, click Add, choose the files you want to add, and then click OK. Siebel Tools does not add duplicate files to the import list.
 - To remove an archive, choose the archive you want to remove, and then click Remove.
 - To view the objects that a SIF file includes, choose this SIF file in the SIF Files list, and then click Preview. After you finish previewing, click OK.
- 6 In the Import from Archives dialog box, Click Import.

Siebel Tools begins the import in the order that you specify. You can click Cancel to cancel the import.

Siebel Tools displays Done in the status bar at the end of the Import from Archive(s) dialog box when the import finishes. It displays the following status for a SIF file in the SIF Files list:

- **Completed.** Import for this SIF file completed successfully.
 - **Failed.** Import for this SIF file encountered an error and did not complete successfully.
- 7 If a conflict occurs, then review the Import from Archive(s) - Review Conflicts and Actions dialog box, and then click Continue.

Using the Command Line to Import an Archive

You can use the command line to import objects from an archive. You can also use the command line to do other work.

To use the command line to import an archive

- 1 In Siebel Tools, make sure the projects that the import affects are checked out or locked. You cannot use the command line to check out or lock these projects. For more information, see Open a command line, and then navigate to the `SIEBEL_TOOLS_ROOT\BIN` folder.

2 Use the following command to run siebdev.exe:

```
siebdev.exe /c tools.cfg /d ServerDataSrc /u SADMIN /p MSSQL /ws dev_sadmin_m9 /  
batchimport "Siebel Repository" merge "C:\Tools75\import.sif" import.log
```

Here /ws refers the workspace name and this workspace needs to be in the editable state for import to be successful. You can use the full path or the relative path to the current folder to specify the SIF file and the log file.

To import, also note the following options:

- *import_mode* determines how to handle a conflict. You can use one of the following values with it:
 - overwrite.
 - merge.
 - skip. If you use skip, and if a corresponding object does not exist in the repository, then there is no conflict and the utility imports the record.

For information, see [“Options You Can Choose to Resolve a Conflict” on page 226](#).

- *sif_files* specifies the name of one or more sif files that contains the objects that batchimport imports, such as *sif_file1*, *sif_file2*, *sif_file_n*, or the path to a folder that includes .sif files.
- *log_file* specifies the name of the file that siebdev uses to log information.

Example of Using the Command Line to Import an Archive

The following example imports an archive:

```
siebdev.exe /c tools.cfg /d ServerDataSrc /u SADMIN /p MSSQL /ws dev_sadmin_m6 /  
batchimport "Siebel Repository" overwrite import1.sif, import2.sif" import.log
```

It uses the overwrite option for conflict resolution. It logs the results to the import.log file. For more information, see [“Options You Can Choose to Resolve a Conflict” on page 226](#).

Example of Importing All Files

The following example imports all files:

```
siebdev.exe /c tools.cfg /d ServerDataSrc /u SADMIN /p MSSQL /ws dev_sadmin_m4 /  
batchimport "Siebel Repository" merge C:\23075\Tools\importfiler import.log
```

This command imports all files in the following folder:

```
C:\23075\Tools\importfiler
```

It uses the merge option for conflict resolution. It logs the results to the import.log file.

Using the Review Conflicts and Actions Dialog Box of the Import Wizard

Figure 11 includes the dialog box that appears if Siebel Tools determines that a difference exists between an object in the repository and an object in the archive. You can use this dialog box to review differences and to modify the work that Siebel Tools does to resolve the conflict.

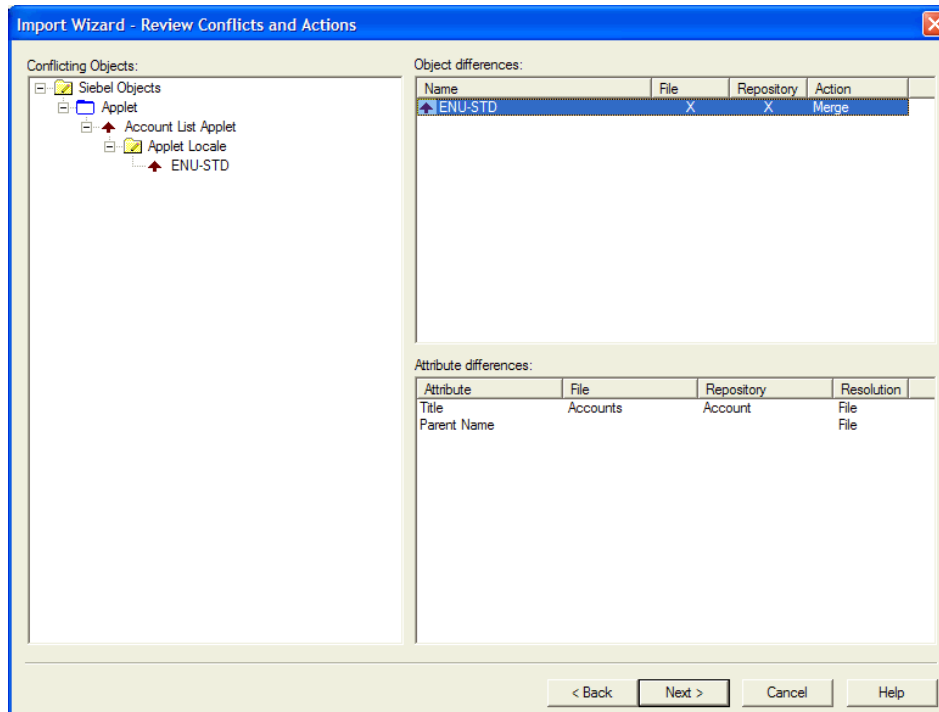


Figure 11. Review Conflicts and Actions Dialog Box of the Import Wizard

Conflicting Objects Section

The Conflicting Objects section displays the hierarchy of objects that include differences. This hierarchy uses the same structure as the object type hierarchy in the repository but it displays only objects that include conflicts rather than all repository objects. The objects in the Object Differences dialog box include objects for all hierarchical levels, not just top-level objects.

Object Differences Section

The Object Differences section lists the objects that include a difference. It includes the following columns:

- **File column.** An X in this column indicates that the object exists in the archive.
- **Repository column.** An X in this column indicates that the object exists in the repository. An object can exist in the archive and in the repository. These columns provide only information. You cannot modify them.

- **Action column.** Indicates the proposed resolution for the object. Siebel Tools uses the choices you make in the Conflict Resolution section of the Preview pane to determine the work that it does to display information in this column. To modify this work, you can right-click the value in the Action column, and then choose another action. For a description of these values, see [“Options You Can Choose to Resolve a Conflict” on page 226.](#)

Attribute Differences Section

The Attribute Differences section displays the property value conflicts for the object you choose in the Object Differences section. It lists only the properties that include a conflict. Siebel CRM uses the terms attribute and property. These terms have the same meaning.

Table 19 describes the columns in the Attribute Differences section.

Table 19. Columns in the Attribute Differences Section

Column	Description
Attribute	Name of the object.
File	Value of the object in the archive.
Repository	Value of the object in the repository.
Resolution	<p>Includes one of the following values:</p> <ul style="list-style-type: none"> ■ File. Siebel Tools deletes the value in the repository during the import and replaces it with the value from the archive. ■ Repository. Siebel Tools does not modify the value in the repository. <p>You can modify this resolution depending on the following value that the Action column in the Object Differences contains:</p> <ul style="list-style-type: none"> ■ Merge. You can modify the resolution. To modify it, you can right-click the value in the Resolution column, and then click Repository or File. ■ Not Merge. You cannot modify the resolution.

Using the Application Deployment Manager

The *Application Deployment Manager* (ADM) is a feature that you can use to administer a Siebel CRM deployment. Siebel Tools supports ADM through the following business service:

Siebel Tools Export Support for ADM

It allows you to export individual objects to a hotfix or to export all objects that Siebel CRM modified after a particular date and time to a mid-level release. For more information, see *Siebel Application Deployment Manager Guide*.

Index

A

- Applet Layout Editor, using** 41
- Applet Menu Layout Editor, using** 41
- applets**
 - controls and columns for editing 36
 - using window for 35
- Application Deployment Manager**
 - about using 230
- Archive versus Archive option** 155
- archives**
 - comparing objects in two different 155
 - exporting objects to 222
 - importing 224
 - importing multiple 226
 - importing objects from 223
 - preparing Siebel Tools environment to import objects from 224
 - using command line to export objects to 223
 - using command-line interface to import objects from 227
 - using to export and import objects 221

B

- batch files**
 - using to consolidate strings 204
 - using to convert strings 204
- bookmarks**
 - and Go menu 249
 - and History toolbar 259
- Bookmarks window, using** 37
- breakpoints, using** 135
- Browser Script Editor, about using** 44
- browser, choosing target** 17

C

- Calls window, using** 135
- Changed property, using** 29
- command-line interface**
 - about using 44
 - using to export objects to archives 223
 - using to export strings that require translation 218
 - using to import a file 217
 - using to import objects from archives 227
 - using to run Locale Management Utility 216
- Compare Objects dialog box**

- elements of 268
- using to compare object in current repository and archive file 154
- using to compare objects in archive files 155
- using to compare objects in different repositories 154
- using to compare objects in the same repository 154
- using to synchronize objects between repositories 155

compound queries, about creating 33

Configuration Context toolbar, about using 263

configuration file, modifying to support symbolic strings 191

Confirmation dialog boxes, controlling display of 19

conflict resolution

- displaying hierarchy of differences for objects 229
- displaying object definition to a row 229
- displaying property value conflicts for object definitions 230

consoleapp.exe

- running 200
- using to import consolidated strings 203
- using to run string conversion utility 197, 202
- using to split export files into smaller files 201

Controls/Columns window, using 36

Conversion Export utility

- and string conversion 201
- parameters for 274

Conversion Import utility

- and string conversion 201
- parameters for 275

D

database

- updating remote 180

Database Configuration Wizard, using to migrate repositories 177

Debug menu, options in 250

debug options

- for development 272
- setting 133

- setting to open Siebel Web Client 219
- Debug toolbar, buttons on** 260
- delivering, about** 48
- design environments, canvas-based** 42
- Detail tab, using** 27
- development environment**
 - enable workspaces 16
 - process of setting up 16
 - resetting options in 19
- development server, recovering from**
 - failed** 220
- dialog boxes**
 - that compare objects 268
 - that set development options 269
 - that validate objects 265

E

- Edit menu, options in** 245
- Edit toolbar, buttons on** 257
- edit tools**
 - displaying from Edit menu 245
 - displaying from Edit toolbar 257
- editors**
 - using 41
 - using canvas-based 42
- environment settings, applying using View menu** 246

F

- File menu, options in** 244
- Fix and Go, using** 128
- Flat tab, using** 27
- Format menu, options in** 250
- Format toolbar, buttons on** 261
- function_name Is An Unknown Function**
 - error message 138

G

- Go menu, options in** 249

H

- Help menu, options in** 256
- hidden windows, navigating to** 255
- History toolbar, buttons on** 259
- HTML source code, displaying for templates** 36
- HTML tabs, migrated to Web Tools** 97

I

- identification numbers, adding to repository modifications** 168
- Import Wizard-Review Conflicts and Actions**

- about 229
- Attribute Differences pane in 230
- Conflicting Objects pane in 229
- Object Difference pane in 229
- Inactive property, using** 30

L

- languages**
 - enabling overrides of 19
 - setting modes for 18
- List toolbar, buttons on** 258
- Locale Management Utility**
 - exporting strings and locale properties using 214
 - finding existing translations using 211
 - finding untranslated text strings using 210
 - importing strings and locale properties using 215
 - marking as Redo modified records for project 212
 - modifying strings using 216
 - using 209
 - using commend line to run 216
 - using to find modified objects since export 212
- locale objects, finding untranslated text strings for** 210
- locale properties**
 - configuring non-translatable 205
 - exporting 214
 - importing 215
 - using command line to export 217
- locales**
 - configuring data for 205
 - displaying controls and list columns according to 206

M

- menu bar**
 - Debug menu in 250
 - Edit menu in 245
 - File menu in 244
 - Format menu in 250
 - Go menu in 249
 - Help menu in 256
 - menus on 243
 - Query menu in 249
 - Screen menu in 248
 - Tools menu in 251
 - using to display menus 30
 - View menu in 246
 - Window menu in 255
- migration script, manually running** 97

modifications

testing and troubleshooting 219

modified objects, finding 212**Multi Value Property Window, using** 37**N****New Object wizard**

about 257

using to create objects 146

non-repository configurations and data, migrating 181**non-translatable locale properties, configuring** 205**O****object comparison**

about 153

in archive files 155

in current repository and archive file 154

in different repositories 154

in the same repository 154

using Compare Objects dialog box in 268

object definitions

comparing using Compare Objects dialog box 268

display of 21

locating and modifying in Object List Editor 23

validating using Validate dialog box 266

validating using Validation Options dialog box 267

Object Explorer

overview of 20

using 20

using lists and tabs in 26

object hierarchies, display of 22**Object List Editor**

controlling display of 28

displaying object types in 25

locating and modifying object definitions in 23

modifying multiple records in 24

overview of 20

using 20, 28

using the link in 29

Object List Editor window

List toolbar for 258

refining queries in 249

showing Visualization views in 152

using Edit menu to apply object definitions in 245

using Edit menu to apply objects in 245

using to modify objects 147

object tagging

and using Siebel Remote 165

object types

display of 21

displaying hidden 25

displaying in Object List Editor 25

object wizards, using to create objects 39**objects**

about comparison and synchronization for 153

comparing in archive files 155

comparing in current repository and archive file 154

comparing in different repositories 154

comparing in the same repository 154

copying 147

creating 146

deleting 149

examining 150

finding modified 212

finding modified since export 212

getting documentation about 33

modifying 147

overriding restrictions preventing modification of 167

overview of customizing 145

restricting those developers can modify 165

synchronizing between repositories 155

using archive files to export and import 221

using object wizards to create 39

validating 148

viewing relationships for 152

P**Palettes window**

buttons in 264

using 36

Project drop-down list, using 26**projects**

creating for symbolic strings 191

Properties window, using to display property settings 34**properties, displaying hidden** 25**Q****queries**

about compound 33

about simple 32

running 31

Query menu, options in 249**R****records, determining when Siebel CRM**

- created or updated 157
- Redo, Locale Management Utility marking projects as** 212
- remote databases, updating** 180
- repimexp utility, using to import and export repositories** 183
- repositories**
 - exporting 173
 - exporting and importing 169
 - files you can use to manage 160
 - guidelines for migrating 185
 - identifying conflicts that occur during upgrades of 174
 - importing in UNIX environments 172
 - importing in Windows environments 171
 - importing or exporting later 173
 - managing 159
 - merging 169
 - overview of exporting and importing 169
 - overview of managing 159
 - preparing to migrate 176
 - process of migrating 176
 - renaming 162
 - searching 150
 - troubleshooting incremental merges of 169
 - upgrading 173
 - using Database Configuration Wizard to migrate 177
 - viewing information about current 160
- repository modifications**
 - adding identification numbers to 168
- results.slf, about** 214
- results.txt, about** 214
- right-click menus**
 - about using 30
 - using to hide windows 37
- Running Tool Tip**
 - how it differs from Tool Tips in Script Assist 130
 - using 129

S

- Screen menu, options in** 248
- Script Assist**
 - about using 125
 - accessing the window for 128
 - using script libraries for 130
- script libraries, using with ST eScript Engine** 130
- scripts**
 - development options for 270
 - tracing 136
- Server Script Editor, about using** 44

- Server scripting language, setting default** 121
- ServerEditRecordTimeStamp parameter, setting** 166
- Siebel Application Upgrader, about** 173
- Siebel Compiler, order considerations and error message in** 138
- Siebel Debugger**
 - accessing 134
 - overview of using 132
 - using 132
 - using to display script variables and values 136
 - using to display subroutines and function calls 135
- Siebel eScript**
 - menu options of debugger for 250
 - using Debug toolbar to access debugger for 260
- Siebel Script Editor**
 - about 44
 - opening 122
 - overview of using 119
 - setting options for 121
 - using 119
- Siebel Script Performance Profiler**
 - about window for 139
 - enabling 141
 - overview of using 139
 - running 142
 - using 138
- Siebel Tools**
 - installing 16
 - overview of using 15
 - roadmap for setting up and using 16
- Siebel Tools features**
 - excluded from Web Tools 50
 - included in Web Tools 48
- Siebel Tools objects**
 - excluded from Web Tools 50
 - included in Web Tools 49
- Siebel VB**
 - menu options of debugger for 250
 - using Debug toolbar to access debugger for 260
- Siebel Web Client, automatically opening** 219
- Siebel Web templates**
 - publishing changes to 99
- simple queries, about** 32
- Simulate toolbar, buttons on** 261
- ST eScript Engine**
 - enabling 127
 - overview of using 123

- setting options for 127
 - using 123
 - using Fix and Go with 128
 - status bar, showing and hiding** 41
 - string consolidation utility**
 - running 202
 - using to import consolidated strings 203
 - string conversion utility**
 - running 197
 - using to split export files into smaller files 201
 - string overrides, entering** 196
 - String Reference pick applet, using to set symbolic string reference** 194
 - string references, fixing after upgrade** 208
 - strings**
 - exporting 214
 - finding existing translations for 211
 - finding modified objects since export 212
 - finding multiple properties for 210
 - finding untranslated 210
 - importing 215
 - modifying 216
 - using batch files to consolidate 204
 - using batch files to convert 204
 - using command line to export 217
 - symbolic string reference, using String Reference pick applet to set** 194
 - symbolic strings**
 - configuring 189
 - consolidating 202
 - converting 197
 - creating 192
 - creating a project for 191
 - how Siebel CRM references at run time 202
 - modifying predefined 192
 - modifying the configuration file to support 191
 - modifying to globally update display values 193
 - overview of configuring 189
 - parameters that convert 273
 - syntax, validating for scripts** 134
- T**
- target browser, choosing** 17
 - tasks**
 - accessing WF/Task Editor toolbar for 263
 - comparing different versions of task UIs 156
 - controlling versions of task UIs 18
 - toolbars**
 - about using Configuration Context 263
 - buttons on 257
 - buttons on Debug 260
 - buttons on Edit 257
 - buttons on Format 261
 - buttons on History 259
 - buttons on List 258
 - buttons on Simulate 261
 - buttons on WF/Task Editor 263
 - showing and hiding 40
 - using 40
 - Tools environment configuration**
 - setting debug options in 219
 - setting options for Siebel Script Editor in 121
 - Tools menu, options in** 251
 - translations**
 - exporting strings for 218
 - finding existing 211
 - Types tab, using** 26
- U**
- untranslated text strings**
 - finding 210
 - finding existing translations for 211
- V**
- Validate dialog box, elements of** 266
 - Validate Options dialog box, elements of** 267
 - View Layout Editor, using** 41
 - View menu**
 - options in 246
 - showing Visualization views in 152
 - Visualization views**
 - development options for 270
 - showing 152
- W**
- W property, using** 29
 - Watch window, using to display script variables and values** 136
 - Web browser, defining layout** 263
 - Web Page Layout Editor, using** 41
 - Web template migration process**
 - troubleshooting and verifying the results of 98
 - Web templates, displaying list of** 36
 - Web Templates, using** 36
 - Web Tools**
 - benefits of 48
 - Web Tools about** 47
 - WF/Task Editor toolbar, buttons on** 263
 - Window menu, options in** 255
 - windows**
 - docking and undocking 38

- hiding docked as tabs 38
- resetting 39
- showing and hiding 37
- stacking dockable 38

workflows

- accessing WF/Task Editor toolbar for 263
- comparing different versions of 156
- controlling versions of 18

Workspaces

- enabling 16

Glossary

Advanced Compile

A feature in Siebel Tools that you can use to assist with localization.

Applets window

A window that displays information about a view and allows you to add applets to that view.

Application Deployment Manager (ADM)

A feature that you can use to administer a Siebel CRM deployment.

bookmark

A feature that allows you to return directly to an item.

Bookmarks window

A window that allows you to navigate directly to an object that you use frequently.

breakpoint

A marker in a line of code that stops code from running at that line.

canvas

A background that appears in different designers.

collection function

A type of function that includes a finite set of values.

compound query

A type of query that locates records according to more than one condition.

conflict

A situation that occurs if an object that Siebel Tools attempts to import from an archive does not match the corresponding object in the repository.

Controls/Columns window

A window that displays the controls and list columns that you can add to an applet layout if you use the Applet Layout Editor.

custom object

A new object that you create.

declarative configuration

A type of programming technique that uses objects and object properties in the Siebel repository to implement the logic that your business requires.

developer conflict

An error that occurs if two separate groups or developers update the same object.

full get

A type of Get that copies all projects from the server repository to your local repository.

get

The act of copying a project from the server repository to your local repository.

hidden object type

An object type that Siebel CRM does not show in the Siebel Web Client.

hotfix

A Siebel Tools feature you can use to quickly update the production environment.

Incremental Repository Merge

A feature that allows you to merge multiple repositories and apply patches.

language mode

A mode that allows you to configure Siebel CRM to display text in a language other than English.

language override

A nontranslatable locale property that you can configure differently for different locales.

Locale Management Utility (LMU)

A utility in Siebel Tools that you can use to manage how you configure Siebel CRM to localize text strings, such as field labels, and other locale properties, such as the height and width of controls.

mid-level release

A type of release that includes objects you modify from a time frame that you specify.

modified object

A predefined or custom object that you modify.

new object wizard

A feature you can use that guides you through the steps of creating a new object.

object definition

Implements one piece of the software. It consists of object properties, which are characteristics of this piece of software.

Object Explorer

A window in Siebel Tools that displays the Siebel object hierarchy.

Object List Editor

A window in Siebel Tools that displays the object definitions of the object type that you choose in the Object Explorer.

object property

A characteristic of a piece of software.

object tagging

A version control feature that Siebel Tools uses to associate a repository modification with a tag or group. You can use it to export all the work that a group of developers performs.

object type

An entity that includes a predefined set of properties.

Palettes window

A window that allows you to add items to an object.

parent and child relationship

A type of hierarchical relationship between one object type and another object type.

predefined object

An object that comes already defined when you first install Siebel CRM or Siebel Tools.

predefined symbolic string

A string that comes predefined with Siebel CRM.

prior standard repository

The repository that comes predefined with Siebel CRM. Siebel Tools uses it to determine if you modified any object in the repository since the prior release.

project

An object type that your development team can use to help make sure only one developer works on an object at one time.

property value

Information that you enter into the column of an object definition.

Properties window

A window that displays the properties and the property value for the object that you choose in the Object List Editor.

pseudolocalization prefix

A prefix that Siebel Tools uses to test the appearance of string in a language.

reference repository

A prior version of the repository.

right-click menu

A type of context-sensitive menu that appears if you right-click an object or an element.

Running Tool Tip

A help feature in the Siebel Script Editor that you can use to write a script.

script library

A part of the ST eScript Engine that allows you to call a business service method from a script.

Siebel Application Upgrader

A utility you can use to get new features from the latest software release while preserving the custom configuration you created in the current repository.

Siebel Delta File (SDF)

A type of file that is similar to an SIF file except that an SDF file contains only information about modifications to an object.

Siebel Repository

A set of tables that includes Siebel objects and server scripts.

Siebel repository patch file (SPF)

A type of file that includes exported objects.

Siebel Script Editor

An editor that allows you to create and maintain scripts that use Siebel VB, Siebel eScript, or Browser Script.

Siebel Script Performance Profiler

A part of the ST eScript Engine that allows you to observe and monitor the performance of a script.

Siebel Tools

An integrated development environment that you can use to configure Siebel CRM.

simple query

A type of query that locates records according to one condition.

symbolic string

An object that you can use to store the value of a string.

symbolic string reference

A reference to a symbolic string.

top-level object type

An object type at the start of the object hierarchy.

Touch

A version control feature in Siebel Tools that allows you to tag an object even if you do not modify this object.

translatable string

A type of string that Siebel CRM can translate to another language.

Type deduction

A feature of the ST eScript Engine that determines the type of local variables that a script uses.

Validate Tool

An error correction tool you can use to validate the semantic consistency of an object.

Web template editor

An application external to Siebel Tools that allows you to edit a Web template.

Web Template Explorer window

A window that displays the HTML code of a Siebel Web Template.

A

Reference Materials for Siebel Tools

This appendix includes reference information for Siebel Tools. It includes the following topics:

- [Menus and Menu Items on the Menu Bar on page 243](#)
- [Buttons on the Toolbars on page 257](#)
- [Buttons in the Palettes Window on page 264](#)
- [Dialog Boxes That Validate Objects on page 265](#)
- [Dialog Boxes That Compare Objects on page 268](#)
- [Dialog Boxes That Set Development Options on page 269](#)
- [Parameters That Convert Symbolic Strings on page 273](#)

Menus and Menu Items on the Menu Bar

This topic describes the menus and menu items that you can click from the menu bar in Siebel Tools. For more information, see [“Using the Menu Bar” on page 30](#).

This topic includes the following information:

- [File Menu on page 244](#)
- [Edit Menu on page 245](#)
- [View Menu on page 246](#)
- [Screens Menu on page 248](#)
- [Go Menu on page 249](#)
- [Query Menu on page 249](#)
- [Format Menu on page 250](#)
- [Debug Menu on page 250](#)
- [Tools Menu on page 251](#)
- [Workspace Menu on page 254](#)
- [Help Menu on page 256](#)

File Menu

Table 20 describes the menu items that you can click from the File menu. To use this menu, you click the File menu on the Menu Bar.

Table 20. Menu Items That You Can Click from the File Menu

Menu Item	Description
Open Repository	If multiple repositories reside in the development database, then this menu item allows you to open a repository other than the repository that is currently open. Siebel Tools sets the repository you choose from the Open Repository menu item as the default repository that it opens each time you open Siebel Tools.
New Object	Starts the New Object Wizard that allows you to create a list applet, form applet, chart applet, tree applet, business component, report, table, command, picklist, MVG, or view.
Close (CTRL+F4)	Closes the Object List Editor.
Save (CTRL+S)	Saves modifications that the current editing window contains if you edit in one of the following windows: Layout, Menu, or Basic Scripts.
Save All	Saves modifications in all open editing windows.
Import	Imports text from an external text file into the Siebel VB Editor window. This text must use an SBL file format. Siebel Tools creates this format when it exports text from the Siebel VB editor.
Export	Allows you to create a text file in delimited or HTML format that lists the property values of an object or all objects that the Object List Editor currently shows.
Print Setup	Modifies the printer and printing options for printing diagrams from the object visualization view.
Print Preview	Opens a print preview window that displays an object visualization view.
Print (CTRL+P)	Prints the active object visualization view diagram.
Exit	Closes Siebel Tools.

Edit Menu

Table 21 describes the menu items that you can click from the Edit menu. The Edit menu items apply to individual objects in the Object List Editor. You can right-click an object in the Object List Editor to display a list of menu items. For more information, see [“Using a Layout Editor” on page 41](#).

Table 21. Menu Items That You Can Click from the Edit Menu

Menu Item	Description
Undo (CTRL+Z)	Reverses the last modification you made to a property value in the Object List Editor or the Property window before you save the object.
Redo (CTRL+Y)	Reapplies modifications after the Undo command runs.
Undo Delete	Siebel Tools displays this menu item if you delete a record in the Object List Editor. It allows you to undo the deletion.
Undo Record	If you have not saved the record, then this menu item removes a new object that you create or reverses modifications you made to an existing object.
New Record (CTRL+N)	Creates a new object in the Object List Editor and positions the cursor in the first required property.
Copy Record (CTRL+B)	Creates a new object that is a copy of the object that you choose. It also creates copies of all child objects. It is recommended that you use the Copy Record menu item only if reusing an existing object is not practical.
Delete Record (CTRL+D)	Deletes the object that you choose and the child objects of the object you choose. It is recommended that you do not use the Delete Record menu item. Instead, use the Inactive property. For more information, see “Using the Inactive Property” on page 30 .
Cut (CTRL+X)	If you use this menu item while your cursor is in a property that contains text, then Siebel Tools copies the text you choose to the clipboard and deletes the existing text. In the Applet Designer, it copies the control you choose to the clipboard and deletes the existing control.
Copy (CTRL+C)	If you use this menu item while your cursor is in a property that contains text, then Siebel Tools copies the text you choose to the clipboard. In the Applet Designer, it copies the control you choose to the clipboard.
Paste (CTRL+V)	Inserts text from the clipboard into a property at the insertion point. Inserts a control from the clipboard in the Applet Designer.
Delete (DEL)	If you use this menu item while your cursor is in a property that contains text, then Siebel Tools deletes the text you choose. In the Applet Designer, it deletes the control you choose.
Select All (CTRL+A)	Chooses all items. In the Applet Designer, it chooses all controls that the applet contains.
Change Records	Modifies multiple records simultaneously.

Table 21. Menu Items That You Can Click from the Edit Menu

Menu Item	Description
Find (CTRL+F)	Finds the text that you specify in the Siebel Script Editor window.
Replace (CTRL+H)	Replaces the text that you specify with different text in the Siebel Script Editor window.

View Menu

Table 22 describes the menu items that you can click from the View menu. You use these menu items to display windows, toolbars, or visualization views.

Table 22. Menu Items That You Can Click from the View Menu

Menu Item	Subitem	Description
Windows	Palette	Displays the Palettes window.
	Properties Window	Displays the Properties window.
	Applets Window	Displays the Applets window.
	Controls Window	Displays the Controls/Columns window.
	Bookmarks Window	Displays the Bookmarks window.
	Web Templates Window	Displays the Web Templates Explorer window.
	Multi Value Properties Window	Displays the Multi Value Property Window.
	Refresh Windows	Requeries and updates the state of dockable windows.
	Reset Windows	Closes all dockable windows except the Object Explorer for the currently active editor. Does not close editor windows.
Editors	Web Applet Editor	Opens the applet you choose in the Applet Layout Editor, including the Controls/Columns and Palettes windows.
	Server Script Editor	Opens the Siebel Script Editor. You can specify the editor to use or you can use the default.
	Browser Script Editor	Opens the Siebel Web Script Editor that you use to access the scripts that control the presentation and behavior of applet controls and list columns in a Web applet template.

Table 22. Menu Items That You Can Click from the View Menu

Menu Item	Subitem	Description
Visualize	View Details	For more information, see “Viewing Object Relationships” on page 152.
	View Relationships	
	View Descendents	
	View Web Hierarchy	
Debug Windows	Calls (CTRL+L)	Opens the Calls window. This window displays the call stack of the Siebel VB script or the Siebel eScript script that you are currently debugging.
	Watch (SHIFT+F9)	Opens the Watch window. This window displays the values of local variables for items you are currently debugging, such as Siebel VB script, Siebel eScript, or Siebel Workflow.
	Errors	Opens the Errors window. This window displays the runtime errors in the Siebel VB script or Siebel eScript script that you are currently debugging.
Preview	Not applicable	Displays a preview of a Web view layout. This preview approximates how Siebel CRM displays the container page, screen bar, and view bar.
ActiveX Methods		Allows you to view the methods for the current ActiveX control in the Applet Designer.
Toolbars		Displays the following toolbars: Edit, History, List, Debug, Web Controls, and Configuration Context.
Status Bar		Displays the Status bar at the end of the Siebel Tools window.
Object Explorer (CTRL+E)		Displays the Object Explorer.
Options		Opens the Development Tools Options dialog box. For more information, see “Development Options for Debugging” on page 272.

Screens Menu

Table 23 describes the menu items that you can click from the Screens Menu. Siebel Tools displays the Screens menu only if you log on to Siebel Tools as a system administrator.

Table 23. Menu Items That You Can Click from the Screens Menu

Menu Item	Subitem	Description
Application Upgrader	Application Upgrade Object List	Displays the Application Upgrades list, Object Differences list, or Attribute Differences list in the Object List Editor. In the Attribute Differences list, you can right-click and select an option to show critical conflicts, non-critical conflicts, and all changes for the item selected in the Object Differences list.
	Application Upgrade Database Version	For internal Oracle use.
	Application Upgrade Attribute List	Displays the Application Upgrades list and the Attribute Differences list in the Object List Editor. In the Attribute Differences list, you can right-click and select an option to show critical conflicts, non-critical conflicts, and all changes for the item selected in the Application Upgrades list.
System Administration	System Preferences	Displays system preferences in the Object List Editor. This information is similar to the information that Siebel CRM shows in the System Preferences view in the Administration - Application screen in the Siebel client. NOTE: In Web Tools, click the Tools menu and then click the System Preferences menu item.
	Analytics Strings	For internal Oracle use.
	List of Values	Displays the lists of values that the development database contains.

Go Menu

Table 24 describes the menu items that you can click from the Go menu. The Go menu allows you to navigate records in a list.

Table 24. Menu Items That You Can Click from the Go Menu

Menu Item	Description
Back	Displays the previous screen in a sequence.
Forward	Displays the next screen in a sequence.
Previous Record (CTRL+UP)	Navigates to the first object that is preceding the current selection.
Next Record (CTRL+DOWN)	Navigates to the first object that is succeeding the current selection.
First Record (CTRL+PAGE UP)	Navigates to the first object in the list.
Last Record (CTRL+PAGE DOWN)	Navigates to the last object in the list.
Add Bookmark	Displays the Add Bookmark dialog box that allows you to create a bookmark to the object you choose. You can use this menu item to create a <i>bookmark</i> , which is a feature that allows you to return directly to an item.
Bookmark List	Displays the Bookmarks dialog box that allows you to choose an existing bookmark. You can also use this dialog box to rename or delete an existing bookmark.

Query Menu

Table 25 describes the menu items that you can click from the Query menu. This menu allows you to create and refine an Object List Editor query. You can use this query to filter the list of objects that appear in the current Object List Editor.

Table 25. Menu Items That You Can Click from the Query Menu

Menu Item	Description
New Query (CTRL+Q)	Allows you to filter the set of objects that appear in the Object List Editor.
Refine Query (CTRL+R)	Allows you to add more filters to the current query.

Table 25. Menu Items That You Can Click from the Query Menu

Menu Item	Description
Execute Query (ENTER)	Runs the query.
Sort Order	Opens the Sort Order dialog box. This dialog box allows you to specify the order that the Object List Editor uses to display the records.

Format Menu

Table 26 describes the menu items that you can click from the Format menu. This menu in the Applet Layout Editor allows you to position controls, configure the grid, and adjust tab or list column order.

Table 26. Menu Items That You Can Click from the Format Menu

Menu Item	Description
Align	Aligns the items you choose.
Make Same Size	Makes all items you choose the same size.
Horizontal Spacing	Adjusts horizontal spacing between items.
Vertical Spacing	Adjusts vertical spacing between items.
Center in Applet	Centers the items you choose horizontally or vertically.
Set Label Alignment	Aligns labels in applets for Web templates that use a grid layout.
Set Tab Order	Sets the tab order for fields in a form applet. This item is not available for list applets.

Debug Menu

Table 27 describes the menu items that you can click from the Debug menu. This menu allows you to control the debugger for Siebel VB or Siebel eScript.

Table 27. Menu Items That You Can Click from the Debug Menu

Menu Item	Description
Check Syntax	Compiles the current script and verifies syntax.
Start (F5)	Starts the Siebel client. It also displays a dialog box that includes startup parameters.
Break (CTRL+BREAK)	Stops the script that is currently running. If Siebel VB or Siebel eScript is not running, then Siebel Tools does nothing.
End	Stops the Siebel client and returns to the Siebel Script Editor window.

Table 27. Menu Items That You Can Click from the Debug Menu

Menu Item	Description
Restart (SHIFT+F5)	Restarts the Siebel client if a break occurs.
Toggle Breakpoint (F9)	Sets or removes a breakpoint on a line of code.
Clear All Breakpoints (CTRL+SHIFT+F9)	Removes all breakpoints from the current script.
Watch (SHIFT+F9)	Displays script variables and their values. For more information, see “Using the Watch Window While it Monitors a Script” on page 136 .
Calls (CTRL+L)	Contains a list of subroutine and function calls that Siebel Tools runs prior to the current line. If you choose an item in this list, then the interpreter shifts to this item.
Step Into (F8)	Runs the next line of a script. If this line calls a subroutine or procedure, then Siebel Tools runs this subroutine or procedure.
Step Over (SHIFT+F8)	Runs the first line of script that occurs after the current subroutine or procedure. Siebel Tools continues to run this script in the current subroutine or procedure.
Step To Cursor (CTRL+F8)	Runs all lines of code up to the line that includes the cursor.

Tools Menu

Table 28 describes the menu items that you can click from the Tools menu.

Table 28. Menu Items That You Can Click from the Tools Menu

Menu Item	Subitem	Description
Compile (F7)		Opens the Object Compiler dialog box to compile one or more projects to a repository.
Compile Selected Objects (CTRL+F7)		Opens the Object Compiler dialog box to compile the objects you choose to a repository.
Lock Project (ALT+L)		Locks the project that the object you choose references.
Unlock Project (ALT+U)		Unlocks the project that the object you choose references.
Add To Archive		Opens the Export To Archive dialog box to add the top-level objects that you choose or projects to an archive.

Table 28. Menu Items That You Can Click from the Tools Menu

Menu Item	Subitem	Description
Import From Archive		Starts the Import wizard to import objects from an archive.
Compare Objects	Selected	Compares two objects that you choose. It uses a list of object names and properties to display similarities and differences.
	Selected vs. Repository	Compares the object you choose to the corresponding object in the repository and displays similarities and differences.
	Selected vs. Archive	Compares the object you choose to the corresponding object in an archive and displays similarities and differences.
	Archive vs. Archive	Compares two files that you choose and displays similarities and differences.
Convert to Grid Layout		Converts a form applet that uses a nongrid layout to grid layout.
Search Repository		Opens the Search Repository dialog box to search for objects according to the object name, another property, or the object type.
Validate Object		Validates the object you choose. Lists errors according to severity, rule number, object name, and error description. Allows you to modify options for rules, severity, and enforcement.

Table 28. Menu Items That You Can Click from the Tools Menu

Menu Item	Subitem	Description
Upgrade	Maintenance Update	Not applicable starting with Siebel CRM version 8.0.
	Prepare Repository	Used to upgrade from a Siebel CRM version that occurs before version 7.x to version 8.0. The Prepare Repository utility runs before it does a repository merge. It migrates strings from the S_MSG table, merges labels and fields, and merges templates to applets for the language that you specify. For more information, see <i>Siebel Database Upgrade Guide</i> .
	Migrate ICL Objects to Standard	Applicable if you choose the Incorporate Custom Layout (ICL) option. You choose this option to preserve the layouts of custom objects during a previous upgrade. Before you can perform a subsequent upgrade, you must migrate the ICL objects to the predefined repository. For more information, see “About Predefined Objects” on page 23 and <i>Siebel Database Upgrade Guide</i> .
	Upgrade Application	Displays the Application Objects Upgrade List in the Application Upgrader screen of Siebel Tools and opens the Merge Repositories dialog box. You use this menu item to merge predefined and custom repositories. For more information, see <i>Siebel Database Upgrade Guide</i> .
	Generate EIM Processing Columns	Opens the EIM Processing Column Generator dialog box. You can use this dialog box to create missing EIM processing columns and indexes after you merge the repository.
	Web Client Migration	Used to upgrade from Siebel CRM version 6.x to version 7.x or version 8.0. It associates Web templates to a group of applets and views so that Siebel CRM can use them in the Siebel client. For more information, see <i>Siebel Database Upgrade Guide</i> .

Table 28. Menu Items That You Can Click from the Tools Menu

Menu Item	Subitem	Description
Utilities	Generate Help IDs	Oracle uses this subitem internally to create the sshelp.hm file for Siebel Tools Online Help. This file includes information about context Id numbers and text help identifiers that the Help Id objects specify.
	Locale Management	Allows you to use the Local Management Utility to import or export translatable strings and locale properties.
	Map Fax Properties	If you click business component in the Object Explorer, then this menu item opens the Map Fax Properties dialog box for the business component that you choose. You can use this dialog box to create mappings between fields in the business component and sheet properties for the fax software. These mappings allow you to customize the fax cover sheet and the fax message.
	Export View Previews	Exports the view that the Preview mode of the View Layout Editor shows to an HTML file.
	Case Insensitivity	Opens the Case and Accent Insensitivity Wizard. It allows you to do a case-insensitive or accent-insensitive search on columns in the Siebel schema. For more information, see <i>Configuring Siebel Business Applications</i> and <i>Siebel Database Upgrade Guide</i> .
	Build Patch	Starts the Patch Builder wizard that allows you to create a patch file.
	Apply Patch	Opens the Apply Patch window that allows you to apply the patch.

Workspace Menu

When you select the Workspace menu, a list of different menu items that are used to configure and manage workspaces appears. [Table 30](#) describes the menu items under the Workspace menu.

Table 29. Menu Items That You Can Click from the Workspace Menu

Menu Item	Description
Create	Select this option to create a new workspace. For more information, see “Creating New Workspaces” on page 56 .
Checkpoint	Select this option to check in the changes that you made to the current version of the workspace. For more information, see “Performing the Checkpoint Version Process” on page 61 .

Table 29. Menu Items That You Can Click from the Workspace Menu

Menu Item	Description
Revert	Select this option to revert the changes that you made to the current version of that workspace. For more information, see “Reverting to Previous Workspace Versions” on page 62 .
Rebase	Select this option to apply and up-take the changes that were made in the parent workspace into the current workspace. For more information, see “Rebasing Workspaces” on page 62 .
Merge Reports	Select this option to view and resolve the conflicts that occurred during the rebase process. For more information, see “Rebasing Workspaces” on page 62 and “Detecting Conflicts in Workspaces and Applying Resolutions” on page 64 .
Submit for Delivery	Select this option to change the status of the workspace and make it ready for delivering the changes to the MAIN workspace. For more information, see “Submitting Workspaces for Delivery” on page 67 .
Deliver	Select this option to deliver the changes in the workspace to the MAIN workspace. NOTE: This option is available only to the user who is also the owner of the MAIN workspace. For more information, see “Delivering Workspaces” on page 69 .
Compare	Select this option to view the differences that are made to the objects in two selected workspace versions. For more information, see “Comparing Workspace Versions” on page 70 .
Undo Submit for Delivery	Select this option to cancel the workspace delivery process. For more information, see “Canceling the Workspace Delivery Process” on page 82 .
Workspace Explorer	Select this option to display the Workspace explorer pane.

Window Menu

The Window menu lists the currently open Object List Editor, Application Designer, visualization view, and other windows, and allows you to navigate to windows that Siebel Tools does not currently show. If one of these windows is open, then Siebel Tools displays the Close menu item as the first item. The Close menu item closes the window that is currently active.

Help Menu

Table 30 describes the menu items that you can click from the Help menu.

Table 30. Menu Items That You Can Click from the Help Menu

Menu Item	Description
Contents	Opens the Siebel Tools Online Help.
Using Help	
Technical Support	Displays the Technical Support Information dialog box that includes information that Technical Support might require, such as the version number of your Siebel Tools installation.
About Record	Opens a dialog box that displays information about the current object, including the object creator and creation date.
About SRF	Opens a dialog box that displays information about the most recent full incremental compile.
About View	Opens a dialog box that displays information about the current screen, business object, and view, including applet layout.
About Visible Views	Displays the list of views in the repository and if each view is visible. Visibility for each view depends on the license key you use when you install Siebel Tools. For more information, see “Description of the About Visible Views Dialog Box” on page 256 .
About Siebel Tools	Opens a dialog box that identifies the version of Siebel Tools.

Description of the About Visible Views Dialog Box

Table 31 describes the information that the About Visible Views dialog box shows. To view this dialog box, you click the Help menu and then click About Visible Views.

Table 31. Description of the About Visible Views Dialog Box

Column	Description
View Name	Displays the view name. The views in this column are not the predefined views in Siebel Tools.
Visible	Shows whether or not the view is visible in Siebel Tools.
Application	Shows whether or not the view is associated with Siebel Tools.
View	Shows whether or not the view always appears as a view in Siebel Tools.
Responsibility	Shows whether or not the view is associated with user responsibilities.

Table 31. Description of the About Visible Views Dialog Box

Column	Description
License	Shows whether or not the view visibility depends on the license key that you use when you install Siebel Tools.
Platform	Shows whether or not the view visibility depends on the platform that Siebel Tools uses.

Buttons on the Toolbars

This topic describes the buttons that you use on the toolbars in Siebel Tools. For more information, see [“Using the Toolbars” on page 40](#).

This topic includes the following information:

- [Edit Toolbar on page 257](#)
- [List Toolbar on page 258](#)
- [History Toolbar on page 259](#)
- [Debug Toolbar on page 260](#)
- [Simulate Toolbar on page 261](#)
- [Format Toolbar on page 261](#)
- [WF/Task Editor Toolbar on page 263](#)
- [Configuration Context Toolbar on page 263](#)

Edit Toolbar

[Table 32](#) describes the buttons that you can click on the Edit toolbar.

Table 32. Buttons You Can Click on the Edit Toolbar









Button	Description	
	New	Starts the New Object Wizard that allows you to create applets, views, charts, and other objects.
	Save	Saves modifications in the current editing window if you use Layout, Menu, or Basic Scripts.
	Save All	Saves modifications in all open editing windows.

Table 32. Buttons You Can Click on the Edit Toolbar

Button	Description	
	Cut	If you use this button while the cursor is in a property that contains text, then Siebel Tools copies the text you choose to the clipboard and deletes the existing text. In the Applet Designer, it copies the control you choose to the clipboard and deletes the existing control.
	Copy	If you use this button while the cursor is in a property that contains text, then Siebel Tools copies the text you choose to the clipboard. In the Applet Designer, it copies the control you choose to the clipboard.
	Paste	Inserts text from the clipboard to a text property at the insertion point. In the Applet Designer, it inserts a control from the clipboard.
	Undo	If you have not saved the object, then this button reverses the last modification you made to a property value in the Object List Editor or Property window.
	Redo	Reapplies modifications after the Undo command runs.

List Toolbar

Table 33 describes the buttons that you can click on the List toolbar.

Table 33. Buttons You Can Click on the List Toolbar











Button	Description	
	Add New Record	Creates a new object in the Object List Editor and positions the cursor in the first required property.
	First Record	Goes to the first object in the list.
	Previous Record	Goes to the object preceding the current selection.
	Next Record	Goes to the object succeeding the current selection.
	Last Record	Goes to the last object in the list.
	New Query	Allows you to specify one or more filters on the set of objects that the Object List Editor shows.





Table 33. Buttons You Can Click on the List Toolbar

Button	Description	
	Execute Query	Runs the query you specify. This button does the same as pressing ENTER.
	Sort Ascending	Modifies the order that Siebel Tools uses to display objects. It sorts them in ascending order according to the currently chosen property column.
	Sort Descending	Modifies the order that Siebel Tools uses to display objects. It sorts them in descending order according to the currently chosen property column.
	Filter Version	Displays only the most recent version of each workflow process or task UI that the Object List Editor shows.

History Toolbar

Table 34 describes the buttons that you can click on the History toolbar.








Table 34. Buttons You Can Click on the History Toolbar

Button	Description	
	Go Back	Returns to the previously displayed screen.
	Go Forward	Returns to the subsequent displayed screen.
	Add Bookmark	Displays the Add Bookmark dialog box that allows you to create a bookmark to the object you choose. You can use this menu item to create a <i>bookmark</i> , which is a feature that allows you to return directly to an item.
	Bookmark List	Displays the Bookmarks dialog box that allows you to choose an existing bookmark. You can also use this dialog box to rename or delete an existing bookmark.

Debug Toolbar

Table 35 describes the buttons that you can click on the Debug toolbar.





Table 35. Buttons You Can Click on the Debug Toolbar

Button	Description	
	Check Syntax	Compiles the current script and verifies syntax.
	Start	Starts the Siebel client. It also displays a dialog box that includes startup parameters.
	Break	Stops the script that is currently running. If Siebel VB or Siebel eScript is not running, then Siebel Tools does nothing.
	End	Stops the Siebel client and returns to the Siebel Script Editor window.
	Toggle Breakpoint	Sets or removes a breakpoint on a line of code.
	Watch	Displays script variables and their values. For more information, see “Using the Watch Window While it Monitors a Script” on page 136 .
	Calls	Contains a list of subroutine and function calls that Siebel Tools runs prior to the current line. If you choose an item in this list, then the interpreter shifts to this item.
	Step Into	Runs the next line of a script. If this line calls a subroutine or procedure, then Siebel Tools runs this subroutine or procedure.
	Step Over	Runs the first line of script that occurs after the current subroutine or procedure. Siebel Tools continues to run this script in the current subroutine or procedure.

Simulate Toolbar

Table 36 describes the buttons that you can click on the Simulate toolbar.

Table 36. Buttons You Can Click on the Simulate Toolbar

Button	Description	
	Start Simulation	Starts the simulation of a workflow process.
	Simulate Next	Simulates the next workflow process step.
	Complete Simulation	Completes the simulation of a workflow process.
	Stop Simulation	Stops the Workflow Simulator.

Format Toolbar

Table 37 describes the buttons that you can click on the Format toolbar.

Table 37. Buttons You Can Click on the Format Toolbar








Button	Description
	Aligns the near edges of controls.
	Aligns the centers of controls along a vertical axis.
	Aligns the far edges of controls.
	Aligns the highest points of controls.
	Aligns the middles of controls along a horizontal axis.
	Aligns the lowest points of controls.
	Makes the controls the same width.





Table 37. Buttons You Can Click on the Format Toolbar

Button	Description
	Makes the controls the same height.
	Makes the controls the same size.
	Makes the horizontal spacing between controls equal.
	Increases the horizontal spacing between controls.
	Decreases the horizontal spacing between controls.
	Removes the horizontal spacing between controls.
	Makes the vertical spacing between controls equal.
	Increases the vertical spacing between controls.
	Decreases the vertical spacing between controls.
	Removes the vertical spacing between controls.
	Centers the controls vertically.
	Centers the controls horizontally.
	Aligns the labels to the near margin.
	Centers the labels.
	Aligns the labels to the far margin.

WF/Task Editor Toolbar

Table 38 describes the buttons that you can click on the WF/Task Editor toolbar.

Table 38. Buttons You Can Click on the WF/Task Editor Toolbar

Button	Description	
	Publish/Activate	Publishes and activates a workflow process or task UI in a single step. This button is available only with the Siebel Web Client. You cannot use it to activate a workflow process or task UI in the production environment.
	Publish	Makes a workflow process or task UI available to activate from the Siebel client.
	Revise	Revises a workflow process or task UI.
	Expire	Makes a workflow process or task UI inactive.

Configuration Context Toolbar

Table 39 describes the items that you can use on the Configuration Context toolbar.

Table 39. Items You Can Use on the Configuration Context Toolbar

Drop-Down List	Description
Target Browser	Allows you to choose a target browser for layout editing and for scripting.
Application	Allows you to configure objects for a specific Siebel application. Typically, you use All Applications. If you choose a single application from the list, then you can configure objects, such as applets or views, to show or behave differently for only the application you choose.
Variable	Allows you to specify a display style for an applet for previewing, such as parent, child, or grandchild. Siebel Tools might display an applet differently depending on the underlying Web template. For example, an applet header might not appear if Siebel Tools displays it as a grandchild.

Buttons in the Palettes Window

Table 40 describes the buttons that the Palettes window shows when you use the Applet Layout Editor. For more information, see [“Using the Palettes Window” on page 36](#).

Table 40. Buttons the Palettes Window Shows with the Applet Layout Editor























Button	Description
	CheckBox. Creates an option.
	RadioButton. Creates a radio button.
	MiniButton. Creates a mini button.
	Field. Creates a field.
	FieldLabel. Creates a field label.
	ComboBox. Creates a combo box.
	RecNavNxt. Creates a control for navigating to the next record.
	RecNavPrv. Creates a control for navigating to the previous record.
	Text. Creates a text box.
	TextArea. Creates a text area.
	FormSection. Creates a section of a form.
	Hidden. Creates hidden HTML.
	Password. Creates a text box where the user enters a password during logon.
	Link. Creates an HTML link control.

Table 40. Buttons the Palettes Window Shows with the Applet Layout Editor

Button	Description
	MailTo. Creates a mail-to link.
	Button. Creates a button.
	Label. Creates a label on templates.
	URL. Creates a link to an external URL on a template.
	ActiveX. Creates an ActiveX control on a template.
	Text List Column. Creates a list column that contains HTML text. Available only for a list applet.
	Checkbox List Column. Creates a list column that contains HTML options. Available only for a list applet.
	Custom Control. Creates a custom control on a template. To create the custom control, you can choose a custom control from the Control Type drop-down list, and then move the Custom Control button to the designer.

Dialog Boxes That Validate Objects

This topic describes dialog boxes that you use to validate objects. For more information, see [“Validating Objects” on page 148](#).

This topic includes the following information:

- [Elements of the Validate Dialog Box on page 266](#)
- [Elements of the Validation Options Dialog Box on page 267](#)

Elements of the Validate Dialog Box

Table 41 describes the elements of the Validate dialog box.

Table 41. Elements of the Validate Dialog Box

Element	Description
Errors List	<p>Displays the results of the validation process. Each row in the list identifies a rule violation for an object. You can do the following:</p> <ul style="list-style-type: none"> ■ Double click the error in the Errors list to drill down on the object that causes the error. ■ Click a column heading to sort the rows. ■ Resize the border of the heading cell to adjust columns.
Severity Column	<p>Includes an icon that indicates one of the following:</p> <ul style="list-style-type: none"> ■ Warning. Yellow icon with an exclamation mark. ■ Error. Red icon with a minus sign. This error might cause a problem in Siebel CRM at run time.
Rule Column	Displays an integer that identifies the number of the violated rule. Siebel Tools lists rules sequentially according to the rule number.
Object Column	Displays the name of the object that failed validation.
Description Column	Displays a description of the error or warning.
Details Window	Displays more information about the error or warning message for the row that you choose in the Errors list.
Go To Button	To drill down on the object that causes an error, you can choose an error in the Error section, and then click Go To. You can also double-click the error message.
Log File Window	Displays the path and file name of the log file that includes the same list of validation errors and warnings that the Errors List shows. To save this list as a log file, you can click Save As, navigate to where you want to save the file, and then specify a file name.
Load Button	Loads the contents of the log file that Siebel Tools saves when you use Save As in the Log File section. You can use the Load button to load the list of error and warning validations back into the Validation Tool at a later time.
Save As Button	Saves the list of validation rows that Siebel Tools currently shows in the Errors List as a log file.

Elements of the Validation Options Dialog Box

Table 42 describes the Rules area of the Validation Options dialog box. If you click Options in the Validate dialog box, then Siebel Tools displays the Validation Options Dialog box. To avoid validating objects that your configuration does not use, you must use the repository validator only in conjunction with the time filter.

Table 42. Elements of the Rules Area of the Validation Options Dialog Box

Element	Description
Rules List	Lists the rules that Siebel Tools enforces during validation. You can click a column heading to sort the rows. You can resize the heading cell to adjust columns.
Severity Column	Displays an icon that indicates one of the following: <ul style="list-style-type: none"> ■ Warning. Yellow icon with an exclamation mark. ■ Error. Red icon with a minus sign. This error can cause a problem in Siebel CRM at run time.
Rule Column	Displays an integer that identifies the number of the violated rule. Siebel Tools lists rules sequentially according to the rule number.
Object Column	Displays the object type that Siebel Tools examines.
Description Column	Displays a description of the rule.
Enforce Column	Indicates if Siebel Tools enforces the rule. A Yes value validates all objects of the object type that the Object column identifies. <p>If you use any of the following buttons in the Validation Options dialog box, then Siebel Tools modifies the value in the Enforce column:</p> <ul style="list-style-type: none"> ■ Enforce ■ Ignore ■ Enforce All ■ Ignore All
Save Button	Saves all the values for the current set of rules to a text file that you specify. If you press ENTER, and if the Validation Options dialog box is open, then Siebel Tools saves other settings to a preferences file.
Enforce Button	Modifies the value in the Enforce column in the row that you choose from No to Yes.
Ignore Button	Modifies the value in the Enforce column in the row that you choose from Yes to No.
Enforce All Button	Modifies all values in the Enforce column to Yes.
Ignore All Button	Modifies all values in the Enforce column to No. If you click Ignore All, then Siebel Tools does not validate any objects.

Table 42. Elements of the Rules Area of the Validation Options Dialog Box

Element	Description
Details Text Box	Displays more information about the error or warning message for the row that you choose in the Errors list.
Last Validated Option	If this check box includes a check mark, then Siebel Tools validates only objects that Siebel CRM modified since the date you enter in the Last Validated window.
Custom Option	If this check box includes a check mark, then Siebel Tools validates only the objects that Siebel CRM modified according to the date range that you enter in the date and time fields next to the Custom option.
Do Not Report Warnings Option	If this check box includes a check mark, then Siebel Tools reports only errors. It does not report warnings. It also modifies the value in the Enforced column of all warning rules to No.
Abort Validation Option	<p>If this check box includes a check mark, and if you enter a number in the Errors window that appears after this option, then it stops validating after it reaches the number of errors you specify.</p> <p>If Siebel Tools identifies the number of errors that you specify in this window, then it stops validating the object and displays the Error dialog box.</p>

Dialog Boxes That Compare Objects

This topic describes the dialog boxes that you use to compare objects.





Elements of the Compare Objects Dialog Box

Table 43 describes the elements of the Compare Objects dialog box.

Table 43. Elements of the Compare Objects Dialog Box

Element	Description
First Selection Section	<p>Displays the object hierarchy as a tree. To expand a tree, you can use the controls in the First Selection window or the Second Selection window. For example, if you expand the tree in the First Selection window, then Siebel Tools does the following:</p> <ul style="list-style-type: none"> ■ Expands the tree in the First Selection window. ■ Expands the tree in the Second Selection window. ■ Displays the child object types that each parent object contains. ■ Displays a dashed line to represent a child object that does not exist in an object.
Second Selection Section	

Table 43. Elements of the Compare Objects Dialog Box

Element	Description
Properties Section	Displays the properties that are different for the objects that Siebel Tools compares.
Display Section	Determines how Siebel Tools displays items in the Compare Objects dialog box. You can choose the following options: <ul style="list-style-type: none"> ■ Show All Objects. Displays all child objects in the First Selection section and the Second Selection section. ■ Show All User Properties. Displays all user properties in the Properties section. ■ Show System Properties. Displays system properties in the Properties section (for example Created, Created By, Updated, and Updated By).
	Synchronizes objects from the repository that the First Selection section represents with the repository that the Second Selection section represents. For more information, see “Comparing and Synchronizing Objects Between Repositories and Archives” on page 153.
	Synchronizes objects from the repository that the Second Selection section represents with the repository that the First Selection section represents. For more information, see “Comparing and Synchronizing Objects Between Repositories and Archives” on page 153.
	Expands the entire tree in the First Selection section and the Second Selection section.
	Collapses the entire tree in the First Selection section and the Second Selection section.
Delete Button	Deletes objects after a comparison.

Dialog Boxes That Set Development Options

This topic describes the dialog boxes that you use to set development options. It includes the following information:

- [Development Options for Visualization Views on page 270](#)
- [Development Options for Scripting on page 270](#)
- [Development Options for Debugging on page 272](#)

Development Options for Visualization Views

Table 44 describes the development options that you can set for visualization views. For information about using this dialog box, see [“Viewing Object Relationships” on page 152](#).

Table 44. Development Options for Visualization Views

Option	Description
Use System Font	Uses a system font for the visualization views.
Use a Custom Font	You can use the Font, Size, and Zoom drop-down lists to choose your preferred font.
Boxes with 3D borders	Displays boxes with a three dimensional border.
Icon and name only	Displays the object name and the same object icon that the Object Explorer shows.
Simple outline boxes	Displays each object name in a simple box.
Always print outline style	Displays visualization details in outline style.

Development Options for Scripting

Table 45 describes the development options that you can set for scripting. For more information, see [“Setting Options for the Siebel Script Editor” on page 121](#).

Table 45. Development Options for Scripting

Section	Option	Description
Font	Name	Sets the font name that appears for a script.
	Size	Sets the font size that appears for a script.

Table 45. Development Options for Scripting

Section	Option	Description
Script Assist		Allows you to set Script Assist options. You must enable the ST eScript Engine so that you can use Script Assist. For more information, see “Overview of Using the ST eScript Engine” on page 123.
	Enable Method Listing	Allows Script Assist to display a drop-down list that includes the methods and properties that are available for a declared object.
	Tab Width	Specifies the number of spaces that Siebel Tools uses for a tab character in the script. The default value is four spaces.
	Enable Auto Complete	If this check box contains a check mark, then Siebel Tools auto completes a method name or property name. It does this if you enter the minimal number of unique characters that are required to complete the name. If it locates strings that are not unique, then it displays a drop-down list.
	Auto Indent	If this check box contains a check mark, then Siebel Tools indents each succeeding line of script to the position that the current line sets.
	Enable Favorites	If this check box contains a check mark, then Siebel Tools displays the object, method, or property name that you use most frequently in italics at the start of the Script Assist window.
	Engine Settings	For more information, see “Setting Options for the ST eScript Engine” on page 127.
Language	Default Language for New Scripts	You can choose eScript or Visual Basic.
	Browser Script Compilation Folder	You can specify the folder where Siebel Tools stores the Browser script that it compiles. For example: C:\Program Files\Siebel\8.0\web client\PUBLIC\enu In this example, browser script files reside in the following folder: C:\Program Files\Siebel\8.0\web client\PUBLIC\enu\genbscript_time stamped_folder\bscripts\all.

Table 45. Development Options for Scripting

Section	Option	Description
Debugging		Allows you to set options for the Siebel Debugger. For more information, see “Setting Debug Options to Open the Siebel Client” on page 219 .
	Adjust Breakpoint to Next Valid Line	If this check box contains a check mark, and if you delete a breakpoint on an invalid code line, then Siebel Tools creates a breakpoint at the next valid line.
	Make Debugger Window Active When Debugging	If this check box contains a check mark, then Siebel Tools displays the Siebel Debugger window when it is in debug mode.
	Always Enter the Debugger When an Error Occurs	If this check box contains a check mark, and if a script error occurs, then Siebel Tools displays the Siebel Debugger window.

Development Options for Debugging

Table 46 describes the development options that you can set for debugging. To access this dialog box in Siebel Tools, you click the View menu, click Options, and then click the Debug tab. Siebel Tools stores the settings you make on the Debug tab in the following user preference file:

`log\prod\Siebel Tools.spf`

It stores this file in the `SIEBEL_TOOLS_ROOT\BIN` folder.

Table 46. Development Options for Debugging

Option	Description
Executable	Enter the name of the Siebel Web Client executable. For example: <code>si ebel . exe</code> The default value is <code>siebel.exe</code> . Siebel Tools runs this executable in debug mode or automatically after compile finishes.
CFG File	Enter the name of the configuration file that the Siebel client uses. For example: <code>C:\Program Files\Siebel\8.0\web client\BIN\ENU\uagent.cfg</code>
Browser	Enter the path to the browser executable. For example: <code>C:\Program Files\Internet Explorer\iexpl ore. exe</code>
Working Directory	Enter the Siebel root folder. This folder includes the Siebel executable and the DLLs (dynamic link libraries). For example: <code>C:\Program Files\Siebel\8.0\web client\BIN</code>

Table 46. Development Options for Debugging

Option	Description
Arguments	Enter the options that Siebel Tools uses when it opens the watch window: <ul style="list-style-type: none"> ■ /h. Enables local debugging of server scripts. ■ /s file name. Enable SQL spooling.
Prompt for This Information Each Time	If this check box contains a check mark, then Siebel Tools displays information each time it runs a debug operation. For example, it can display the name of the executable, the name of the CFG file, the browser configuration, and so on.
Show Workflow Primary Business Component Data	If this check box contains a check mark, then the Watch window in the Workflow Simulator displays information about the workflow process. It displays the name of each business component field and the value for each of these fields. These fields include fields from the primary business component of the business object that the workflow process references.
User Name	Enter the user name that Siebel CRM requires to log in to the Siebel application you are debugging.
Password	Enter the password that Siebel CRM requires to log in to the Siebel application you are debugging.
Data Source	Choose a default data source. The values you can choose depend on Oracle's Siebel Tools configuration file that you specify in the CFG File option. The Siebel Web Client connects to this local database.
Enable Profiler	For more information, see "Using the Script Profiler" on page 138 .

Parameters That Convert Symbolic Strings

This topic describes the parameters that you can use to convert a symbolic string. You use these parameters with a conversion utility. For more information, see ["Converting Symbolic Strings" on page 197](#).

This topic includes the following information:

- [Parameters You Use with the Conversion Export Utility on page 274](#)
- [Parameters That You Can Use with the Conversion Import Utility on page 275](#)

Parameters You Use with the Conversion Export Utility

Table 47 describes the parameters that you can use with the conversion export utility.

Table 47. Parameters You Use with the Conversion Export Utility

Parameter	Description
Filename	Required. Specifies the name of the export file.
Repository	Required. Specifies the name of the repository. The repository name is case sensitive.
Object	Required. Specifies the object type that contains the strings that this utility exports. For example: Control The object name is case sensitive.
LogFile	Specifies the name of the log file.
Language	Specifies the language that this utility uses as the primary language to match when it searches for duplicate symbolic strings. For example, each symbolic string includes the following child records: <ul style="list-style-type: none"> ■ English (ENU) ■ French (FRA) ■ German (DEU) <p>If you set the Language parameter to ENU, then the conversion export searches for matches between the ENU records. If it finds matches, then it examines the other child records of the other languages. If all child records match, or if one language includes a superset of one of the other languages, then this utility considers them as matching symbolic strings.</p>
MatchMin	Specifies the minimum number of matches in a set of matching symbolic strings before this utility writes them to the file. The default value is 2.
SQLLog	Specifies the SQL log file name. If you set this parameter, then this utility logs all SQL that it runs to this file.
ExcludeNull	Specifies a TRUE or FALSE value. If TRUE, then this utility excludes null values for conversion consideration. The default value is TRUE.
UseFullMatch	Specifies a TRUE or FALSE value. If TRUE, then this utility matches records against all the other possible match candidates before it discards them. The default value is TRUE.

Table 47. Parameters You Use with the Conversion Export Utility

Parameter	Description
UseExactMatch	Specifies a TRUE or FALSE value. If TRUE, then this utility considers records as a match only if all of the records include the same number of language records, and if the same values exist for each language. It does not consider partial matches. The default value is FALSE.
SkipInactive	Specifies a TRUE or FALSE value. If TRUE, and if the Inactive property of the record is set to Y, then this utility skips this record. The default value is TRUE.

Parameters That You Can Use with the Conversion Import Utility

Table 48 describes the parameters that you use with the conversion import utility.

Table 48. Parameters You Use with the Conversion Import Utility

Parameter	Description
Filename	Required. Specifies the name of the import file. You must use the same name that you use when you export symbolic strings.
Repository	Required. Specifies the name of the repository.
LogFile	Specifies the log file.
UnlockProjects	Specifies a TRUE or FALSE value. If TRUE, then this utility unlocks all projects when the conversion finishes. This configuration is useful if multiple instances of the conversion service run against the same database. The default value is TRUE.
SkipParentUpdates	Specifies a TRUE or FALSE value. If TRUE, then this utility does not update parent objects to use the symbolic string. The default value is FALSE. Set SkipParentUpdates to TRUE only if you do not simultaneously run multiple instances of the import. If you set SkipParentUpdates to TRUE, and if you run multiple instances, then errors might occur. The utility might abort an update or delete records because another instance updates the project at the same time.
SQLLog	Specifies the log file name. If you use this parameter, then this utility logs all SQL that it runs to the file you specify.
Project	Required. Specifies the name of the project in the repository that includes the new strings. Predefined symbolic strings reside in the Symbolic Strings project. You can configure this utility to import custom strings. For more information, see “About Predefined Objects” on page 23 .

Table 48. Parameters You Use with the Conversion Import Utility

Parameter	Description
DeleteLocales	Specifies a TRUE or FALSE value. If TRUE, and if all translatable strings are NULL, and if language override is not enabled, then this utility deletes locale records. If FALSE, then this utility sets the locale record to Inactive. The default value is TRUE. For more information, see “Enabling Language Override” on page 19 .
CheckTranslateFlag	Specifies a TRUE or FALSE value. If TRUE, and if the Translate property for the object is N, then this utility does not convert this object. The default value is TRUE.
LogErrorRecords	Specifies a TRUE or FALSE value. If TRUE, then this utility exports all error records to a separate log file. The default value is FALSE.