Siebel Object Interfaces Reference

Siebel Innovation Pack 2015,
Rev. A

May 2015

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 4: Using Siebel Visual Basic and Siebel eScript

Chapter 5: Siebel Object Interfaces Reference

Chapter 6: Browser Script Quick Reference

Chapter 7: Siebel VB Quick Reference

Chapter 8: Siebel eScript Quick Reference

Chapter 9: COM Data Server Quick Reference

Chapter 10: COM Data Control Quick Reference

Chapter 11: Web Client Automation Server Quick Reference

Chapter 12: Mobile Web Client Automation Server Quick Reference

Chapter 13: Siebel Java Data Bean Quick Reference

Index

1 What's New in This Release

What's New in Siebel Object Interfaces Reference, Siebel Innovation Pack 2015, Rev. A

Table 1 lists the changes in this revision of the documentation to support this release of the software.

NOTE: Siebel Innovation Pack 2015 is a continuation of the Siebel 8.1/8.2 release.

Table 1.   New Product Features in Siebel Object Interfaces Reference, Siebel Innovation Pack 2015, Rev. A

Topic	Description
"Allowing Development Environments Access to the Siebel Object Model" on page 35	Modified topic. It describes updated information about allowing development environments access to the Siebel Object Model in Siebel Innovation Pack 2015.
"Accessing the Web Client Automation Server" on page 35	Modified topic. It describes updated information about accessing the Web Client Automation Server in Siebel Innovation Pack 2015.

2 About Object Interfaces and the Programming Environment

This chapter describes Oracle's Siebel Object Interfaces and the programming environment you use to customize them. It includes the following topics:

- Object Interfaces You Can Use to Access Siebel Objects on page 9
- About the Siebel Programming Environment on page 17
- Siebel Object Interface Methods That You Can Use to Control Data and Objects on page 20

Object Interfaces You Can Use to Access Siebel Objects

This topic describes object interfaces you can use to access Siebel objects. It includes the following topics:

- "Overview of Interfaces You Use to Access Siebel Objects" on page 9
- "Objects You Can Access Through a Siebel Object Interface" on page 10
- "About the Siebel Java Data Bean Object Interface" on page 12
- "About the Siebel COM Object Interface" on page 12

Overview of Interfaces You Use to Access Siebel Objects

A Siebel *object interface* is a collection of object interface methods that reside on Siebel objects that make their data and functions available to custom code that you write in Server Script, and also to other languages that are external to Siebel CRM. These interfaces provide access to Siebel business objects that contain object interface methods, object interface events, and data.

A Siebel object interface can provide an interface between Siebel CRM and an external application. Siebel object interface definitions reference Siebel business objects and object definitions that you can configure so that Siebel CRM automatically upgrades them during a release update.

You can integrate client and server applications from different third-party vendors. Application integration typically requires that software programs interactively pass data back and forth. Application integration sometimes requires that one application controls another application.

An *object interface method* is a function that allows you to control data and objects. Siebel CRM provides object interface methods to perform operations, such as manipulating files that Siebel CRM stores in the Siebel File System, or updating records through a Siebel object, such as a business component.

Objects You Can Access Through a Siebel Object Interface

You can use the following Siebel object interfaces to create or modify a Siebel object:

■ Scripting using Server Script or Browser Script

■ Component Object Model (COM) using the Web Client Automation Server, COM Data Control, COM Data Server, or Mobile Web Client Automation Server

■ Java using Siebel Java Data Bean

Table 2 lists the types of objects you can access. If a table cell includes Yes, then you can use the object type listed in the Object Type column with the Siebel object interface listed in the column header.

Table 2.   Types of Objects You Can Access Through a Siebel Object Interface

Object Type	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Siebel Java Data Bean
Applet	Yes	Yes	No	No	No	No	No
Application	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Business Component	Yes	Yes	No	Yes	Yes	Yes	Yes
Business Object	Yes	Yes	No	Yes	Yes	Yes	Yes
Business Service	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Control	No	Yes	No	No	No	No	No

Siebel CRM uses other object types that this topic does not describe, including some specialized types. If this topic does not describe an object type, then it is not available through a Siebel object interface. If you reference it, then Siebel CRM might not pass it to an external DLL, such as a Microsoft Visual Basic COM DLL.

For more information about the objects that Table 2 describes, see *Configuring Siebel Business Applications*.

Applets

You can add a script to an applet to access this applet through an object interface. In Siebel Tools, you right-click the applet, and then choose the Edit Server Scripts or Edit Browser Scripts menu item. This work is similar to adding a script to a business component. For more information, see "Using Script to Add Business Logic to a Business Component" on page 78.

You can use the following scripting languages with an applet:

■ Siebel VB and Siebel eScript in a Server Script

■ Browser JavaScript in Browser Script

If the Siebel client runs in high interactivity mode, then Siebel CRM does not run some applet events, such as WebApplet_ShowControl and WebApplet_ShowListColumn.

Business Services

A *business service* is an object type that contains a set of predefined methods. Siebel CRM uses C++ code to implement them. It can also contain custom methods that reside in Siebel script. It allows you to configure Siebel CRM to call C++ code or to call a scripted business service method from a script that you create. You can use a business service in the following ways:

■ Called from a script or from an object interface.

■ Reusable and can persist through a session.

■ Simulate a global procedure.

■ Provide a generic code library that Siebel CRM calls from multiple scripts.

■ Modify object properties. You can write a script in Siebel VB or Siebel eScript that configures a business service that modifies object properties.

You can do one of the following to create a custom business service:

■ Add a record in the Business Services list in Siebel Tools.

■ Use administrative views in the Siebel client.

To use the Web Client Automation Server or Browser Script to call a business service, you must register the business service in Siebel Tools as an application user property. This configuration prevents Service Not Found errors. For more information, see "GetService Method for an Application" on page 141.

You can use the following types of business services:

■ **Repository.** Defined in Siebel Tools and stored in the Siebel repository file (SRF).

■ **Run-time.** Defined in the Siebel client and stored in the Siebel database.

For more information, see *Integration Platform Technologies: Siebel Enterprise Application Integration*.

Repository Business Services

You can use the following types of repository business services:

■ **Standard.** References the CSSService class. You can script or modify a standard business service.

■ **Specialized.** References a specialized C++ class. If Siebel Bookshelf documents a specialized business service, then you can script or modify it.

You cannot configure Siebel CRM to modify a repository business service at run time, or to use a run-time script to override a predefined business service.

Property Sets

A *property set* is a collection of properties that you can use to store data. It can include a child property set that forms a tree data structure. You use a property set to handle inputs to and outputs from a business service. For more information, see *Integration Platform Technologies: Siebel Enterprise Application Integration*.

User Interface Controls

A *user interface control* is an object type that defines a user interface element, such as a text box, check box, or a button. Browser Script can access the properties of a control. The controls on the applet that are currently visible are the only controls that are available to Browser Script.

About the Siebel Java Data Bean Object Interface

The *Siebel Java Data Bean* is a set of Java libraries that use the J2SE Development Kit (JDK). It is similar to the interfaces that are available through COM Data Control. It allows you to do the following work:

■ Use an external application, external component, or Java applet to access Siebel objects without displaying the Siebel client.

■ Access a Siebel application to read and write data.

■ Incorporate the Java libraries in Java applications, applets, servlets, JSPs, or Enterprise Java Beans. You can add these items to your Java application.

For more information about:

■ Developer resources for Java technology, see the following:

> http://www.oracle.com/technetwork/java/index.html

■ Communication with an external application, see "How an External Application Communicates with a Siebel Application" on page 13.

■ Operating systems and JDKs that you can use, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

About the Siebel COM Object Interface

You can access a Siebel COM object interface in any of the following ways:

■   COM Data Control

■   COM Data Server

■   Web Client Automation Server

■   Mobile Web Client Automation Server

You can use any of the following languages to access a Siebel COM interface:

■   JavaScript

■   Visual Basic

■   C++

You cannot use the Perl programming language to access a Siebel COM interface.

The programming environment you use might limit the features that Siebel CRM can use the Siebel
COM servers. For example, do not use Siebel VB code for the Data Server as a Windows NT service.

How an External Application Communicates with a Siebel Application

COM Data Control is a type of Siebel Object Interface that allows an external application to connect
and communicate with the *Siebel Application Object Manager*, which is a multithreaded, multiprocess
application server that hosts Siebel business objects and allows session connections with Siebel
clients. This connection allows the external application to access Siebel business objects. The Siebel
Internet Session Network API (SISNAPI) protocol allows this communication.

Figure 1 illustrates how an external application uses COM Data Control to communicate with the
Siebel application.
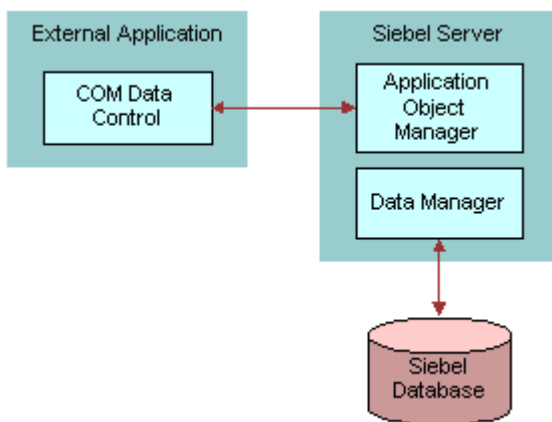
Figure 1.   How an External Application Uses COM Data Control

To use COM Data Control to develop a Siebel application, you must install, configure, and make sure
Siebel CRM is running a Siebel Application Object Manager on a Siebel Server. For more information,
see *Siebel System Administration Guide*.

For information about the SISNAPI protocol, see *Siebel Deployment Planning Guide*.

Servers That the Siebel COM Interface Uses

This topic describes the servers that the Siebel COM Interface uses.

Web Client Automation Server

The Web Client Automation Server does the following:

■ Allows an external application to call a business service and manipulate property sets.

■ Runs as a small Siebel COM object in the Web browser in Internet Explorer version 5.0 or later.

■ Can be used with the Siebel Web Client and the Siebel Mobile Web Client.

Figure 2 illustrates how an external application can call a business service and manipulate a property
set that resides on the Web Client Automation Server.
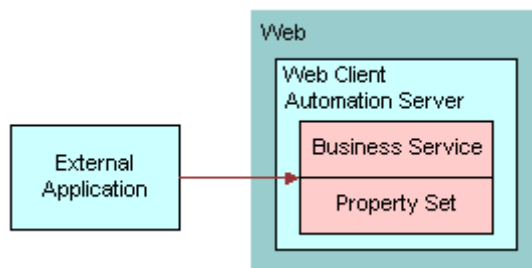
Figure 2.   How an External Application Interacts with the Web Client Automation Server

The Web Client Automation Server includes the following requirements:

■ Runs only with a high interactivity client.

■ The Siebel Web Client must be running.

■ You must set the EnableWebClientAutomation parameter for the Application Object Manager to
TRUE. This setting configures Siebel CRM to download a small ActiveX control to the desktop and
start the SiebelHTMLApplication process.

■ It might be necessary for you to adjust ActiveX controls, plug-ins, and security settings in the
browser.

You cannot configure Siebel CRM to call the Web Client Automation Server directly from an active
instance of a Siebel application.

Siebel CRM uses one of the following names for the process that represents the Web Client
Automation Server. The Windows Task Manager displays this name:

■ siebelhtml.exe

■ siebelhtmlapplication.exe

■ SIEBEL~1.EXE

If the user ends the Siebel Web Client session, then Siebel CRM stops this process.

For more information, see "Allowing Development Environments Access to the Siebel Object Model" on
page 35.

Mobile Web Client Automation Server

The Mobile Web Client Automation Server accesses the server object that the Siebel application
starts. If your configuration can access this object, then it can get other Siebel objects and run Siebel
object interface methods through these other objects.

Figure 3 illustrates how an external application can control a Siebel application that uses the Web
Client Automation Server.
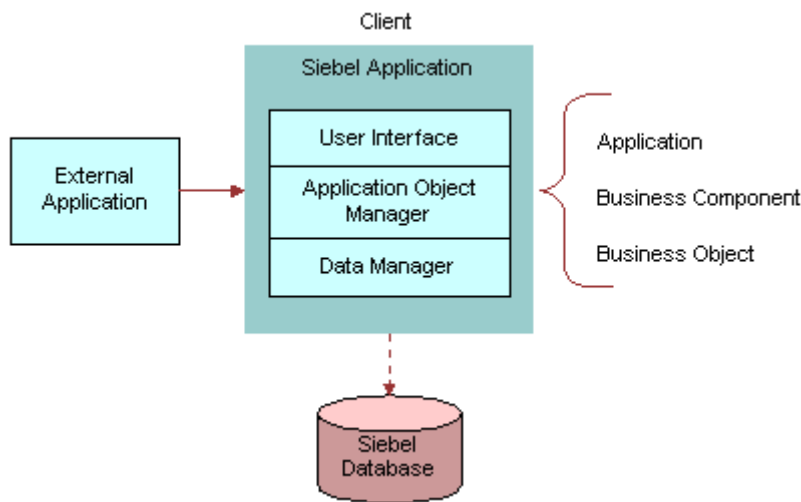
Figure 3.   How External Applications Can Control a Siebel Application That Uses the Web Client
Automation Server

The Mobile Web Client Automation Server includes the following requirements:

■   The Siebel Mobile Web Client must be running.

■   The EnableWebClientAutomation parameter that resides in the InfraUIFramework section of the
    Siebel application configuration (CFG) file must be set to TRUE.

■   If you use Microsoft Visual Basic version 5.0 or later, then the sobjsrv.tlb file must reside in the
    same folder where the Siebel application configuration (CFG) file resides. If this file does not
    reside in the correct folder, then the COM Data Server does not work.

■   A call that you configure Siebel CRM to make to the Mobile Web Client Automation Server is *out
    of process*. If your customization creates a DLL that runs *in process* with the Siebel application,
    then the calls that Siebel CRM makes from the DLL to the Mobile Web Client Automation Server
    are *out of process*. For more information, see "How Siebel CRM Uses Memory and Resources with
    the Mobile Web Client Automation Server" on page 16.

For more information, see "Accessing the Mobile Web Client Automation Server" on page 37.

How Siebel CRM Uses Memory and Resources with the Mobile Web Client Automation Server
Siebel CRM starts a process to run the Siebel Mobile Web Client. This process uses memory and
resources that are specific to this process, which are *in process*. If your configuration communicates
with the Siebel Mobile Web Client while it is running, then the resources that Siebel CRM uses in this
communication are separate form the memory and resources that it uses in the process that it
started to run the Siebel Mobile Web Client. These separate resources are *out of process*.

COM Data Server
Figure 4 illustrates how an external application uses the COM Data Server that does not include user
interface objects. The COM Data Server uses the same technology that the Siebel Mobile Web Client
uses to connect to the Siebel database.
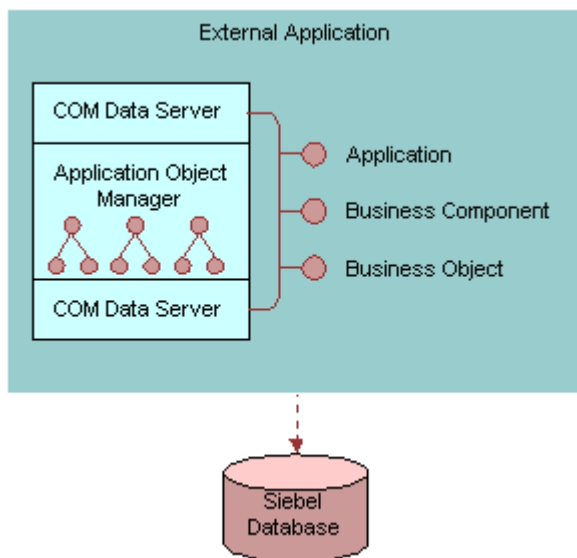
Figure 4.   Siebel COM Data Server

The Mobile Web Client Automation Server includes the following requirements:

- The way your configuration starts a Siebel COM server depends on the programming tool or
  language you use.

- The COM Data Server runs without the Siebel client, so you must use the Login method to set
  up your Data Server object.

- No current active Siebel objects exist, so you cannot use an object interface method that returns
  active Siebel objects. You must use your own Siebel objects.

- If you use Microsoft Visual Basic version 5.0 or later, then the sobjsrv.tlb file must reside in the
  same folder as the Siebel application configuration (CFG) file. If this file does not reside in the
  correct folder, then the COM Data Server does not work.

- Do not run the Microsoft VB Debug environment while your configuration communicates with the
  COM Data Server.

■ If your configuration uses the COM Data Server, then the COM client cannot create multiple
connections to the Siebel COM Server. You must restart the COM client before you can attempt
another connection. Use COM Data Control instead.

■ Calls made to the COM Data Server are *in process*. For more information, see "How Siebel CRM
Uses Memory and Resources with the Mobile Web Client Automation Server" on page 16.

Note the different ways that the following servers handle DLLs:

■ **COM Data Server.** A DLL runs in the same address space where the calling program runs.

■ **Mobile Web Client Automation Server.** An executable runs in a dedicated address space. A
DLL that a server task accesses must be capable of running in a multithread environment.

For more information, see "Accessing the COM Data Server" on page 41.

About the Siebel Programming Environment

This topic describes the Siebel programming environment.

Programming Languages

You can use the following programming languages to access object interface methods and object
interface events:

■ **Siebel VB (Siebel Visual Basic).** A programming language that is syntactically and
semantically compatible with Microsoft Visual Basic. It includes an editor, debugger, interpreter,
and compiler. It runs only on the Windows operating system.

■ **Siebel eScript.** A programming language that is syntactically and semantically compatible with
JavaScript. It uses the same tools that Siebel VB uses. Siebel eScript runs on the Windows and
UNIX operating systems.

For more information, see Chapter 4, "Using Siebel Visual Basic and Siebel eScript."

Server Script

A *Server Script* is a type of script that the Siebel Server interprets and runs. You can use the following
scripting languages in a Server Script:

■ **Siebel VB.** Siebel VB uses most of the same commands and standards as Microsoft Visual Basic,
so you can customize your Siebel application and reduce training costs. Siebel CRM supports
Siebel VB only on the Microsoft Windows operating system.

■ **Siebel eScript.** Siebel eScript uses most of the same commands and standards as JavaScript,
so it provides you the same advantages in an alternative language. You can use Siebel eScript
on all operating systems that Siebel CRM supports.

For more information, see Chapter 4, "Using Siebel Visual Basic and Siebel eScript."

Browser Script

A *Browser Script* is a type of script that the browser interprets and runs. It interacts with the
Document Object Model and with the Siebel Object Model in the browser through the Browser
Interaction Manager. You write Browser Script in JavaScript. You can script the behavior of Siebel
events and the browser events that the Document Object Model makes available. The Document
Object Models for Internet Explorer and Netscape Navigator are different.

You can use Browser Script only with a Siebel application that runs in high interactivity mode. The
only exception is if you script a control event that you use with the Browser Document Object Siebel
Model. Siebel CRM version 7 introduced Browser Script. For more information, see "Document Object
Model Events You Can Use" on page 334.

Do not use Browser Script to manipulate the location of a frame or a form in Siebel CRM because this
configuration causes Siebel CRM to load a new page. This configuration is a violation of preferred
security practices, so the result is a permission denied error.

A high interactivity Siebel application can contain a standard interactivity view. For example, the
Home Page view or the Dashboard view. If the Siebel application uses high interactivity or standard
interactivity, then you cannot use Browser Script on an applet in these views. Instead, the Siebel
application calls the WebApplet_ShowControl Server Script that resides on the applet.

For more information, see Chapter 6, "Browser Script Quick Reference." For information about creating
Browser Script, see *Configuring Siebel Business Applications*.

Siebel Script Editor

The *Siebel Script Editor* is an integrated editor that you can use to create, view, edit, and save
custom code. It includes the Script Assist code editor. This editor includes the following features to
help reduce errors when you develop a script:

- Autocomplete.

- Autoindentation.

- A list of object interface methods.

- Method signature capabilities. Some methods that the Script Assist editor lists include the input
  parameter names and types, outputs from the method, and the method type. For example, if a
  method returns chars, then it lists the following term: chars.

Siebel CRM version 7.8 introduced Script Assist. For more information, see *Using Siebel Tools*.

Siebel Debugger

The *Siebel Debugger* is a tool that helps you detect errors that occur in the code of a Siebel
programming language. It does not help you detect errors that occur outside of the context of the
code. You can configure Siebel CRM to start the Siebel Debugger automatically from a Siebel
application if a run-time error occurs. You can also start the debugger from the Debug toolbar or the
Debug menu in Siebel Tools. For more information, see *Using Siebel Tools*.

Siebel Compiler and Run-Time Engine

The *Siebel Compiler* and *Run-Time Engine* is a nonvisual component of a Siebel programming language that compiles and runs custom code. It is similar to Microsoft's Visual Basic Language Interpreter. Siebel CRM compiles Siebel code and stores it in the SRF file.

You can click the Compile icon on the Debugger toolbar in Siebel Tools to start the Siebel Compiler and Run-Time Engine. You can also start it if you compile a project that contains an object definition that is associated with a Siebel script. The Siebel Compiler and Run-Time Engine do not include a user interface. The compiler compiles the custom code, and then returns a message that indicates success or failure.

Compilation Order

The Siebel Compiler compiles Siebel VB functions and procedures in alphabetical order as they occur in the 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 can use the Declare statement to declare the function or procedure in the declarations section of the general section.

Siebel eScript does not require you to declare a function before you use it.

For more information, see *Siebel VB Language Reference*.

ST eScript Engine

The ST eScript engine is available in Siebel CRM version 7.8 and later. It is the default Siebel eScript scripting engine in Siebel CRM version 8.0 and later. It allows you to use strongly typed objects that are compliant with the ECMAScript edition 4 specification. It also provides early and late binding. For information about:

■ Binding, see "About Early and Late Binding" on page 19.

■ The differences that exist between the ST eScript engine and the T engine, see *Siebel eScript Language Reference*

■ Using the ST engine, see *Using Siebel Tools*.

About Early and Late Binding

Early binding occurs if you bind a specific object instance to a variable. The following code binds an object to a variable at design time. It is an example of early binding:

```
var lo_bo = TheApplication().GetBusObject("Account");
```

Late binding occurs if you bind an object to a variable only at run time. The following code is an example of late binding:

```
if (TheApplication().ActiveBusObject().Name() == "Account")
var ls_bo_name = TheApplication().ActiveBusObject().Name();
```

This late binding code does not specify a specific object. The compiler cannot identify this object and
it cannot identify that the Name method is part of the object. Siebel CRM can only bind this object
to a variable at run time.

Siebel Script Profiler

The *Siebel Script Profiler* is a tool that gathers and displays data for the scripts that Siebel CRM runs
when you start a Siebel application in Debug mode from Siebel Tools. Siebel Tools displays the
profiler data in a window that is similar to the Watch window. It automatically updates information
in this window while a script runs in the Siebel application.

The Script Profiler includes the following features:

■ Tree view that displays how the script runs

■ Allows you to profile functions and profile lines of chosen functions

■ Allows you to use the Siebel Debugger and Script Profiler at the same time

■ Allows you to view the compilation time that the script requires to run

You can use this data to do the following work:

■ Monitor the performance of a script.

■ Identify performance bottlenecks.

■ Compare profile data with previous script runs.

You can use the Script Profiler only with the ST eScript Engine.

For more information, see *Using Siebel Tools*.

Siebel Object Interface Methods That You Can Use to Control Data and Objects

This topic describes object interface methods that you can use to control data and objects. It includes
the following topics:

Methods That Locate Objects

This topic describes object interface methods that allow your configuration to locate an active
instance of an object that resides in a Siebel application so that another method can use this object:

■   The active object is an instance of an object that Siebel CRM currently displays as active.

■   The active control is the control that Siebel CRM currently displays as active.

■   The active applet is the applet that contains the active control.

■   The active business component is the business component that the active applet references.

If a Siebel object interface can locate an object, then it can use or manipulate this object.

You can use any of the following object interface methods in your configuration to locate an object:

■   "ActiveMode Method for an Applet" on page 103

■   "BusObject Method for an Applet" on page 104

■   "ActiveBusObject Method for an Application" on page 127

■   "ActiveViewName Method for an Application" on page 129

■   "GetBusObject Method for an Application" on page 136

■   "BusComp Method for a Control" on page 297

■   "Name Method for a Control" on page 299

■   "GetValue Method for a Property Set" on page 316

■   "TheApplication Method" on page 324

Methods That Access Data from Business Components

This topic describes the object interface methods that allow your configuration to access and modify data that resides in a Siebel application. A business component can provide data for each field of each business component record, such as the fields of an opportunity. You can use a business component to read data, manipulate data, and then write this data to the Siebel database.

You can use a custom script that you write in Siebel VB or Siebel eScript. For example, if you create a script in Siebel VB or Siebel eScript that references the NewRecord event in a business component, then Siebel CRM calls this script. This situation is true if any of the following items calls the event:

■   The NewRecord method

■   Another Siebel VB or Siebel eScript script

■   A Siebel object interface

An event is available only with Siebel VB or Siebel eScript.

Adding and Inserting Records

You can use Siebel VB or Siebel eScript to mimic one of the following commands in the context of a many-to-many relationship:

■   **Add New Record.** Associates a new child record.

■   **Insert Record.** Creates a new record in the child business component.

You can use one of the following methods to associate a new child record:

■ GetAssocBusComp

■ Associate

You can use one of the following methods to create a new record in the child record:

■ The NewRecord method in a child business component

■ The GetMVGBusComp method and the NewRecord method

How Siebel CRM Saves a Record to the Siebel Database

Siebel CRM saves a record to the Siebel database in the following situations:

■ Explicitly by using the BusComp.WriteRecord method.

■ Navigating away from the current record by any of the following object interface methods:

 ■ BusComp.Associate.

 ■ BusComp.DeleteRecord. It moves the cursor to another record, so this method automatically saves the record.

 ■ BusComp.FirstRecord.

 ■ BusComp.LastRecord.

 ■ BusComp.NextRecord.

 ■ BusComp.PreviousRecord.

■ Closing a business component by setting the BusComp method to Nothing.

Example of Accessing Data from an Existing Business Component Instance

If Siebel CRM starts an event, then the code in this example calls an object interface method that resides on an existing business component instance. The term *instance* describes the current, run-time state of an object. For example, a *business component instance* is a run-time occurrence of a business component. It includes all of the run-time data that the business component currently contains, such as the values for all business component fields and the values for all properties of this business component. For example, an instance of the Contact business component includes the current, run-time value of the City field that resides in this business component, such as San Francisco. You can configure Siebel CRM to get a business component instance, and then modify this data or call the methods that this business component references.

In the following example, the VB script resides in the SetFieldValue event of the business component:

```
Sub BusComp_SetFieldValue (FieldName As String)
Dim desc As String
Dim newDesc As String

TheApplication.TraceOn "c:\temp\trace.txt", "Allocation", "All"
If FieldName = "Type" Then
```

```
        newDesc = "Any valid string that contains the new description."
        desc = Me.GetFieldValue("Description")
        TheApplication.Trace "The previous description is " & desc
        Me.SetFieldValue "Description", newDesc
        TheApplication.Trace "The new description is " & newDesc

    End If
    TheApplication.TraceOff

    End Sub
```

Example of Accessing Data from a New Business Component Instance

The example in this topic describes how to create a new business object instance and a business
component instance. It uses the PreSetFieldValue event of the Opportunity business component. If
the user updates the Sales Stage to 07 - Verbal Agreement, then Siebel CRM requires the user to
associate a decision maker with the opportunity. Otherwise, Siebel CRM resets it to the previous
value. To determine if a vice president or president is associated with the opportunity, Siebel CRM
searches the contacts that it associates with the opportunity.

The following steps describe the logical flow of object interface methods that Siebel CRM uses to
create a new business component instance:

1 GetBusComp.

2 SetViewMode. This method is optional. You can use it to modify the default value of the view
mode.

3 ActivateField.

4 ClearToQuery.

5 SetSearchSpec or SetSearchExpr.

It is not necessary to activate a field that includes a search specification and a search expression,
unless the GetFieldValue method or the SetFieldValue method also references this field.

6 ExecuteQuery.

Example of Using Siebel VB to Access Data from a New Business Component Instance

The following example uses Siebel VB to access data from a new business component instance:

```
    Function BusComp_PreSetFieldValue (FieldName As String, FieldValue As String) As
    Integer

    Dim RetValue As Integer
    RetValue = ContinueOperation
    Select Case FieldName
        Case "Sales Stage"
            If FieldValue = "08 - Negotiation" Then
                ' Do not allow the sales cycle to be changed to this value
                ' if the decision-maker is not a contact for the Oppty.
                ' Decision-maker defined as anyone with rank VP and above
                Dim oBusObj As BusObject
                Dim sRowId As String
```

```
            Dim iViewMode As Integer
            sRowId = GetFieldValue("Id")
            iViewMode = GetViewMode
            Set oBusObj = TheApplication.ActiveBusObject

            ' Parent-child relationship is established if
            ' BusComps are instantiated from the same BusObject.
            ' The ContactBC has all contact records for the
            ' current Oppty record.
            Set ContactBC = oBusObj.GetBusComp("Contact")
            With ContactBC
                .ClearToQuery
                .SetSearchSpec "Job Title", "*VP*"
                .ExecuteQuery ForwardBackward
                If (.FirstRecord = 1) Then
                    TheApplication.RaiseErrorText "Found a decision maker"
                Else
                    RetVal = ContinueOperation
                End If
            End With
            Set ContactBC = Nothing
            Set oBusObj = Nothing
        End If
    End Select
    BusComp_PreSetFieldValue = RetValue
End Function
```

Example of Using Siebel eScript to Access Data from a New Business Component Instance
The following example uses Siebel eScript to access data from a new business component instance:

```
function BusComp_PreSetFieldValue (FieldName, FieldValue)
{
    var RetValue = ContinueOperation;
    switch (FieldName)
    {
        case "Sales Stage":
        if (FieldValue == "08 - Negotiation")
        {
        //Do not allow the sales cycle to be changed to this value
        //if the decision-maker is not a contact for the Oppty.
        //Decision-maker defined as anyone with rank VP and above
            var oBusObj;
            var sRowId;
            var iViewMode;
            sRowId = this.GetFieldValue("Id");
            iViewMode = this.GetViewMode();
            oBusObj = TheApplication().ActiveBusObject();
            //Parent-child relationship is established if
            //BusComps are instantiated from the same BusObject.
            //The ContactBC has all contact records for the
            //current Oppty record.
            ContactBC = oBusObj.GetBusComp("Contact");
            with (ContactBC)
```

```
                    {
                        ClearToQuery();
                        SetSearchSpec("Job Title", "*VP*");
                        ExecuteQuery(ForwardBackward);
                        if (FirstRecord())
                        {
                            TheApplication().RaiseErrorText("Found a decision maker");
                        }
                        else
                        {
                            RetVal = ContinueOperation;
                        }
                    }
                    ContactBC = null;
                    oBusObj = null;
                }
                break;
            }
    return(RetVal);
    }
```

Methods That Get Data From Business Components

The following object interface methods get data from a business component:

- "ActivateField Method for a Business Component" on page 186

- "ActivateMultipleFields Method for a Business Component" on page 188

- "Associate Method for a Business Component" on page 190

- "ClearToQuery Method for a Business Component" on page 193

- "CountRecords Method for a Business Component" on page 194

- "DeactivateFields Method for a Business Component" on page 194

- "DeleteRecord Method for a Business Component" on page 196

- "ExecuteQuery Method for a Business Component" on page 197

- "ExecuteQuery2 Method for a Business Component" on page 200

- "FirstRecord Method for a Business Component" on page 200

- "FirstSelected Method for a Business Component" on page 202

- "GetAssocBusComp Method for a Business Component" on page 204

- "GetFieldValue Method for a Business Component" on page 205

- "GetFormattedFieldValue Method for a Business Component" on page 206

- "GetMultipleFieldValues Method for a Business Component" on page 209

- "GetMVGBusComp Method for a Business Component" on page 211

- "GetNamedSearch Method for a Business Component" on page 212

- "GetPicklistBusComp Method for a Business Component" on page 213

Methods That Control Navigation Flow of Siebel Applications

The following object interface methods allow your configuration to control the navigation flow of a Siebel application:

These object interface methods explicitly specify the view, applet, or control that Siebel CRM displays or makes active. The following items apply for these methods:

■ Sets the active view to the view that you specify.

■ Your configuration cannot call these methods from Browser Script.

■ They are useful only if you access a Siebel object interface in one of the following ways:

■ From Siebel VB

■ From the Mobile Web Client Automation Server

If you access a Siebel object interface through COM Data Control, COM Data Server, or Siebel Java Data Bean, then no Siebel user interface is present.

Siebel CRM stores the properties of a Siebel object in the Siebel repository file (SRF). You cannot use an object interface method in Siebel VB to modify these properties at run time. A business component is an example of a Siebel object.

Methods That Get and Display Information About the Current State

The following object interface methods allow your configuration to use the application object to get information about the current state of properties and functions. This information is useful if your configuration must process rows of data or create query criteria:

■ "CurrencyCode Method for an Application" on page 133

■ "EnableExceptions Method for an Application" on page 134

■ "GetLastErrCode Method for an Application" on page 138

■ "GetLastErrText Method for an Application" on page 139

■ "LoginId Method for an Application" on page 155

■ "LoginName Method for an Application" on page 155

■ "LookupMessage Method for an Application" on page 156

■ "PositionName Method for an Application" on page 161

■ "RaiseError Method for an Application" on page 162

■ "RaiseErrorText Method for an Application" on page 164

■ "SetPositionId Method for an Application" on page 165

■ "SetPositionName Method for an Application" on page 166

Methods That Control Debug Tracing

The following object interface methods allow your configuration to control debug tracing:

■ "Trace Method for an Application" on page 173

■ "TraceOff Method for an Application" on page 175

■ "TraceOn Method for an Application" on page 176

3 Customizing Siebel Object Interfaces

This chapter describes how to customize Siebel object interfaces. It includes the following topics:

- Process of Customizing a Siebel Object Interface on page 29
- Accessing a Siebel Object Interface on page 34
- Customizing Object Interface Events and Extension Events on page 57
- Configuring Error Handling on page 61

Process of Customizing a Siebel Object Interface

To customize a siebel object interface, perform the following tasks:

1. "Determining the Type of Siebel Object Interface You Must Use" on page 29
2. "Setting the Connect String" on page 30
3. "Accessing a Siebel Object Interface" on page 34
4. "Customizing Object Interface Events and Extension Events" on page 57
5. "Configuring Error Handling" on page 61

Determining the Type of Siebel Object Interface You Must Use

This task is a step in "Process of Customizing a Siebel Object Interface" on page 29.

This topic describes how to determine the type of Siebel Object Interface you must use.

To determine the type of Siebel Object Interface you must use

1. in Table 3, examine the Usage column, and then choose the row that most closely matches your requirements.

2 To identify the type of Siebel Object Interface you must use, examine the other columns in Table 3 in the row that you identified in Step 1.

Table 3.   Determining the Type of Siebel Object Interface You Must Use

Usage	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Siebel Java Data Bean
Control the Siebel client from an external application.	Yes	Yes	No	No	No
Access Siebel business objects without using the Siebel client.	No	No	Yes	Yes	Yes
Run objects on the Siebel Server.	Yes	No	Yes	No	Yes
Run objects in the Siebel client in a mobile environment.	No	Yes	No	Yes	No

Use Caution If You Customize a Siebel Object Interface

Oracle does not support the following items:

■ Functions developed through custom programming

■ Specific performance characteristics of third-party software

Oracle defines a Siebel business object or a Siebel object interface at the sole discretion of Oracle. Oracle reserves the right to modify the behavior, properties, and events of a Siebel business object or a Siebel object interface at any time without notice.

CAUTION: Your Siebel application is a Web application or a client and server application that can meet the sales and marketing information requirements of your organization. Use caution if you customize a Siebel application or access it through a Siebel object interface. Only trained, technical professionals must perform this work. Improper use of a Siebel object interface can reduce the performance and reliability of your Siebel application. Test your customization thoroughly before you deploy it.

Setting the Connect String

The *connect string* is a text string that describes the URL that is required to connect to a server component on the Siebel Server. It specifies the protocol and the details of the Client Application Manager service on the Siebel Server. The Siebel client or a program that is external to Siebel CRM must use this string to connect to the Siebel Server.

Format of the Connect String Parameter

The connect string uses the following format:

host="siebel.*transport*.*encryption*.*compression*://host:*port*/*EnterpriseServer*/
AppObjMgr_lang" lang="*lang_code*"

For example:

SiebelApplication.Login "host=""siebel://*host*/*EnterpriseServer*/SCCObjMgr_enu""
"lang="ENU"", "CCONWAY", "CCONWAY"

Table 4 describes how to set each variable in the connect string.

Table 4.   Variable Substitutions You Can Use to Log In to a Siebel Server

Variable	Description
transport	Use the default value, tcpip, or leave empty.
encryption	Use one of the following values: ■ none. This value is the default value. ■ mscrypto. You cannot use mscrypto with Siebel Java Data Bean. ■ rsa. You can use rsa with Siebel Java Data Bean.
compression	Use one of the following values: ■ none. ■ zlib. This value is the default value.
host	Use the name of the computer where you installed the Siebel Server.
port	Enter the number for the SCBroker port. The default value is 2321. Modify this value only if you also modify the default value when you install the Siebel Server. For information about load-balancing with SCBroker, see *Siebel Deployment Planning Guide*, *Siebel System Administration Guide*, and *Siebel Installation Guide* for the operating system you are using.

Table 4.   Variable Substitutions You Can Use to Log In to a Siebel Server

Variable	Description
EnterpriseSer ver	Enter the name of the Siebel Enterprise Server.
AppObjMgr	Enter the name of the Application Object Manager that the Siebel client must access. You can enter a custom server component or one of the following predefined server components: ■ ISSObjMgr_*lang* ■ SCCObjMgr_*lang* ■ SSEObjMgr_*lang* ■ SSVObjMgr_*lang* For more information, see *Siebel System Administration Guide*.

The format of the connect string is optional. You can enter only the transport variable and use a period (.) to separate it from siebel. For example:

    siebel.tcpip://host/siebel/AppObjMgr_*lang*

If you specify any of the other variables, then you must use a period (.) as a placeholder for each variable that you do not specify. For example:

    siebel...zlib://myhost/siebel/SCCObjMgr_enu

Examples of Using the Connect String
This topic includes examples of using the connect string.

Example Connect String for COM Data Control in Server Mode
The following example includes a connect string for COM Data Control that operates in server mode:

```
'COM Data Control : SERVER Mode
lstr = "host=" + """siebel://frashid/Siebel/SSEObjMgr_enu"""
'Format of the connect string is
'"host=" + """siebel://host/enterprise/App. Object Mgr_lang"""
lng = "lang=" + """ENU"""
retval = siebDataCtl.Login(lng + lstr, "username", "password")
```

Example Connect String for COM Data Control in Local Mode
The following example includes a connect string for COM Data Control that operates in Local Mode:

```
'COM Data Control : LOCAL Mode
lstr = "cfg=" + """C:\Siebel\8.1\Client_2\BIN\ENU\siebel.cfg,ServerDataSrc"""
```

```
'Format of the connect string is
'"cfg=" + """Absolute path of the CFG file, DataSource"""
'Datasource = ServerDataSrc or Local or Sample
lng = "lang=" + """ENU"""
retval = siebDataCtl.Login(lng + lstr, "username", "password")
```

If in Local Mode, then COM Data Control must reside on the same computer as the Siebel Mobile Web Client.

Example Connect String for COM Data Control When Using Siebel VB

The following example includes a connect string for COM Data Control that uses Siebel VB. The Char(34) code indicates a double quote:

```
ConnStr = "host =" & char(34) & "siebel://HOST/ENTERPRISE_SERVER/SCCObjMgr_enu/
SIEBEL_SERVER" + char(34) & " Lang = " & char(34) & "LANG" & char(34)
```

Using Load Balancing with the Connect String

You can use Siebel native load balancing across Siebel Servers with the following Siebel object interfaces:

■ COM Data Control that operates in server mode

■ Siebel Java Data Bean

To use load balancing with the connect string

1 Modify the predefined connect string so that it directs requests to an appropriate virtual host.

This host includes specific Siebel Servers. Each Siebel Server includes the required object manager.

2 Specify the path to the file that defines the virtual host.

Connect String That Uses Load Balancing with COM Data Control

A connect string that uses native Siebel load balancing with COM Data Control uses the following format:

```
host="siebel://VirtualHost/EnterpriseServer/AppObjMgr_lang"vhosts="path to
lbconfig.txt"
```

where:

■ *lbconfig.txt* is the file that identifies the virtual hosts.

For more information about the lbconfig.txt file, see *Siebel System Administration Guide*.

Connect String That Uses Load Balancing with Siebel Java Data Bean

A connect string that uses native Siebel load balancing with Siebel Java Data Bean uses the following format:

```
host="siebel://VirtualHost/EnterpriseServer/AppObjMgr_lang"
```

If you use Java code to connect to the Siebel Server, then Siebel CRM reads virtual host definitions from the following property in the siebel.properties file:

```
siebel.conmgr.virtualhosts
```

The siebel.properties file must reside in the classpath of the Java Virtual Machine.

For information about using virtual hosts in the siebel.properties file, see *Transports and Interfaces: Siebel Enterprise Application Integration*.

Example Connect String That Uses Load Balancing

The following example includes a connect string for COM Data Control that operates in server mode in an environment that uses Siebel round-robin load balancing across Siebel Servers:

```
'COM Data Control : Load Balancing
lstr = "host=" + """siebel://VirtualServer1/Siebel/SSEObjMgr_enu""" + "vhosts=" +
"""m:\siebel\admin\lbconfig.txt"""
lng = "lang=" + """ENU"""
retval = siebDataCtl.Login(lng + lstr, "username", "password")
```

where:

VirtualServer1 matches the value in the VirtualServer parameter in the session manager rules in the lbconfig.txt file. For example:

```
"VirtualServer1=1:SiebServA:2321;2:SiebServB:2321;"
```

For information about the lbconfig.txt file, see *Siebel System Administration Guide*.

Accessing a Siebel Object Interface

This task is a step in .

This topic describes how to access a Siebel Object Interface.

To access a Siebel Object Interface

■ To access a Siebel object interface, do one of the following:

- ■ "Allowing Development Environments Access to the Siebel Object Model" on page 35
- ■ "Accessing the Web Client Automation Server" on page 35
- ■ "Accessing the Mobile Web Client Automation Server" on page 37
- ■ "Accessing the Siebel COM Interface" on page 39
- ■ "Accessing the COM Data Server" on page 41
- ■ "Accessing the COM Data Server with Microsoft Visual Studio" on page 43
- ■ "Accessing COM Data Control" on page 47

■   "Accessing the Siebel Java Data Bean" on page 49

These topics assume you use Microsoft Visual Basic to access the interface.

Allowing Development Environments Access to the Siebel Object Model

This topic describes how to allow an interactive development environment, such as the Microsoft Visual Studio, to have access to the Siebel Object Model.

To allow a development environment access to the Siebel Object Model

1   On the Siebel application that users will use to login, for example, Siebel Call Center; enable the Web Client Automation Server by setting the EnableWebClientAutomation component parameter to TRUE.

2   Restart the Siebel Component that was modified in Step 1.

3   From the development machine, using Internet Explorer, login to Siebel Application in high interactivity mode.

This will download the Siebel Web Client Automation Server and register it in the Windows Registry.

4   Start Microsoft Visual Studio.

5   Choose Standard EXE.

6   Choose the Project menu, and then the References menu item.

7   In the list box, choose SiebelHTML 1.0 Type Library.

8   Add the required code.

For more information, see "Examples of Accessing the Web Client Automation Control" on page 36.

Accessing the Web Client Automation Server

This topic describes how to access the Web Client Automation Server.

NOTE: The Web Client Automation Server only works in high interactivity mode.

In order for the Web Client Automation Server to accept requests from an external ActiveX control, the external control must provide a Session Random Number (SRN). If the Web Client Automation Server recognizes the SRN, then it executes the request. If the Web Client Automation Server does not recognize the SRN, the request is ignored. The SRN is unique to a single browser session, so it cannot be used indefinitely: once the user initiates a logout sequence, the SRN is no longer valid.

Examples of Accessing the Web Client Automation Control

This topic lists multiple examples relating to accessing the Web Client Automation Control.

Example of Implementing SetSiebelToken on the External ActiveX Control

The following example includes the code you use to implement SetSiebelToken on the external ActiveX control:

NOTE: This is part of the source for the external ActiveX control.

```
Private m_sSRN as String

Public Sub SetSiebelToken(sSRN as String)
 m_sSRN = sSRN
End Sub


Private Function GetSiebelToken() as String
 GetSiebelToken = m_sSRN
End Function
```

Example of Instantiating a New External ActiveX Control and Passing an SRN

The following example includes the code you use to instantiate a new external ActiveX control and pass an SRN:

NOTE: This example uses Siebel browser script.

```
var MyObject = new ActiveXObject("My.Control");
MyObject.SetSiebelToken(theApplication().GetSRN());
```

Example of Connecting to an Instantiated ActiveX Control and Passing an SRN

The following example includes the code you use to connect an existing instantiated ActiveX control and pass an SRN.

NOTE: This example uses Siebel browser script.

```
var MyObject = GetObject(, "My.Control");
MyObject.SetSiebelToken(theApplication().GetSRN());
```

Example of Accessing the Web Client Automation Control from the External ActiveX Control

Figure 5 includes the code you use to call the Siebel Web Client Automation Control from the external ActiveX control.

```
Sub MyFunctionThatCallsSiebel(some arguments)
 On Error Goto ErrorHandler
 Dim oSiebel as SiebelHTMLApplication
 Dim oBS As SiebelService
 Dim oPSInput As SiebelPropertySet, oPSOutput As SiebelPropertySet

 Set oSiebel = CreateObject("Siebel.Desktop_Integration_Application.1")
 If (Not oSiebel Is Nothing) Then
  Set oBS = oSiebel.GetService("My Service")
  Set oPSInput = oSiebel.NewPropertySet
  Set oPSOutput = oSiebel.NewPropertySet
  With (oPSInput)
   oPSInput.SetProperty "SRN", GetSiebelToken()          (1)
   oPSInput.SetProperty "My Property", "My Value"
    ' Repeat for other properties as needed
  End With
  oService.InvokeMethod "My Method", oPSInput, oPSOutput
   ' Work with oPSOutput as needed
 End If

ErrorHandler:
 Set oPSOutput = Nothing
 Set oPSInput = Nothing
 Set oService = Nothing
 Set oSiebel = Nothing
End Sub
```

Figure 5.   Code Used to Call Siebel Web Client Automation Control from the External ActiveX Control

Explanation of Callouts
The code includes the following callout:

1   Sets an input PropertySet property to pass in the trusted SRN.

NOTE: SRN does not need to be explicitly added as an input property on a business service. It is implicitly available.

Accessing the Mobile Web Client Automation Server

This topic describes how to access the Mobile Web Client Automation Server. For more information, see "Mobile Web Client Automation Server" on page 15.

To access the Mobile Web Client Automation Server

1   Install the Siebel Mobile Web Client.

Siebel CRM installs the Mobile Web Client Automation Server by default when you install the Siebel Mobile Web Client.

2   Start Microsoft Visual Basic.

3 Choose Standard EXE.

4 Choose the Project menu, and then the References menu item.

5 In the list box, choose Mobile Web Client Automation Server.

6 Add the required code.

For more information, see "Example of Accessing the Mobile Web Client Automation Server" on page 38.

Example of Accessing the Mobile Web Client Automation Server

The following example includes the code you use in Microsoft Visual Basic 6.0 to access the Mobile Web Client Automation Server:

```
Private Sub Command1_Click()
'Siebel Application Object
Dim siebApp As SiebelWebApplication
Dim siebBusObj As SiebelBusObject
Dim siebBusComp As SiebelBusComp
Dim siebSvcs As SiebelService
Dim siebPropSet As SiebelPropertySet
Dim bool As Boolean
Dim errCode As Integer
Dim errText As String
Dim connStr As String
Dim lng As String
'Create The Siebel WebApplication Object
Set siebWebApp = CreateObject("TWSiebel.SiebelWebApplication.1")

If Not siebWebApp Is Nothing Then


    'Create A Business Object
    Set siebBusObj = siebWebApp.GetBusObject("Contact")
    If Not siebBusObj Is Nothing Then
        'Create a Business Component
        Set siebBusComp = siebBusObj.GetBusComp("Contact")

    Else
        errCode = siebWebApp.GetLastErrCode
        errText = siebWebApp.GetLastErrText
        siebWebApp.RaiseErrorText "Business Object Creation failed." & errCode & "::" &
errText;

    End If

    'Create A New Property Set
    Set siebPropSet = siebWebApp.NewPropertySet
    If Not siebPropSet Is Nothing Then
        Set siebPropSet = Nothing

    Else
```

```
        errCode = siebWebApp.GetLastErrCode
        errText = siebWebApp.GetLastErrText
        siebWebApp.RaiseErrorText "Property Set Creation failed." & errCode & "::" &
errText;
End If

'Get A Siebel Service
Set siebSvcs = siebWebApp.GetService("Workflow Process Manager")
If Not siebSvcs Is Nothing Then
    Set siebSvcs = Nothing
Else
    errCode = siebWebApp.GetLastErrCode
    errText = siebWebApp.GetLastErrText
    siebWebApp.RaiseErrorText "Could not Get Siebel Service." & errCode & "::" &
errText;
End If

If Not siebBusComp Is Nothing Then
    Set siebBusComp = Nothing
End If

If Not siebBusObj Is Nothing Then
    Set siebBusObj = Nothing
End If

    Set siebWebApp = Nothing
End If

End Sub
```

Accessing the Siebel COM Interface

This topic describes how to access the Siebel COM Interface.

To access the Siebel COM Interface

1 in the Siebel application configuration (CFG) file, set the EnableOLEAutomation parameter to TRUE.

2 Use the object browser of your COM programming tool to determine the correct format for the object interface method.

   For more information, see "Example of an Object Browser" on page 40.

Example of an Object Browser

Figure 6 includes an example of the object browser in Microsoft Visual Basic 5.0, which is a COM programming tool. The format window at the bottom displays the *method signature* for the method chosen in the Object Browser window. This signature includes information about the method, such as the inputs, data types, and the information the method returns.
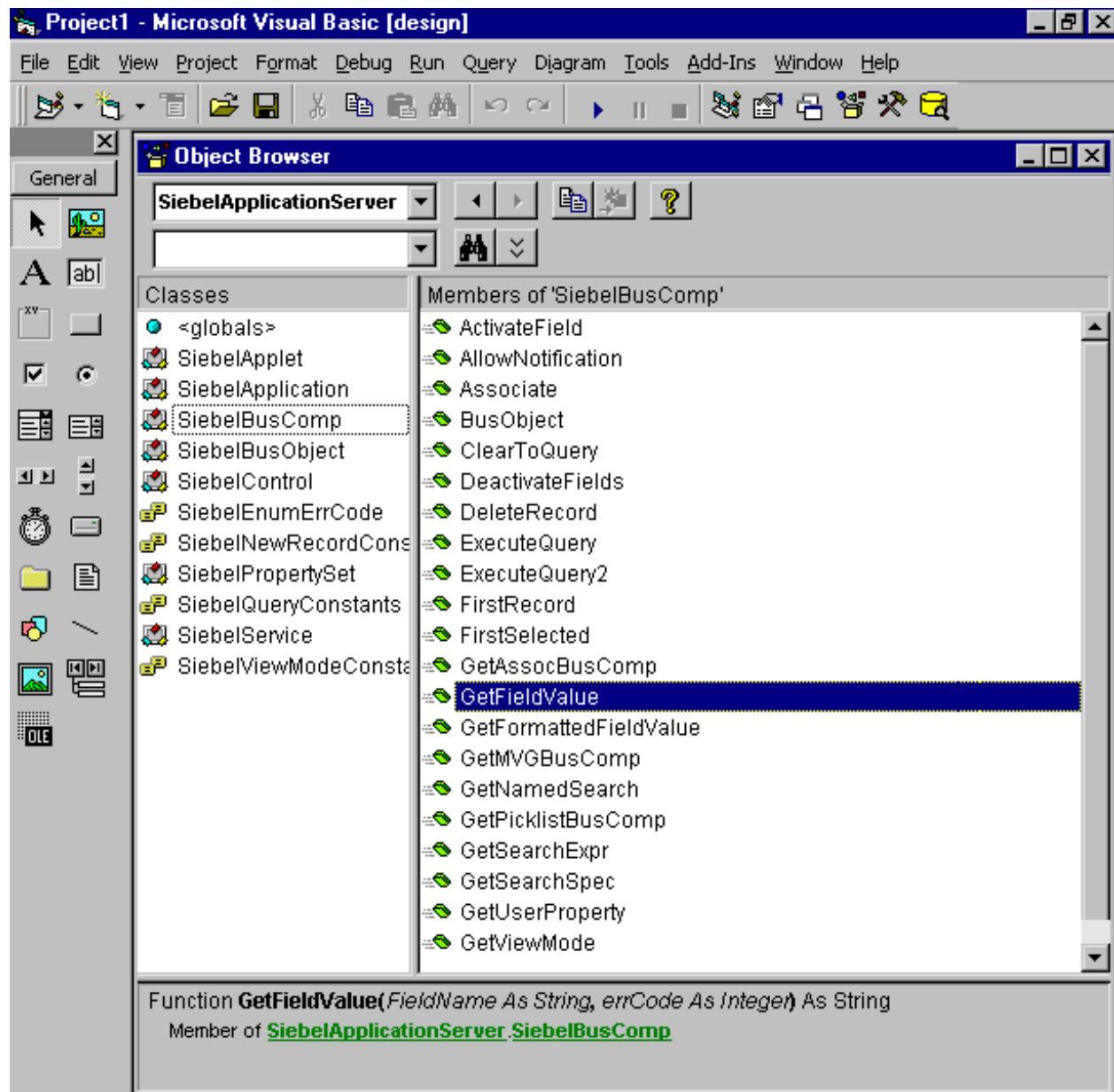
Figure 6.   Example of an Object Browser in a COM Programming Tool

Accessing the COM Data Server

This topic describes how to access the COM Data Server. For more information, see "COM Data Server" on page 16.

To access the COM Data Server

1 Install the Siebel Mobile Web Client.

Siebel CRM installs the COM Data Server by default when you install the Siebel Mobile Web Client.

2 In the Siebel application configuration (CFG) file, set the DataSource parameter to the Siebel database where Siebel CRM must connect.

3 Start Microsoft Visual Basic.

4 Choose Standard EXE.

5 Choose the Project menu, and then the References menu item.

6 In the References dialog box, in the Available References window, click Siebel Data BusObject Interfaces.

Do not add a check mark to the Siebel Data BusObject Interfaces.

7 In the Siebel Data BusObject Interfaces section, note the name of the folder that contains the sobjsrv.tlb file.

8 In the Available References window, make sure the Siebel Data BusObject Interfaces item contains a check mark, and then click OK.

9 Add the required code.

For more information, see "Example of Accessing the COM Data Server" on page 41.

Example of Accessing the COM Data Server

The following example includes the code you use in Microsoft Visual Basic 6.0 to access the COM Data Server. You must write and run this code outside of Siebel Tools. For example, in Microsoft Visual Basic:

```
Private Sub Command1_Click()
'Siebel Application Object
Dim siebApp As SiebelApplication
Dim siebBusObj As SiebelBusObject
Dim siebBusComp As SiebelBusComp
Dim siebSvcs As SiebelService
Dim siebPropSet As SiebelPropertySet
Dim bool As Boolean
Dim errCode As Integer
Dim errText As String
Dim connStr As String
Dim lng As String
Dim cfgLoc As String
```

```
ChDrive "C"
ChDir "C:\Server\siebsrvr\bin"

'Create The COM Data Server Object
Set siebApp = CreateObject("SiebelDataServer.ApplicationObject")

If Not siebApp Is Nothing Then

'''COM Data Server
cfgLoc = " C:\Siebel\8.1\Server\BIN\ENU\siebel.cfg,ServerDataSrc"
siebApp.LoadObjects cfgLoc, errCode
If errCode = 0 Then
   'Log in to the Siebel Server
   siebApp.Login "username", "password", errCode
   If errCode = 0 Then
      'Creat A Business Object
      Set siebBusObj = siebApp.GetBusObject("Contact", errCode)
      If errCode = 0 Then
         'Create a Business Component
         Set siebBusComp = siebBusObj.GetBusComp("Contact")
   Else
      errText = siebApp.GetLastErrText
      siebApp.RaiseErrorText("Business Object Creation failed: " & errCode & "::" &
errText);
   End If

   'Create A New Property Set
   Set siebPropSet = siebApp.NewPropertySet(errCode)
   If errCode = 0 Then
      Set siebPropSet = Nothing
   Else
      errText = siebApp.GetLastErrText
      siebApp.RaiseErrorText("Property Set Creation failed: " & errCode & "::" &
errText);
   End If

   'Get A Siebel Service
   Set siebSvcs = siebApp.GetService("Workflow Process Manager", errCode)
   If Not siebSvcs Is Nothing Then
      Set siebSvcs = Nothing
   Else
      errText = siebApp.GetLastErrText
      siebApp.RaiseErrorText("Could not Get Siebel Service: " & errCode & "::" &
errText);
   End If

   If Not siebBusComp Is Nothing Then
      Set siebBusComp = Nothing
   End If
   If Not siebBusObj Is Nothing Then
      Set siebBusObj = Nothing
   End If
Else
    errText = siebApp.GetLastErrText
    siebApp.RaiseErrorText("Login Failed: " & errCode & "::" & errText);
```

```
      End If
   Else
      errText = siebApp.GetLastErrText
      siebApp.RaiseErrorText("Load Objects Failed: " & errCode & "::" & errText);
   End If

   Set siebApp = Nothing

   End If

   End Sub
```

Accessing the COM Data Server with Microsoft Visual Studio

This topic describes how to create a simple COM client in Microsoft Visual C++ and the Microsoft Foundation Class (MFC) library that accesses the Siebel Data Server.

To access the COM Data Server with Microsoft Visual Studio

1 In Microsoft Visual C++, choose the File menu, New, and then the Project menu item.

2 Choose the MFC AppWizard (exe) project type.

3 In the Project name field, enter SiebelCOM, and then click OK.

4 In the MFC AppWizard, choose the Dialog-based option and then click Next.

5 In the What Other Support Would You Like to Include frame, do the following:

  a Make sure the Automation option contains a check mark.

  b Make sure the ActiveX Controls does not contain a check mark.

  c Click Next.

  d Click Next.

6 Click Finish, and then click OK.

The Application Wizard creates the MFC code that you use for this project, including the headers and libraries that COM automation requires. For more information about the MFC libraries, see the documentation for Microsoft MSDN Visual Studio.

7   Modify the new dialog box.

Microsoft Visual C++ displays a new dialog box. To resize and modify the text in this dialog box, right-click the label in the dialog box and edit the properties. Modify the dialog box so that it resembles the following illustration.
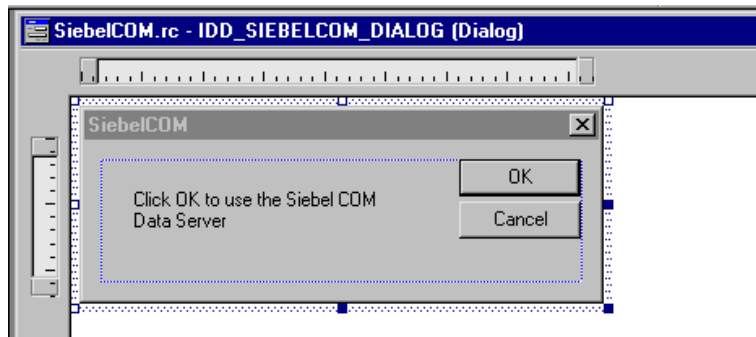
8   Choose the View menu, ClassWizard, and then the Automation menu item.

9   Click Add Class, and then click From a Type Library.

10 Navigate to the *SIEBSRVR_ROOT*\bin folder, and then choose sobjsrv.tlb.

11 In the Confirm Classes dialog box, make sure all Siebel classes are chosen, click OK, and then click OK again to close the Class Wizard.

12 Add code to communicate with the Siebel COM Server.

a   In the workspace window, click the FileView tab.

b   Expand the Source Files folder and the Header Files folder.

c   Double-click the SiebelCOMDlg.h file.

d   In the code window, add the following code to the SiebelCOMDlg.h file. Add only the code that uses bold typeface:

```
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

#include "sobjsrv.h" // Include Siebel wrapper classes

class CSiebelCOMDlgAutoProxy;

/////////////////////////////////////////////////////////
// CSiebelCOMDlg dialog

class CSiebelCOMDlg : public CDialog{
   DECLARE_DYNAMIC(CSiebelCOMDlg);
   friend class CSiebelCOMDlgAutoProxy;
   SiebelApplication sApp; // Declare Siebel object
```

```
//Construction
public:
   CSiebelCOMDlg(CWnd* pParent = NULL);  //standard constructor
   virtual  ~CSiebelCOMDlg();
```

e Choose Open from the File menu, and then choose the SiebelCOMDlg.cpp file.

f Add the following code to the OnInitDialog procedure. Add only the code that uses bold typeface:

```
CDialog::OnInitDialog();

...

// TODO: Add extra initialization here
// Start the Siebel Data Server
if (!sApp.CreateDispatch(_T("SiebelDataServer.ApplicationObject)))
{
   AfxMessageBox("Cannot start Siebel Data Server.");
EndDialog(-1); // Fail
} else
{
   AfxMessageBox("Siebel Data Server initialized.");
}

return TRUE; // Return TRUE unless you make a control active
...
```

g In the same file, add the following code to the OnOK procedure.

To add this code correctly, do the following:

❏ Make sure that the line that begins with sApp.LoadObjects references the location of the Siebel application configuration (CFG) file you intend to use.

❏ In the line that begins with sApp.Login, make sure you use a valid logon name and password.

❏ Add only the code that uses bold typeface.

```
void CSiebelCOMDlg::OnOK()
{

   short sErr;

   // Load configuration file
   // Make sure that the following line references the correct file
   sApp.LoadObjects(C:\Siebel\8.1\Server\BIN\ENU\siebel.cfg", &sErr);
   if(sErr)
   {
      AfxMessageBox("LoadObject failed.");
      return;
   } else
   {
      AfxMessageBox("CFG file loaded.");
   }
```

```
// Log in as SADMIN
sApp.Login("SADMIN", "SADMIN", &sErr);
if(sErr)
{
   AfxMessageBox("Login failed.");
   return;
} else
{
   AfxMessageBox("Logged in to Siebel database.");
}

// Get Account business object
LPDISPATCH lpdBo;
lpdBo = sApp.GetBusObject("Account", &sErr);
if(sErr)
{
   AfxMessageBox("GetBusObject failed.");
   return;
} else
{
   AfxMessageBox("Account business object returned.");
}
SiebelBusObject Bo(lpdBo);

// Get Account business component
LPDISPATCH lpdBc;
lpdBc = Bo.GetBusComp("Account", &sErr);
if(sErr)
{
   AfxMessageBox("GetBusComp failed.");
   return;
} else
{
   AfxMessageBox("Account business component returned.");
}
SiebelBusComp Bc(lpdBc);

// Get the name of the first account
if (sErr) return;
Bc.ClearToQuery(&sErr);
if (sErr) return;
Bc.SetSearchSpe("Name", "*", &sErr);
if (sErr) return;
Bc.ExecuteQuery(ForwardOnly, &sErr);
if (sErr) return;
Bc.FirstRecord(&sErr);
if (sErr) return;

// Display the account name in a message box
CString csAcctName;
csAcctName = Bc.GetFieldValue("Name", &sErr);
AfxMessageBox(csAcctName);
```

```
        Bc = null;
        lpdBc = null;
        Bo = null;
        lpdBo = null;

        return;

        if (CanExit())
           CDialog::OnOK();

    }
```

13 Test your work:

a Start the Siebel client.

Make sure you use the same Siebel application configuration (CFG) file and login arguments that you specified in the code.

b Navigate to the Accounts screen, and then the All Accounts view.

c Verify that at least one account is visible in the Account list applet.

If at least one account is not visible, then create one.

d Exit the Siebel client.

e Open the Siebel application configuration (CFG) file you specified in the code and make sure the DataSource parameter indicates the correct Siebel database source.

f In Microsoft Visual C++, choose the Build menu, and then the SiebelCOM.exe menu item.

If Microsoft Visual C++ displays an error or warning in the output window, then correct the error and repeat this step.

g Choose the Build menu, and then the Execute SiebelCOM.exe menu item.

h Wait for Microsoft Visual C++ to display the following message:

```
    Siebel Data Server initialized.
```

i Click OK.

The Siebel application displays the following series of messages:

```
    CFG file loaded.
    Logged in to Siebel database.
    Account business object returned.
    Account business component returned.
```

The Siebel application displays the name of the first account in the All Accounts view.

Accessing COM Data Control

This topic describes how to access COM Data Control. A call to COM Data Control is *in process*. For more information, see "How Siebel CRM Uses Memory and Resources with the Mobile Web Client Automation Server" on page 16.

To access COM Data Control

1 Install COM Data Control.

Use the Siebel Enterprise Server Installer. Make sure the EAI Siebel Connectors option contains a check mark. For more information, see the *Siebel Installation Guide* for the operating system you are using.

2 Start Microsoft Visual Basic.

3 Choose Standard EXE.

4 Choose the Project menu, and then the References menu item.

In the References dialog box, in the Available References window, make sure the Siebel Business Object Interfaces Type Library item contains a check mark.

5 To open the Object Browser, click OK.

6 Determine the correct format for the object interface method.

You must use the CreateObject method and the Login method. You cannot use an object interface method that returns an active Siebel object because no Siebel objects are currently active. You must use your own Siebel objects.

7 Verify that you can view the Siebel objects.

8 Add the required code.

For more information, see "Example of Accessing COM Data Control" on page 48.

Example of Accessing COM Data Control

The following example includes the code you use in Microsoft Visual Basic 6.0 to access COM Data Control:

```
Sub CreateDataControl()
Dim errCode As Integer
Set SiebelApplication = CreateObject("SiebelDataControl.SiebelDataControl.1")
SiebelApplication.Login "host=""siebel://hostname/EnterpriseServer/AppObjMgr""",
"CCONWAY", "CCONWAY"
errCode = SiebelApplication.GetLastErrCode()
If errCode <> 0 Then
    ErrText = SiebelApplication.GetLastErrText
    SiebelApplication.RaiseErrorText ErrText;
    Exit Sub
End If
set OpptyBO = SiebelApplication.GetBusObject("Opportunity",errCode)
set OpptyBC = OpptyBO.GetBusComp("Opportunity", errCode)
End Sub
```

To determine values to substitute for the variables in the login string, see "Setting the Connect String" on page 30.

Example of Using Siebel Server ASP Script to Access COM Data Control

To set off an ASP script in HTML code, you use the following format:

■ To indicate the beginning of the ASP script, you use the less than symbol and the percent symbol (<%).

■ To indicate the end of the ASP script, you use the percent symbol and the greater than symbol (%>).

The following example code starts COM Data Control from a Siebel Server ASP script:

```
<%

    Dim SiebelApplication, BO, BC, ConnStr, logstat
    Dim strLastName, strFirstName, errCode, errText

        Set SiebelApplication = CreateObject("SiebelDataControl.SiebelDataControl.1")

    ' Test to see if object is created
    If IsObject(SiebelApplication) = False then
        Response.Write "Unable to initiate Siebel Session.
    Else
        connStr = "host=" & Chr(34) & "siebel.tcpip.none.none://hostname:2321/
EntServer/ObjMgr" & Chr(34) & " lang=" & Chr(34) & "lang" & Chr(34)
        logstat = SiebelApplication.Login ConnStr, "SADMIN", "SADMIN"

    response.write("Login Status: " & logstat)
    Set BO = SiebelApplication.GetBusObject("Employee")
    Set BC = BO.GetBusComp("Employee")
End If

%>
```

Accessing the Siebel Java Data Bean

A Java client that uses the Siebel Java Data Bean to connect to the Siebel Server requires JAR files. These files allow the Java language to access the objects and methods of the Siebel Object Interface. These files are specific to the version of the Siebel application. Do not use these JAR files with other versions. For more information, see "About the Siebel Java Data Bean Object Interface" on page 12.

To access the Siebel Java Data Bean

1 Add the following JAR files to the CLASSPATH:

■ Siebel.jar

■ SiebelJI_lang.jar

2 To install the Siebel Java Data Bean interface, do one of the following:

■ Use the Siebel Enterprise Server Installer. Make sure the EAI Siebel Connectors option contains a check mark. For more information, see the *Siebel Installation Guide* for the operating system you are using.

■ Install Siebel Tools. The Oracle Universal Installer installs the Siebel Java Data Bean interface by default when you install Siebel Tools.

3 Start a new SiebelDataBean Java object.

4 To call the Login method for the object you started in Step 3, use the following code:

```
SiebelDataBean l_sdb = new SiebelDataBean();

l_sdb.login(<parameters>);
```

You must use the Login method. You cannot use an object interface method that returns an active Siebel object because no Siebel objects are currently active. You must use your own Siebel objects. For more information, see Step 2 on page 39.

Example of Accessing the Siebel Java Data Bean

The following example code accesses the Siebel Java Data Bean. You can use a Java IDE to compile and run this code:

```java
import com.siebel.data.*;
import com.siebel.data.SiebelException;

public class DataBeanDemo
{
    private SiebelDataBean m_dataBean = null;
    private SiebelBusObject    m_busObject = null;
    private SiebelBusComp       m_busComp = null;

    public static void main(String[] args)
    {
        DataBeanDemo demo = new DataBeanDemo();
    }

    public DataBeanDemo()
    {
        try
        {
            // instantiate the Siebel Java Data Bean
            m_dataBean = new SiebelDataBean();

            // log in to the Siebel Server
            // SiebelServerhost = the name or IP address of your Siebel Server
            // SCBPort = listening port number for the SCBroker component (default 2321)
            m_dataBean.login("Siebel://SiebelServerhost:SCBPort/enterpriseServer/
             AppObjMgr_enu", CCONWAY, CCONWAY, "enu");

            // get the business object
            m_busObject = m_dataBean.getBusObject("Opportunity");

            // get the business component
            m_busComp = m_busObject.getBusComp("Opportunity");

            // log off
            m_dataBean.logoff();
        }

        catch (SiebelException e)
        {
            System.out.println(e.getErrorMessage());
```

```
            }
        }
    }
```

Using Single Sign-on (SSO) with Siebel Java Data Bean

If you use single sign-on (SSO) with Siebel Java Data Bean, then you must include the following items in the login:

■ Login ID of an employee as the username.

■ The value of the TrustToken parameter in the connect string. To determine the value for the TrustToken, examine the TrustToken parameter in the Siebel application configuration (CFG) file. For more information, see "Setting the Connect String" on page 30.

For example:

```
m_dataBean.login("Siebel://gatewayserver:2321/enterpriseServer/SCCObjMgr_enu",
SADMIN, HELLO,"enu");
```

where:

■ SADMIN is an employee.

■ The TrustToken parameter is HELLO in the LDAPSecAdpt section of the Siebel application configuration (CFG) file.

Customizing the Parameters a Third-Party Application Uses to Connect Through the Siebel Java Data Bean

You can customize the parameters that a third-party application uses when it connects to a Siebel application through the Siebel Java Data Bean.

To customize the parameters a third-party application uses to connect through the Siebel Java Data Bean

1 Open the siebel.properties file.

This file is located in your *classpath*, which is an operating system environment variable that a Java program references. The siebel.properties file can exist in any location. The CLASSPATH environment variable must include an entry for this file so that the Java Virtual Machine can find the file when it starts.

2 Set the properties.

For more information, see "Properties of the Siebel Properties File" on page 52.

Properties of the Siebel Properties File

Table 5 describes the properties of the siebel.properties file.

Table 5.   Properties of the Siebel Properties File

Property Type	Property	Description
Siebel Connection Manager	siebel.conmgr.txtimeout	The transaction timeout in milliseconds. The default value is 600000, which is 10 minutes. The maximum value is 2,147,483,647, which is approximately 25 days.
	siebel.conmgr.poolsize	The connection pool size. For more information, see "Determining the Total Number of Open Connections" on page 53.
	siebel.conmgr.sesstimeout	The transaction timeout in seconds on the Siebel client. The default value is 2700, which is 45 minutes. The maximum value is 2,147,483,647, which is approximately 68 years.
	siebel.conmgr.retry	The number of open session retries. The default value is 3.
	siebel.conmgr.jce	Sets the Java Cryptography Extension (JCE): ■ To use JCE, set the value to 1. ■ To not use JCE, set the value is 0. For more information, see "Encrypting Communication Between the Java Data Bean and the Siebel Server" on page 55.
Siebel created code for Java EE Connector Architecture and Java Data Bean	siebel.connection.string	The Siebel connection string.
	siebel.user.name	The user name to log in to the Object Manager.
	siebel.user.password	The password to log in to the Object Manager.
	siebel.user.language	The preferred language for the user.
	siebel.user.encrypted	Determines if Siebel CRM encrypts the username and password.
	siebel.jdb.classname	The default Java Data Bean (JDB) classname.
Java System Properties	file.encoding	The character encoding on the Siebel client. For example, cp1252, utf8, unicodeBig, or cp942. Java system properties are not Siebel properties.

Example of the Siebel Properties File

The following code is an example of the siebel.properties file:

```
siebel.connection.string = siebel.tcpip.rsa.none://test.siebel.com/siebel/
sseobjmgr_enu/test

siebel.user.name          = User1

siebel.user.password      = password

siebel.user.language      = enu

siebel.user.encrypted     = false

siebel.conmgr.txtimeout   = 3600

siebel.conmgr.poolsize    = 5

siebel.conmgr.sesstimeout = 300000

siebel.conmgr.retry       = 5

siebel.conmgr.jce         = 1
```

Determining the Total Number of Open Connections

The connection pool maintains a set of connections to a specific server process. The default value for the siebel.conmgr.poolsize property is 2. The maximum value is 500.

The siebel.conmgr.poolsize property and the Min MT Server parameter on the object manager determine the total number of open connections. Each MT server process is a Windows process that includes a connection pool. The total number of open connections is the value in the siebel.conmgr.poolsize property multiplied by the value in the Min MT Server parameter.

For example, if the siebel.conmgr.poolsize is 2, and if the Min MT Server parameter is 3, then the total number of open connections is six.

Customizing Character Encoding for the Siebel Java Data Bean

The character encoding of the Siebel Server and the character encoding of the Siebel client must be the same. This allows the Siebel client and the Siebel Server to communicate correctly. If the Siebel client and the Siebel Server default character encoding cannot be the same, then you can modify the Siebel client character encoding.

To customize character encoding for the Siebel Java Data Bean

■   To set the file.encoding system property to the proper character encoding, do one of the following:

■   Set it for the entire Java Virtual Machine on the command line. For example:

```
java -Dfile.encoding=ascii java_application
```

■   Set it in the environment variable. For more information, see your particular Java Virtual Machine.

■   Set it for a particular Java component. Add the following line to the Java component:

```
System.setProperty("file.encoding", CodePageValue);
```

where:

❑ *CodePageValue* is a Siebel value that specifies character encoding for the Java Data Bean.

Table 6 lists character encoding mappings you can use for the Java Data Bean. The Siebel Value column contains the codes you can specify in the CodePageValue variable.

Table 6. Character Encoding Mappings You Can Use for the Java Data Bean

Java Value	Siebel Value
ascii	1
cp1252	1252
iso8859_1	1252
iso8859-1	1252
unicodebig	1201
unicodelittle	1200
utf8	65001
big5	950
cp942	932
cp942c	932
cp943	932
cp943c	932
cp949	949
cp949c	949
cp950	950
cp1250	1250
cp1251	1251
cp1253	1253
cp1254	1254
cp1255	1255
cp1256	1256
cp1257	1257
cp1258	1258
gbk	936
ms874	874

Table 6.   Character Encoding Mappings You Can Use for the Java Data Bean

Java Value	Siebel Value
ms932	932
ms936	936
ms949	949
ms950	950
sjis	932
tis620	874

Encrypting Communication Between the Java Data Bean and the Siebel Server

To encrypt communication between the Siebel Java Data Bean and the Siebel Server, you can use the Rivest, Shamir and Adleman (RSA) encryption libraries. For information about platforms you can use with encryption, see *Siebel System Requirements and Supported Platforms* on Oracle Technology Network.

To encrypt communication between the Siebel Java Data Bean and the Siebel Server

1   Enable encryption in the Object Manager server component that you use for the communication between the Java Data Bean and the Siebel Server.

For more information, see *Siebel System Administration Guide*.

2   Set the encryption parameter of the connect string in the Siebel Java Data Bean to rsa.

For example:

si ebel . tcpi p. rsa. none: *//gateway/enterprise/ObjMgr*

where:

■   *gateway* is the name of the gateway

■   *enterprise* is the name of the enterprise

■   *ObjMgr* is the name of the Object Manager

Encrypting Communication on a Platform That the RSA Libraries do Not Support

To use encryption on a platform that the RSA libraries do not support, Oracle uses the Java Cryptography Extension (JCE) v1.2.1 specification. JCE provides a way to encrypt, create the encryption key, create the key agreement, and handle Message Authentication Code. With JCE, you can use some other qualified cryptography library as a service provider. For information about developer resources for Java technology, see the following Web site:

http://www.oracle.com/technetwork/java/java-sun-com-138872.html

To encrypt communication on a platform that the RSA libraries do not support

1 Download and install the JCE v1.2.1 software, policy files, and documentation.

For information about installing and configuring your Java Virtual Machine for use with JCE, see the following Web site:

http://java.sun.com/products/archive/jce/

Note that you can only use a static specification of JCE providers with the Siebel Java Data Bean.

2 Modify the java.security file to specify your provider of choice.

3 Make sure the classpath variable includes the necessary provider JAR files.

4 Set the siebel.conmgr.jce property in the siebel.properties file to 1.

Login Errors You Might Encounter When You Use the Siebel Java Data Bean

The Siebel Java Data Bean might return a login error that is similar to the following:

Siebel Exception thrown invoking login Method. Code--1. Message-Logon request 75 was abandoned after 2ms connection.

Any of the following items can cause this error:

■ An Object Manager process is down.

■ A hardware reset is required. For example, Object Manager hardware, router, switch, and so forth.

■ There is a problem with an operating system setting or the operating system network.

■ There is a network failure.

■ There is a network address translation timeout.

Using the Siebel Java Data Bean with Multiple Threads

Multiple threads of a single process must not access a common instance of the Siebel Java Data Bean. If a process with multiple threads must use the Siebel Java Data Bean, then each thread must create a separate instance of the Siebel Java Data Bean.

Do not reuse an instance of any other object that the Siebel Java Data Bean makes available across multiple threads of the same process. This requirement includes the following objects:

■ SiebelBusObject

■ SiebelBusComp

■ SiebelService

■ SiebelPropertySet

CAUTION: You must configure Siebel CRM to create one instance of the Siebel Java Data Bean for each thread that must use it. If a thread gets Siebel Java Data Bean Objects, then do not configure Siebel CRM to share these objects with any other thread.

Customizing Object Interface Events and Extension Events

This topic describes object interface events and extension events. It includes the following topics:

- "Overview of Object Interface Events and Extension Events" on page 57
- "Format of the Object Interface Event" on page 58
- "Customizing the Outcome of an Object Interface Event" on page 58
- "Customizing How Siebel CRM Continues an Operation" on page 58
- "Using Tracing to Determine When an Event Occurs" on page 60

For more information, see the following topics:

- "Applet Events" on page 109
- "Application Events" on page 179
- "Business Component Events" on page 262

Overview of Object Interface Events and Extension Events

An *object interface event* is a type of object interface method that Siebel Engineering creates. A Siebel object includes a set of events that correspond to different points of execution during the lifetime of the object. An event acts as a placeholder in this Siebel object. It replies to a method that executes on the object.

Some object interface events allow you to associate custom code with a Siebel application. This code is available in Server Script or Browser Script. If the Siebel application starts the event, then Siebel CRM calls the custom code and the predefined Siebel code that is associated with the event.

You can use the following types of object interface events:

- **Preoperation event.** Occurs before the predefined Siebel operation runs. The PreDeleteRecord event is an example of a preoperation event. This event occurs before the DeleteRecord event occurs. To modify the behavior of a predefined Siebel application, you can use a preoperation event. For example, to perform custom validation on a record that Siebel CRM is about to delete, you can use the PreDeleteRecord event. If the validation fails, then you can instruct Siebel CRM to cancel the DeleteRecord operation.

- **Postoperation event.** Starts after Siebel CRM finishes executing the preoperation event. The DeleteRecord event is an example of a postoperation event. For example, Siebel CRM starts the DeleteRecord event after it finishes executing the PreDeleteRecord event. The postoperation event handler is rarely scripted, but you can use it for some postoperation events, such as posting a notice to a log if the event completes successfully.

Format of the Object Interface Event

The object interface event uses the following format:

■ *ObjectReference_EventName* (*arguments*) As *RetValu*e

where:

■ *ObjectReference* is the variable name of the object where Siebel CRM calls the event.

■ *EventName* is the event that Siebel CRM calls.

Customizing the Outcome of an Object Interface Event

A preoperation event handler exists for every Siebel operation event handler. You typically place a script in the preoperation event. The PreInvokeMethod event results in the most important outcome. In a PreInvokeMethod event, you can call an object interface method that substitutes the predefined Siebel code.

To customize the outcome of an object interface event

■ Attach a script to the preoperation event handler.

Customizing How Siebel CRM Continues an Operation

This topic describes how to customize the way Siebel CRM continues an operation.

To customize how Siebel CRM continues an operation

■ To process data before the default event method runs, set the return value for this predefined event to ContinueOperation.

The return value for a preoperation event is ContinueOperation. It configures the calling Siebel object to continue processing the remaining operations that Siebel CRM associates with the event.

If you handle a custom method in a preevent, then that event must return CancelOperation or you must handle the custom method somewhere in the process. For important caution information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Caution About Using the Cancel Operation Event Handler

Including the CancelOperation return value configures the Siebel application to cancel the remaining operations that Siebel CRM associates with the event.

CAUTION: If you define a custom object interface method, then you must include the CancelOperation return value. If you do not, then Siebel CRM issues an unknown method name error.

CancelOperation does not stop the code in a script that follows CancelOperation, but it does prevent
Siebel CRM from running any predefined code that is associated with the method or event that is
running. If you handle the method or event entirely through scripting, and if you must prevent the
predefined code from executing, then the method or event must return CancelOperation.

For more information, see "How Siebel CRM Handles a Predefined Business Service Method" on
page 293.

Example of Using Siebel VB to Create a Validation

The following Siebel VB example creates a validation that queries a specific field to determine if the
object interface event completed successfully or completed with a run-time error:

```
Function BusComp_PreSetFieldValue (FieldName As String,
               FieldValue As String) As Integer
' code to check if a quote discount > 20%
'       if it is, notify user and cancel the operation
Dim value as Integer
Dim msgtext as String
   If FieldName = "Discount" then
       value = Val(FieldValue)
       If value > 20 then
           msgtext = "Discounts greater than 20% must be approved"
          TheApplication.RaiseErrorText msgtext ' cancels the run
       Else
           BusComp_PreSetFieldValue = ContinueOperation
        End if
End If
End Function
```

Note the If statement in the following pseudocode:

```
If condition is true
   call custom code
   raise error text to cancel operation
Else
   returnValue = ContinueOperation
End If
```

In this If statement, Siebel CRM runs the custom code only if the condition is true:

■ If the condition is true, then Siebel CRM uses the custom code instead of the predefined code.

■ If the condition is not true, then the event handler returns ContinueOperation, and Siebel CRM
uses the predefined code.

You can also use the following alternative If statement:

```
returnValue = ContinueOperation
If condition is true
   call custom code
End If
```

Note that with a PreInvokeMethod event, you use the method name to determine if the script
conditionally runs. For example, consider the following code in Siebel eScript:

```
if (methodName == "PushOpportunity")
```

Example of Using Siebel eScript to Create a Validation

The following Siebel eScript example creates a validation that queries a specific field to determine if
the object interface event completed successfully or completed with a run-time error:

```
function BusComp_PreSetFieldValue (FieldName, FieldValue)
{
    var iReturn = ContinueOperation;
    //code to check if a quote discount > 20%
    //if it is, notify user and cancel the operation
    var varvalue;
    var msgtext;
    if (FieldName == "Discount")
    {
        varvalue = ToNumber(FieldValue);
        if (varvalue > 20)
        {
            msgtext = "Discounts greater than 20% must be approved";
            TheApplication().RaiseErrorText(msgtext); // cancels the run
        }
        else
        {
            iReturn = ContinueOperation;
        }
    }
}
```

Using Tracing to Determine When an Event Occurs

Many different events can occur if a view becomes current or if a script calls an object, so a simple
way to determine when various events occurs does not exist. It is recommended that you use tracing
to determine when events occur.

To use tracing to determine when an event occurs

1 To determine the exact order of events, use the Application_Start event to enable tracing when
the Siebel application starts.

In Siebel VB, use the following code:

```
TheApplication.TraceOn "filename, type, selection"
TheApplication.Trace "Event_Name has fired."
```

In Siebel eScript, use the following code:

```
TheApplication().TraceOn("filename, type, selection");
TheApplication().TraceOn(" Event_Name has fired.");
```

2 Add the following code in each event handler for the object:

> TheApplication.Trace "*Event_Name* fired."

Make sure you add this code to each of the following items:

- ■ Each relevant event, such as insert, delete, write, business component, and so forth

- ■ Each relevant preevent handler

3 Perform a few simple inserts, updates, and deletes.

4 Make a note of each message as Siebel CRM displays it.

Your notes will list the order that Siebel CRM uses to start events on the view or for the object.

Configuring Error Handling

This topic describes how to configure error handling.

COM Error Handling

The errCode parameter is the last parameter for every COM Data Server interface method. It is not available in the following object interfaces:

- ■ COM Data Control

- ■ Mobile Web Client Automation Server

- ■ Web Client Automation Server

- ■ Siebel Java Data Bean

Examples of Configuring Error Handling

This topic includes examples of configuring error handling.

Example of Configuring Error Handling for the COM Data Server

The following code is an example of error handling only for the COM Data Server:

```
GetBusObject (BusObjectName as string, errcode as integer) -> businessObject
```

Example of Configuring Error Handling for COM Data Control and Mobile Web Client Automation Server

The following code is an example of error handling for COM Data Control and Mobile Web Client Automation Server:

```
GetBusObject (BusObjectName as string) -> businessObject
```

Example of Configuring Error Handling for Siebel Java Data Bean

The SiebelException object handles errors in Siebel Java Data Bean. You can use the getErrorCode method and getErrorMessage method with the SiebelException object. The SiebelException object is defined in the com.siebel.data.SiebelException file. This file is a class file in one of the.jar files included in any java project that must communicate with Siebel CRM. For example:

```
...

import com.siebel.data.SiebelException;
import com.siebel.data.SiebelDataBean;
...
SiebelDataBean mySiebelBean=null;
try

{
    mySiebelBean = new SiebelDataBean();
    mySiebelBean.login("Siebel://SOMSERVER/somsiebel/AppObjMgr/", "CCONWAY",
"CCONWAY","enu");
}
catch (SiebelException e){
    // Exception handling code
    System.out.println (e.getErrorMessage ());
mySiebelBean = null; //avoid using mySiebelBean if login is unsuccessful
}


...
```

The ellipsis (…) in this code indicates code that was removed from the example in this book for brevity.

For more object interface methods on the SiebelException object, see the Siebel Java Data Bean JavaDoc that Oracle Universal Installer installs when you install Siebel Tools. Note that Oracle Universal Installer installs the JavaDoc only if you install the Siebel Java Integration option. It installs a zipped file that contains the JavaDoc in the *Tools_ROOT*\CLASSES folder.

Error Message Tracking

For error message tracking in ActiveX, you can use exceptions or object interface methods. This topic describes the methods that you can use.

EnableExceptions Method

The EnableExceptions method allows Siebel CRM to use native COM error handling. If the method is about to fail due to error, then Siebel CRM creates a COM exception and does not return the method. The COM host receives the control instead. Siebel CRM might display the error message, which is the default behavior for Microsoft Internet Explorer or Siebel VB. You cannot use script to modify this behavior.

The following code is an example of using the EnableExceptions method:

```
EnableExceptions(enable as integer)
```

GetLastErrCode Method and GetLastErrText Method

After Siebel CRM runs an object interface method, you can do the following:

■ To determine if Siebel CRM returned an error from the previous operation, you can call the GetLastErrCode method.

■ To return the text of the error message, you can call the GetLastErrText method.

For example:

```
GetLastErrCode() ' returns errCode As Integer

GetLastErrText() ' returns text As String
```

4 Using Siebel Visual Basic and Siebel eScript

This chapter describes how to use Siebel Visual Basic and Siebel eScript. It includes the following topics:

Overview of Using Siebel Visual Basic and Siebel eScript

You can use Siebel VB or Siebel eScript to customize and configure Siebel CRM beyond the capabilities that defining object properties provides. These languages integrate with other Siebel tools, such as the Applet Designer, Siebel CTI, and Siebel SmartScript. To define object properties, you can use the Applet Designer or attach scripts.

It is recommended that you use coding only after you determine that you cannot use any other tool. Siebel Tools provides many ways to configure Siebel CRM without coding. The following reasons explain why you must use Siebel Tools before you write your own code:

- Using Siebel Tools is easier than writing code.

- Your code might not work with an upgrade. Siebel CRM automatically updates a customization that you create in Siebel Tools during an upgrade. It does not update custom code you create. It might be necessary for you to manually update the code.

- Configuration through Siebel Tools results in better performance than using the same features through code. For more information, see *Siebel Performance Tuning Guide*.

Examples of Using Siebel Visual Basic and Siebel eScript

Siebel Visual Basic and Siebel eScript allow you to customize Siebel CRM behavior.

Validating Data

To meet the validation requirements for your business, you can use Siebel Visual Basic or Siebel eScript to create a custom code that uses validation rules before Siebel CRM records or deletes a record. You can use data validation to access the following types of data:

■ **Internal data.** For example, you can write custom code that configures Siebel CRM to verify that the revenue amount for an opportunity is greater than zero if the probability of the opportunity is greater than 20 percent.

■ **External data.** For example, to verify the availability of a conference room before Siebel CRM inserts a new activity, you can write custom code that reads data from the database table of an external application.

Modifying and Controlling Data

Siebel Visual Basic and Siebel eScript allow you to modify and control data, such as update, insert, or delete a record. For example, you can control the value of one field according to the value of another field:

■ Set the probability of the opportunity, such as 98%, according to the sales stage of the opportunity, such as 03 - Closing.

■ If the sales cycle is at or past the Quote Submitted stage, then do not allow the user to modify the Revenue field.

You can use an object interface method to manipulate data to notify a Siebel programming language of an error and provide it information. This capability allows you to configure the Siebel application to handle the error and take appropriate action.

Manipulating data in a Siebel programming language conforms to the same visibility rules that a predefined Siebel application uses. For example, assume the visibility rules that exist in a predefined Siebel application result in a business object that Siebel CRM can read but not edit. In this situation, a configuration that you create through a Siebel programming language can also read but not edit this same object. You cannot use a Siebel programming language to circumvent the visibility rules or the security constraints that a predefined Siebel application enforces.

Customizing Behavior for User Interface Elements

To add a user interface element to an applet, you can use the Applet Layout Editor in Siebel Tools. To associate a behavior with this element, you can use a Siebel programming language. For example, you can add a button on an applet that opens another application, such as Microsoft Excel.

Guidelines for Using Siebel VB and Siebel eScript

This topic describes guidelines for using Siebel VB and Siebel eScript. It includes the following topics:

■ "Declare Your Variables" on page 67

■ "Use a Standardized Naming Convention" on page 67

■ "Use Constants to Standardize Code" on page 68

For introductory information about Siebel VB, see *Siebel VB Language Reference*.

Declare Your Variables

To help other developers understand your code and to help you debug your code, it is recommended that you declare your variables.

Declaring Your Variables in Siebel VB

You can use the Dim statement in the Option Explicit statement to declare a variable before you use it. To reduce the amount of memory that your code uses and to improve processing speed, it is recommended that you avoid using a Variant variable. You can declare a variable without specifying a data type. If you do not specify a data type, then Siebel VB assumes the Variant type. This type requires 16 bytes and uses twice as much memory as the next smallest data type.

Use a Standardized Naming Convention

To improve efficiency and reduce errors, it is recommended that all developers in your programming group use the same standardized naming convention. The convention that you use does not matter. Table 7 describes a common convention that prefixes each variable with a letter that indicates the type. If necessary, you can also use a suffix.

Table 7.   Naming Conventions for Variables in Scripts

Data Type	Naming Convention	Example
String	s	sName
Integer	i	iReturn
Long integer	l	lBigCount
Single-precision number	si	siAllowance
Double-precision number	d	dBudget

Table 7.   Naming Conventions for Variables in Scripts

Data Type	Naming Convention	Example
Object	o	oBusComp
Currency	c	cAmtOwed

Use Constants to Standardize Code

Siebel Visual Basic and Siebel eScript provide constants that you can use to make your code more readable by other developers. A constant clarifies the intent of the operation. Use the constant name in your code. Do not use the integer value in your code. The integer value is included only to aid in debugging. If you store the constant in a local variable, and if the value of the local variable is available, then Siebel CRM displays the integer value in the Debugger.

Table 8 lists the Siebel constants you can use.

It is recommended that you use the constant and that you do not use the integer value because integer values are subject to modification.

Table 8.   Siebel Constants

Used With	Constant Name	Integer Value
Pre Event Handler Methods	ContinueOperation	1
	CancelOperation	2
Search Methods	ForwardBackward	256
	ForwardOnly	257
NewRecord Method	NewBefore	0
	NewAfter	1
	NewBeforeCopy (Not available with Siebel Java Data Bean)	2
	NewAfterCopy (Not available with Siebel Java Data Bean)	3
Siebel ViewMode Methods. For more information, see "Constants You Can Use with the SetViewMode Method" on page 247.	SalesRepView	0
	ManagerView	1
	PersonalView	2
	AllView	3
	OrganizationView	5
	GroupView	7
	CatalogView	8
	SubOrganizationView	9

Avoid Nested If Statements

To avoid a nested If statement, you can use one of the following statements:

■ In Siebel VB, use the Select Case statement

■ In Siebel eScript, use the Switch statement

Each of these statements chooses from multiple alternatives according to the value of a single variable. It is recommended that you use the Select Case statement instead of a series of nested If statements. It simplifies code maintenance and improves performance. Siebel CRM evaluates the variable only once.

The following is an example use of the Switch statement:

```
switch (FieldName)
{
    case "Status":
    {
        var sysdate = new Date();
        var sysdatestring = ((sysdate.getMonth() + 1) + "/" + sysdate.getDate() +
            "/" + sysdate.getFullYear()+ " "+ sysdate.getHours() + ":" +
            sysdate.getMinutes()+":" + sysdate.getSeconds());
        this.SetFieldValue("Sales Stage Date",sysdatestring);
        if ((FieldValue) == "Not Attempted")
        {
            if (this.GetFieldValue("Primary Revenue Amount") > 0)
            this.SetFieldValue("Primary Revenue Amount",0);
        }
        break;
    }
    case "Revenue":
    {
        if (newrecSw =="Y")
        {
            newrecSw = "";
            this.SetFieldValue("Account Revenue",(FieldValue));
        }
        break;
    }
}
```

Applying Multiple Object Interface Methods to a Single Object

To apply multiple object interface methods to a single object, you can use the With statement in Siebel VB or Siebel eScript. It reduces typing and makes the code easier to read.

Example of Using the With Statement in Siebel VB

The following example uses the With statement in Siebel VB:

```
Set oBusObject = TheApplication.GetBusObject("Opportunity")
Set oBusComp = oBusObject.GetBusComp("Opportunity")
With oBusComp
    .ActivateField "Account"
```

```
    .ClearToQuery
    .SetSearchSpec "Name", varname
    .ExecuteQuery ForwardBackward
    If (.FirstRecord = 1) Then
        sAccount = .GetFieldValue "Account"
    End If
End With
. . .

Set oBusComp = Nothing
Set oBusObject = Nothing
```

The following example is not recommended. It does not use the With statement:

```
Set oBusObject = TheApplication.GetBusObject("Opportunity")
Set oBusComp = oBusObject.GetBusComp("Opportunity")
oBusComp.ActivateField "Account"
oBusComp.ClearToQuery
oBusComp.SetSearchSpec "Name", varname
oBusComp.ExecuteQuery ForwardBackward
If (oBusComp.FirstRecord = 1) Then
    sAccount = oBusComp.GetFieldValue "Account"
End If
. . .
```

Example of Using the With Statement in Siebel eScript

The following example uses the With statement in Siebel eScript:

```
var oBusObject = TheApplication().GetBusObject("Opportunity");
var oBusComp = oBusObject.GetBusComp("Opportunity");
with (oBusComp)
{
    ActivateField("Account");
    ClearToQuery();
    SetSearchSpec("Name", varname);
    ExecuteQuery(ForwardBackward);
    if (FirstRecord())
    {
        var sAccount = GetFieldValue( "Account");
    }
} //end with
```

The following example is not recommended. It does not use the With statement:

```
var oBusObject = TheApplication().GetBusObject("Opportunity");
var oBusComp = oBusObject.GetBusComp("Opportunity");
oBusComp.ActivateField("Account");
oBusComp.ClearToQuery();
oBusComp.SetSearchSpec("Name", varname);
oBusComp.ExecuteQuery(ForwardBackward);
if oBusComp.FirstRecord();
{
```

```
        var sAccount = oBusComp.GetFieldValue("Account");
    }
. . .
```

Use a Self-Reference to Indicate the Current Object

To indicate the current object, you can use the following statements:

■ In Siebel VB, use the Me statement.

■ In Siebel eScript, use the This keyword.

You can use the statement or keyword instead of referencing an active business object.

Example of Using the Me Statement

The following business component event handler uses the Me statement instead of the ActiveBusComp statement:

```
Function BusComp_PreSetFieldValue(FieldName As String, FieldValue As String) As
Integer

If Val(Me.GetFieldValue("Rep %")) >75 Then
    TheApplication.RaiseErrorText("You cannot set the Rep% to greater than 75")
End If
BusComp_PreSetFieldValue = ContinueOperation

End Function
```

For examples of using the Me statement, see the following topics:

■ "ParentBusComp Method for a Business Component" on page 223

■ "SetViewMode Method for a Business Component" on page 246

■ "BusComp_PreQuery Event" on page 271

■ "BusComp_PreWriteRecord Event" on page 273

■ "ActiveMode Method for an Applet" on page 103

Example of Using the This Keyword

The following business component event handler uses the This keyword instead of the ActiveBusComp statement:

```
if (condition)
{ ...
    this.SetSearchSpec(...);
    this.ExecuteQuery();
    return (CancelOperation);
}
else
     return(ContinueOperation);
```

Delete Objects You Have Created That You No Longer Require

Although the interpreter performs object cleanup, it is recommend that you write code that explicitly deletes objects it created that you no longer require. Your code must delete each Siebel object in the same procedure it used to create it.

To delete objects, do the following:

■ In Siebel VB, set each object to Nothing.

■ In Siebel eScript, set each object to Null.

You can delete these objects in the reverse order that the code created them. Make sure you code deletes child objects before it deletes parent objects.

Example of Deleting Objects in Siebel VB

The following code is an example of deleting objects in Siebel VB:

```
Set oBusObj  = TheApplication.GetBusObject("Contact")
Set oBusComp= oBusObj.GetBusComp("Contact")

Your code here

Set oBusComp = Nothing
Set oBusObj  = Nothing
```

Example of Deleting Objects in Siebel eScript

The following code is an example of deleting objects in Siebel eScript:

```
var oBusObject = TheApplication().GetBusObject("Contact"");
var oBusComp = oBusObject.GetBusComp("Contact");

Your code here

oBusComp = null;
oBusObject = null;
```

Make Sure Function Names Are Unique

Make sure that the name is unique for every function you create. If two functions use the same name, and if those functions are in the same view, then results are unpredictable. Consider using a naming convention, such as using the view name as a function name prefix.

Manage the Script Buffer

The size limit of a non-Unicode script buffer is 65530 bytes. The amount of available memory limits the Unicode script buffer. Make sure your computer possesses enough memory to accommodate this buffer.

Using Siebel VB and Siebel eScript Formats

There are some important differences between the formats that Siebel VB and Siebel eScript use:

■ Siebel eScript is case-sensitive. For example, theApplication is different from TheApplication. Siebel VB is not case-sensitive.

■ Siebel eScript does not distinguish between a subroutine and a function. A subroutine cannot accept an argument. A function can accept an argument. In Siebel eScript, because every object interface method is a function, you must follow it with a pair of parentheses. You must use this technique if the function does or does not accept an argument.

In many instances, the only difference between the Siebel VB format and the Siebel eScript format is that the Siebel eScript format requires a pair of parentheses at the end. In these instances, this book only includes the Siebel VB format. To determine the Siebel eScript format, add the parentheses.

Differences Between Siebel eScript and ECMAscript

ECMAscript is a programming language that developers use to script a client on the Web. JavaScript is a type of ECMAscript. Siebel eScript does not include user interface functions. You cannot use it to animate or control a Web page. It includes the following functions that are not part of ECMAscript:

■ SELib

■ Clib

You can use these functions to interact with the operating and file systems, and for performing input and output file operations. These objects include functions that are similar to functions that the C programming language uses. For more information, see *Siebel eScript Language Reference*.

ECMAscript does not require you to declare a variable. It declares a variable implicitly as soon as you use it.

Handling the Date Format in Siebel VB

If you use an object interface method that includes a date, then use caution regarding the date format. The GetFieldValue method returns the date in the following format:

dd/mm/yyyy

The CVDate function expects the regional setting. If you apply it, then Siebel CRM might return an error. The GetFormattedFieldValue method uses the regional settings of the operating system that is installed on the computer that runs the Siebel client. The regional setting might specify the year with two digits, and can cause an error with the year 2000 problem. For these reasons, use the following procedure for performing date arithmetic.

To handle the date format in Siebel VB

1 To return the value of the date fields, use the GetFieldValue object interface method.

For more information, see "GetFieldValue Method for a Business Component" on page 205.

2 Use the DateSerial function convert the value of the date field to a date variable.

3 Perform the required date arithmetic.

For example, you can use the following Siebel VB code:

```
Dim strDate as String, varDate as Variant
strDate = oBC.GetFieldValue("Date Field")
varDate =DateSerial(Val(Mid(strDate,7,4)),Val(Left(strDate,2)),_
    Val(Mid(strDate,4,2)))
any date arithmetic
```

Returning Run-Time Errors in Siebel VB

This topic describes how to return run-time errors in Siebel VB.

To return run-time errors in Siebel VB

■ Return a run-time error code with one of the following items:

■ **Predefined Siebel VB properties.** You can use some combination of Err, ErrText, and Error.

■ **Custom Siebel VB method.** If you access a Siebel object interface through Component Object Model (COM) or ActiveX, then use the following code to view the text of the error message:

```
If errCode <> 0 Then
    ErrText = GetLastErrText
    TheApplication.RaiseErrorText ErrText
    Exit Sub
End If
```

The GetLastErrText method is only available if you use an interface that is external to Siebel Tools. You can use it in Microsoft VB but not in Siebel VB.

Object interface methods use numeric error codes in a range of 4000 to 4999.

For more information about error-handling and error codes, see *Siebel VB Language Reference*.

Opening the Siebel Script Editor

This topic describes how to open the Siebel Script Editor.

To open the Siebel Script Editor

1 In Siebel Tools, in the Object Explorer, click the object type you must modify.

For example, click Applet.

2 In the Object List Editor, locate and then right-click the object you must modify.

For example, in the Applets list, locate and then right-click Contact List Applet.

3 In the Scripting Language dialog box, choose one of the following menu items:

- ■ Edit Server Scripts
- ■ Edit Browser Scripts

4 In the Scripting Language dialog box, choose Visual Basic or eScript, and then click OK.

Declaring a Variable

This topic describes how to declare a variable.

Declaring a Local Variable

This topic describes how to declare a local variable. You can access the value of a local variable only in the script where you define the local variable.

To declare a local variable

1 Open the Siebel Script Editor.

For more information, see .

2 In the navigation tree of the script editing window, expand the object tree, and then click the script you must modify.

For example, expand the WebApplet tree, and then click WebApplet_PreInvokeMethod.

3 In the script editing window, use one of the following statements in your custom script:

- ■ In Siebel VB, use the Dim statement.
- ■ In Siebel eScript, use the Var statement.

Example of Declaring a Local Variable in Siebel VB
The following example declares a local variable in Siebel VB:

```
Sub WebApplet_Load
    Dim localStr As String
End Sub
```

Example of Declaring a Local Variable in Siebel eScript
The following example declares a local variable in Siebel eScript:

```
function WebApplet_Load ()
{
    var localStr;
}
```

Declaring a Module Variable

This topic describes how to declare a module variable. In this situation, a *module* is a group of methods contained in an object that you can script. For example, a business service, business component, application object, and so forth. You can access the value of a module variable in the script where you define the module variable and in other scripts in the object or module where you define the module variable. To access a module variable, an instance of the object where you define the variable must exist.

To declare a module variable

1   Open the Siebel Script Editor.

For more information, see "Opening the Siebel Script Editor" on page 74.

2   In the navigation tree of the script editing window, expand the general tree, and then click declarations.

3   In the script editing window, use one of the following statements in your custom script:

■   In Siebel VB, use the Dim statement.

■   In Siebel eScript, use the Var statement.

The following example declares a module variable in Siebel VB:

```
(general)
(declarations)
Dim ContactId as String
```

Declaring a Global Variable

This topic describes how to declare a global variable.

To declare a global variable

1   Open the Siebel Script Editor for the object you must modify.

For more information, see "Opening the Siebel Script Editor" on page 74.

2   Use the Global statement to declare the variable.

The following example includes the Global statement in Siebel eScript:

```
TheApplication().gVar = "some value";
```

3   Repeat Step 1 and Step 2 for each object that must access the value of the global variable.

Do Not Use a Global Variable to Reference a Siebel Object

Do not use a global variable to reference a Siebel object, such as a business component or business object. If you must reference a Siebel object, then set the global variable to Nothing when you no longer require the object, or in the Application_Close event.

If you do not set the variable to Nothing, then a memory problem might occur. Siebel CRM cannot release from memory the object that the global variable references until the variable no longer references the object. If you must create a global variable for a business component, then make sure a global variable for the business object exists.

For more information, see .

Calling More Than One Object Interface Method In a Script

You can call more than one object interface method in a script.

To call more than one object interface method in a script

- Use one of the following statements:

  - Select statement in Siebel VB

  - Switch statement in Siebel eScript.

Example of Calling More Than One Object Interface Method in Siebel VB

The following example uses the Select statement in Siebel VB:

```
Dim iReturn As Integer
iReturn = ContinueOperation
Select Case methodName
    Case "PushOpportunity"
        your custom code
        iReturn = CancelOperation
    Case "Stage3"
        your custom code
        iReturn = CancelOperation
End Select
object_PreInvokeMethod = iReturn
```

Example of Calling More Than One Object Interface Method in Siebel eScript

The following example is in Siebel eScript:

```
var iReturn;
switch (methodName)
{
    case "PushOpportunity":
        //your custom code
        iReturn = CancelOperation;
        break;
    case "Stage3":
        //your custom code
```

```
            iReturn = CancelOperation;
            break;

        default:
            iReturn = ContinueOperation;
    }
    return (iReturn);
```

Using Script to Add Business Logic to a Business Component

You can use Server Script or Browser Script to add business logic to a business component.

To use script to add business logic to a business component

1   Open the Siebel Script Editor.

For more information, see "Examples of Using Siebel Visual Basic and Siebel eScript" on page 65.

2   In the navigation tree of the Siebel Script Editor, choose an event in the BusComp Tree.

3   In the Siebel Script Editor window, write your script.

4   Choose the Debug menu, and then the Check Syntax menu item.

The Check Syntax menu item is available only for Server Script.

5   Save the modifications.

6   Choose the Tools menu, and then the Compile Selected Objects menu item.

7   Choose the Debug menu, and then the Start menu item.

Using a MiniButton Control to Call a Custom Method

This topic describes how to use a minibutton control to call a custom method.

To use a minibutton control to call a custom method

1   Open Siebel Tools.

2   Expose the Applet User Prop object type:

    a   Choose the View menu, and then the Options menu item.

    b   In the Development Tools Options dialog box, click the Object Explorer tab.

    c   Expand the Applet tree, and then make sure the Applet User Prop object type contains a check mark.

    d   Click Ok.

3   In the Object Explorer, click Applet.

4   In the Applets list, locate the applet you must modify.

5   In the Object Explorer, expand the Applet tree, and then click Control.

6   In the Controls list, add a new control using values from the following table.

Property	Value
Name	ButtonTest
Caption	Test
HTML Type	MiniButton
Method Invoked	MyTest

7   In the Applets list, right-click the applet and then choose the Edit Web Layout menu item.

8   In the Controls/Columns window, modify the template mode to Edit List.

9   Drag and then drop the ButtonTest control from the Controls/Columns window to an appropriate location on the canvas of the Web Layout Editor.

10   Choose the File menu, and then the Save menu item.

11   Close the Web Layout Editor.

12   Enable the button:

a   In the Object Explorer, click Applet User Prop.

b   In the Applet User Props list, create a new user property using values from the following table.

Property	Value
Name	CanInvokeMethod: MyTest
	For more information about the CanInvokeMethod applet user property, see *Siebel Developer's Reference*.
Value	TRUE

As an alternative, you can use script to enable the button. For more information, see "Using Script to Enable a MiniButton" on page 80.

13   In the Applets list, right-click the applet, and then choose Edit Browser Scripts.

14   In the BrowserApplet window, add the following script:

```
function Applet_PreInvokeMethod (name, inputPropSet)
{

switch (name) {

  case "MyTest":

    theApplication().SWEAlert("Browser Script!");
```

```
          return("CancelOperation");

      break;

      }

  return("ContinueOperation");

  }
```

15 Close the BrowserApplet window.

16 In the Applets list, right-click the applet, and then choose Compile Selected Objects.

17 In the Object Compiler window, click Compile.

18 Start the Siebel client, and then navigate to the Accounts screen.

19 Click Test.

This is the button you created in Step 6.

20 Make sure the Siebel client displays an alert box that includes the following message:

Browser Script!

Using Script to Enable a MiniButton

To enable a minibutton, it is strongly recommended that you use the declarative technique described in Step 6 on page 79. In most situations, declarative programming does not negatively impact performance as much as scripting does. However, in certain situations, you can use a script to enable a button and improve performance. For example, you can use script to avoid a complicated Value expression that is longer than 255 characters that requires multiple calculated fields and declarative programming.

To use script to enable a minibutton

1 Complete Step 1 on page 78 through Step 11 on page 79.

2 In the Applets list, right-click the applet you must modify, and then choose Edit Server Scripts.

3 In the Scripting Language dialog box, choose Visual Basic or eScript, and then click OK.

4 In the Script Editor, expand the WebApplet tree, and then click the WebApplet_PreCanInvokeMethod function.

5 In the Script Editor, add the following script:

```
    function WebApplet_PreCanInvokeMethod (MethodName, &CanInvoke)
    {

    if (MethodName == "MyTest")

    {

      CanInvoke = "TRUE";

      return(CancelOperation);
```

```
        }

        return(ContinueOperation);

    }
```

6 Continue with Step 13 on page 79.

Tracing a Script

As part of debugging a script you can run a trace on allocations, events, and SQL commands. You can start tracing for a user account, such as your development team. The Siebel Server sends trace information to a log file.

For information about:

■ Configuring server components, see *Siebel Applications Administration Guide*

■ Logging events, see *Siebel System Monitoring and Diagnostics Guide*

■ File tracing, see "Trace Method for an Application" on page 173

To enable logging for the local object manager, you can set the SIEBEL_LOG_EVENT environment variable to a value of 2 through 5. For more information, see *Siebel Applications Administration Guide*.

To trace a script

1 In the Siebel client, navigate to the Administration - Server Configuration screen, and then the Servers view.

2 In the Components list, choose a component to log.

3 In the Events list, locate the Object Manager Extension Language Log event.

If this record does not exist, then you cannot use the component you chose in Step 2 for logging.

4 Set the Log Level to 1.

5 (Optional) Modify tracing parameters:

   a Click the Parameters tab.

   b In the Component Parameters list, click Menu, and then choose the Columns Displayed menu item.

   c Move the Parameter Alias and Subsystem columns to the Selected Columns window, and then click Save.

   d In the Component Parameters list, click Query.

e Enter the following values, and then click Go.

Field	Value
Parameter Alias	Trace*
Subsystem	Object Manager

f Set one or more tracing parameters using values from the following table.

Information to Trace	Parameter Alias	Settings for Current Value and Value on Restart
Allocations	TraceAlloc	Enter 1 to enable logging. Enter 0 to disable logging.
Events	TraceEvents	Enter 1 to enable logging. Enter 0 to disable logging.
SQL Commands	TraceSql	Enter 1 to enable logging. Enter 0 to disable logging.
Users	TraceUser	Enter a list of user names. Use a comma to separate each user name. For example: sadmin,mmasters. Do not use spaces. You cannot enter more than 20 characters in this parameter. **CAUTION:** Tracing on the Siebel Server can affect performance. If you simultaneously trace multiple users, then use caution.

To instruct Siebel CRM to immediately modify these parameters, enter values in the Current Value column.

To instruct Siebel CRM to modify these parameters only after a restart, enter values in the Value on Restart column.

6 Test your work, and then examine the results.

7 When you are finished logging, set the Log Level that you set in Step 4 to 0.

The following is part of an example of the trace output:

```
2021 2003-04-09 15:37:20 2003-04-09 16:40:52 -0700 00000022 001 001f 0001 09
SCCObjMgr_enu 47126 1680 1584 C:\sea752\siebsrvr\log\SCCObjMgr_enu_47126.log 7.5.3
[16122] ENU

ObjMgrSessionInfoObjMgrLogin32003-04-09 15:37:20Login name : SADMIN

ObjMgrSessionInfoObjMgrAuth32003-04-09 15:37:20Authentication name : SADMIN

ObjMgrSessionInfoObjMgrLogin32003-04-09 15:37:20Session Type: Regular Session

GenericLogGenericError12003-04-09 15:37:20Invocation of Applet Menu New
Service::NewExpense is not allowed.

GenericLogGenericError12003-04-09 15:37:20Invocation of Applet Menu New
Service::NewTimeSheet is not allowed.
```

```
ObjMgrExtLangLogObjMgrExtLangLog02003-04-09 15:38:27[User: SADMIN] EVENT, BEGIN,
BusComp [Account], BusComp_Query.
```

```
ObjMgrExtLangLogObjMgrExtLangLog02003-04-09 15:38:27[User: SADMIN] EVENT, END,
BusComp [Account], BusComp_Query.
```

```
ObjMgrExtLangLogObjMgrExtLangLog02003-04-09 15:38:58[User: SADMIN] EVENT, BEGIN,
BusComp [Account], BusComp_NewRecord.
```

```
ObjMgrExtLangLogObjMgrExtLangLog02003-04-09 15:38:58[User: SADMIN] EVENT, END,
BusComp [Account], BusComp_NewRecord.
```

```
ObjMgrExtLangLogObjMgrExtLangLog02003-04-09 15:39:08[User: SADMIN] EVENT, BEGIN,
BusComp [Account], BusComp_PreSetFieldValue.
```

```
ObjMgrExtLangLogObjMgrExtLangLog02003-04-09 15:39:08[User: SADMIN] EVENT, END,
BusComp [Account], BusComp_PreSetFieldValue.
```

```
ObjMgrSessionInfoObjMgrLogin32003-04-09 16:40:52Username: SADMIN, Login Status:
Attempt,  Session Id: !1.690.b816.3e94a0a0, IP Address: 172.20.94.66
```

5 Siebel Object Interfaces Reference

This chapter describes object interface methods and events. It includes the following topics:

- Format of the Object Interface Method on page 85
- Technologies You Can Use to Access Object Interface Methods and Events on page 87
- Object Interfaces Reference on page 102

Format of the Object Interface Method

This topic describes formats for object interface methods, arguments, and return values. A Siebel object interface method uses the following format:

ObjectType.MethodName(*arg1*[, *arg2, …, argn*])

where:

- Italicized text indicates a variable.

- Square brackets [ ] indicate an optional argument. The description of the argument indicates the default value for each optional argument.

- *ObjectType* is the object type. For example, BusComp indicates the business component that Siebel CRM defines for the object interface method.

- *MethodName* is the name of the object interface method that you call. A method can be a subroutine that does not return a value, such as SetViewMode, or a method that returns a value, such as GetFieldValue.

- *arg1*, *arg2*, or *argn* is a string, constant, integer, or object. Use parenthesis in the following ways:

  - In Siebel VB, if an object interface method returns a value, then enclose these arguments in parentheses.

  - in Siebel VB, if an object interface method does not return a value, then do not enclose these arguments in parentheses.

  - In Siebel eScript, always enclose these arguments in parentheses.

  If you use parentheses ( ) when none are required, or if you fail to use them if they are required, then Siebel CRM creates a Type Mismatch error that includes error code 13. Siebel CRM also creates this error if you use an incorrect number of arguments.

  If you use the COM Data Server interface, then you must include the errCode argument as the last argument.

Note how this book uses the following terms:

■ *ObjectReference* is an ObjectType variable name that identifies the object that calls the object interface method. If you call a method on an object in the event handler of that object, then you are not required to explicitly specify the ObjectReference.

■ *returnValue* is the value that the object interface method returns. Some methods, such as GetBusComp, return a business component object. Some methods return a string or integer. Some methods do not return any value.

Formats for Siebel VB

If there is a return value, then use the following format:

    returnValue = *ObjectReference.MethodName*(*arg1, arg2, ..., arg*n)

If there are no arguments, then use the following format:

    returnValue = *ObjectReference.MethodName*

If there is no return value, then use the following format:

    *ObjectReference.MethodName arg1, arg2, ..., arg*n

The following examples use Siebel VB:

    acctName = acctBC.GetFieldValue("Name")

    acctBC.SetViewMode AllView

Formats for Siebel eScript

If there is a return value, then use the following format:

    returnValue = *ObjectReference.MethodName*(*arg1, arg2, ..., arg*n);

If there are no arguments, then use the following format:

    returnValue = *ObjectReference.MethodName*();

If there is no return value, then use the following format:

    ObjectReference.MethodName(*arg1, arg2, ..., arg*n);

The following examples use Siebel eScript:

    acctName = acctBC.GetFieldValue("Name");

    acctBC.SetViewMode(AllView);

Formats for the Component Object Model

The format that Siebel CRM uses for the Component Object Model (COM) depends on the language you use to call the COM interface. For Microsoft Visual Basic and equivalent languages, the format is similar to the format you use for Siebel VB, except that if you use COM Data Control, then Siebel CRM passes an error code as the final argument.

Technologies You Can Use to Access Object Interface Methods and Events

This topic describes technologies you can use to access object interface methods and events. It includes the following topics:

- "Technologies You Can Use to Access Object Interface Methods" on page 87
- "Technologies You Can Use to Access Object Interface Events" on page 100

Technologies You Can Use to Access Object Interface Methods

This topic lists the technologies you can use to access object interface methods. It includes the following topics:

- "Applet Methods" on page 87
- "Application Methods" on page 88
- "Business Component Methods" on page 92
- "Business Object Methods" on page 95
- "Business Service Methods" on page 96
- "Control Methods" on page 97
- "Property Set Methods" on page 97
- "Miscellaneous Methods" on page 99

The term Yes indicates an object interface that you can use with an application method.

Applet Methods

Table 9 lists the technologies you can use to access applet object interface methods. You can use an applet object interface method only with Server Script and Browser Script.

Table 9.   Applet Methods

Method	Server Script	Browser Script
"ActiveMode Method for an Applet" on page 103	No	Yes
"BusComp Method for an Applet" on page 104	Yes	Yes
"BusObject Method for an Applet" on page 104	Yes	Yes
"FindActiveXControl Method for an Applet" on page 105	No	Yes
"FindControl Method for an Applet" on page 106	No	Yes

Table 9.   Applet Methods

Method	Server Script	Browser Script
"Examples of Using the FindControl Method" on page 106	Yes	Yes
"Name Method for an Applet" on page 108	Yes	Yes

Application Methods

Table 10 lists the technologies you can use to access application methods.

Table 10.   Application Methods

Method	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
ActiveApplet Method for an Application	No	Yes	No	No	No	No	No
ActiveBusComp Method for an Application	No	Yes	No	No	No	No	No
ActiveBusObject Method for an Application	Yes	Yes	No	Yes	No	No	No
ActiveViewName Method for an Application	Yes	Yes	No	Yes	No	No	No
Attach Method for an Application	No	No	No	No	Yes	No	Yes
CurrencyCode Method for an Application	Yes	Yes	No	Yes	Yes	Yes	Yes
Detach Method for an Application	No	No	No	No	Yes	No	Yes
EnableExceptions Method for an Application	No	No	No	Yes	Yes	No	No
FindApplet Method for an Application	No	Yes	No	No	No	No	No
GetBusObject Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes

Table 10.   Application Methods

Method	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
GetDataSource Method for an Application Called only with InvokeMethod	Yes	No	No	Yes	Yes	No	Yes
GetLastErrCode Method for an Application	No	No	Yes	Yes	Yes	No	No
GetLastErrText Method for an Application	No	No	Yes	Yes	Yes	Yes	No
GetProfileAttr Method for an Application	Yes	Yes	No	Yes	Yes	Yes	Yes
GetService Method for an Application	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GetSharedGlobal Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
GotoView Method for an Application	Yes	No	No	No	No	No	No
InvokeMethod Method for an Application	Yes	Yes	No	Yes	Yes	Yes	Yes
IsViewReadOnly Method for an Application Called only with InvokeMethod	Yes	Yes	No	Yes	Yes	Yes	Yes
Language Method for an Application Called only with InvokeMethod	Yes	No	No	No	No	No	No
LoadObjects Method for an Application	No	No	No	No	No	Yes	No

Table 10.   Application Methods

Method	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
LoadUserAttributes Method for an Application	Yes	No	No	No	No	No	No
Login Method for an Application	No	No	No	Yes	Yes	Yes	Yes
LoginId Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
LoginName Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
Logoff Method for an Application	No	No	No	No	Yes	No	Yes
LookupMessage Method for an Application	Yes	No	No	No	No	No	No
LookupValue Method for an Application Called only with InvokeMethod	Yes	No	No	Yes	Yes	No	Yes
Name Method for an Application	No	Yes	Yes	No	No	No	No
NewPropertySet Method for an Application	Yes	Yes	Yes	Yes	Yes	Yes	Yes
PositionId Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
PositionName Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
RaiseError Method for an Application	Yes	No	No	No	No	No	No
RaiseErrorText Method for an Application	Yes	No	No	No	No	No	No

Table 10. Application Methods

Method	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
SetPositionId Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
SetPositionName Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
SetProfileAttr Method for an Application	Yes	Yes	No	Yes	Yes	Yes	Yes
SetSharedGlobal Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
"ShowModalDialog Method for an Application"	No	Yes	No	No	No	No	No
SWEAlert Method for an Application	No	Yes	No	No	No	No	No
Trace Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
TraceOff Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes
TraceOn Method for an Application	Yes	No	No	Yes	Yes	Yes	Yes

Business Component Methods

Table 11 lists the technologies you can use to access business component methods. You cannot use these method with the Web Client Automation Server.

Table 11.   Business Component Methods

Method	Server Script	Browser Script	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
ActivateField Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
ActivateMultipleFields Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
Associate Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
BusObject Method for a Business Component	Yes	Yes	Yes	Yes	Yes	Yes
ClearLOVCache Method for a Business Component Called only with InvokeMethod	Yes	Yes	Yes	Yes	Yes	Yes
ClearToQuery Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
CreateFile Method for a Business Component Called only with InvokeMethod	Yes	No	Yes	Yes	Yes	Yes
DeactivateFields Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
DeleteRecord Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
ExecuteQuery Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
ExecuteQuery2 Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
FirstRecord Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
FirstSelected Method for a Business Component	Yes	No	No	No	No	No
GenerateProposal Method for a Business Component Called only with InvokeMethod	Yes	No	Yes	Yes	Yes	Yes

Table 11. Business Component Methods

Method	Server Script	Browser Script	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
GetAssocBusComp Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
GetFieldValue Method for a Business Component	Yes	Yes	Yes	Yes	Yes	Yes
GetFile Method for a Business Component Called only with InvokeMethod	Yes	No	Yes	Yes	Yes	Yes
GetFormattedFieldValue Method for a Business Component	Yes	Yes	Yes	Yes	Yes	Yes
GetLastErrCode Method for a Business Component	No	No	Yes	Yes	No	No
GetLastErrText Method for a Business Component	No	No	Yes	Yes	No	No
GetMultipleFieldValues Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
GetMVGBusComp Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
GetNamedSearch Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
GetPicklistBusComp Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
GetSearchExpr Method for a Business Component	Yes	Yes	Yes	Yes	Yes	Yes
GetSearchSpec Method for a Business Component	Yes	Yes	Yes	Yes	Yes	Yes
GetSortSpec Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
GetUserProperty Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
GetViewMode Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
InvokeMethod Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
LastRecord Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes

Table 11.   Business Component Methods

Method	Server Script	Browser Script	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
Name Method for a Business Component	Yes	Yes	Yes	Yes	Yes	Yes
NewRecord Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
NextRecord Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
NextSelected Method for a Business Component	Yes	No	No	No	No	No
ParentBusComp Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
Pick Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
PreviousRecord Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
PutFile Method for a Business Component Called only with InvokeMethod	Yes	No	Yes	Yes	Yes	Yes
RefineQuery Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
RefreshBusComp Method for a Business Component Called only with InvokeMethod	Yes	Yes	Yes	Yes	Yes	Yes
RefreshRecord Method for a Business Component Called only with InvokeMethod	Yes	Yes	Yes	Yes	No	Yes
Release Method for a Business Component	No	No	No	No	No	Yes
SetAdminMode Method for a Business Component Called only with InvokeMethod	Yes	No	Yes	Yes	Yes	Yes
SetFieldValue Method for a Business Component	Yes	Yes	Yes	Yes	Yes	Yes
SetFormattedFieldValue Method for a Business Component	Yes	Yes	Yes	Yes	Yes	Yes

Table 11.   Business Component Methods

Method	Server Script	Browser Script	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
SetMultipleFieldValues Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
SetNamedSearch Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
SetSearchExpr Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
SetSearchSpec Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
SetSortSpec Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
SetUserProperty Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
SetViewMode Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
UndoRecord Method for a Business Component	Yes	No	Yes	Yes	Yes	Yes
WriteRecord Method for a Business Component	Yes	Yes	Yes	Yes	Yes	Yes

Business Object Methods

Table 12 lists the technologies you can use to access business object methods. You cannot use these methods with the Web Client Automation Server.

Table 12.   Business Object Methods

Method	Server Script	Browser Script	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
GetBusComp Method for a Business Object	Yes	Yes	Yes	Yes	Yes	Yes
GetLastErrCode Method for a Business Object	No	No	Yes	Yes	No	No
GetLastErrText Method for a Business Object	No	No	Yes	Yes	No	No

Table 12.   Business Object Methods

Method	Server Script	Browser Script	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
Name Method for a Business Object	Yes	Yes	Yes	Yes	Yes	Yes
Release Method for a Business Object	No	No	No	No	No	Yes

Business Service Methods

Table 13 lists the technologies you can use to access business service methods.

Table 13.   Business Service Methods

Method	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
Business Service Methods	Yes	Yes	No	Yes	Yes	Yes	Yes
GetNextProperty Method for a Business Service	Yes	Yes	No	Yes	Yes	Yes	Yes
GetProperty Method for a Business Service	Yes	Yes	No	Yes	Yes	Yes	Yes
InvokeMethod Method for a Business Service	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Name Method for a Business Service	Yes	Yes	Yes	Yes	Yes	Yes	Yes
PropertyExists Method for a Business Service	Yes	Yes	No	Yes	Yes	Yes	Yes
Release Method for a Business Service	No	No	No	No	No	No	Yes

Table 13.   Business Service Methods

Method	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
RemoveProperty Method for a Business Service	Yes	Yes	No	Yes	Yes	Yes	Yes
SetProperty Method for a Business Service	Yes	Yes	No	Yes	Yes	Yes	Yes

Control Methods

You can use the following control methods. You can use these methods only with Browser Script:

- ■ "Applet Method for a Control" on page 296

- ■ "BusComp Method for a Control" on page 297

- ■ "GetProperty Method for a Control" on page 297

- ■ "GetValue Method for a Control" on page 298

- ■ "Name Method for a Control" on page 299

- ■ "SetLabelProperty Method for a Control" on page 299

- ■ "SetProperty Method for a Control" on page 303

- ■ "SetValue Method for a Control" on page 304

Property Set Methods

Table 14 lists the technologies you can use to access property set methods.

Table 14.   Property Set Methods

Method	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
AddChild Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Copy Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GetByteValue Method for a Property Set	No	No	No	No	No	No	Yes

Table 14.   Property Set Methods

Method	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
GetChild Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GetChildCount Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GetFirstProperty Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GetLastErrCode Method for a Property Set	No	No	No	Yes	No	No	No
GetLastErrText Method for a Property Set	No	No	No	Yes	No	No	No
GetNextProperty Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GetProperty Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GetPropertyCount Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GetType Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
GetValue Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Table 14.   Property Set Methods

Method	Server Script	Browser Script	Web Client Automation Server	Mobile Web Client Automation Server	COM Data Control	COM Data Server	Java Data Bean
InsertChildAt Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
PropertyExists Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
RemoveChild Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
RemoveProperty Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Reset Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SetByteValue Method for a Property Set	No	No	No	No	No	No	Yes
SetProperty Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SetType Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes
SetValue Method for a Property Set	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Miscellaneous Methods

Table 15 lists technologies you can use to access other methods that you can use. You cannot use
these methods with the following technologies:

■   Web Client Automation Server

■   Mobile Web Client Automation Server

■   COM Data Control

■ COM Data Server

Table 15.   Miscellaneous Methods

Method	Server Script	Browser Script	Java Data Bean
GetErrorCode Method	No	No	Yes
GetErrorMessage Method	No	No	Yes
TheApplication Method	Yes	Yes	No

Technologies You Can Use to Access Object Interface Events

This topic lists the types of object interface events. It includes the following topics:

■ "Applet Events" on page 100

■ "Application Events" on page 101

■ "Business Component Events" on page 101

■ "Business Service Events" on page 102

These object interface events are available in Server Script or Browser Script in Siebel Tools.

Applet Events

Table 16 lists applet events. You can use these events only with Server Script or Browser Script.

Table 16.   Applet Events

Event	Server Script	Browser Script
Applet_ChangeFieldValue Event	No	Yes
Applet_ChangeRecord Event	No	Yes
Applet_InvokeMethod Event	No	Yes
Applet_Load Event	No	Yes
Applet_PreInvokeMethod Event	No	Yes
WebApplet_InvokeMethod Event	Yes	No
WebApplet_Load Event	Yes	No
WebApplet_PreCanInvokeMethod Event	Yes	No
WebApplet_PreInvokeMethod Event	Yes	No

Table 16. Applet Events

Event	Server Script	Browser Script
WebApplet_ShowControl Event Not available in high interactivity mode.	Yes	No
WebApplet_ShowListColumn Event Not available in high interactivity mode.	Yes	No

Application Events

Table 17 lists application events. You can use these events only with Server Script or Browser Script.

Table 17. Application Events

Event	Server Script	Browser Script
Application_Close Event	Yes	No
Application_InvokeMethod Event	Yes	Yes
Application_Navigate Event	Yes	No
Application_PreInvokeMethod Event	Yes	Yes
Application_PreNavigate Event	Yes	No
Application_Start Event	Yes	No

Business Component Events

Table 18 lists business component events. You can use these events only with Server Script or Browser Script.

Table 18. Business Component Events

Event	Server Script	Browser Script
BusComp_Associate Event	Yes	No
BusComp_ChangeRecord Event	Yes	No
BusComp_CopyRecord Event	Yes	No
BusComp_DeleteRecord Event	Yes	No
BusComp_InvokeMethod Event	Yes	No
BusComp_NewRecord Event	Yes	No
BusComp_PreAssociate Event	Yes	No
BusComp_PreCopyRecord Event	Yes	No

Table 18.   Business Component Events

Event	Server Script	Browser Script
BusComp_PreDeleteRecord Event	Yes	No
BusComp_PreGetFieldValue Event	Yes	No
BusComp_PreInvokeMethod Event	Yes	No
BusComp_PreNewRecord Event	Yes	No
BusComp_PreQuery Event	Yes	No
BusComp_PreSetFieldValue Event Available only in high interactivity mode. Requires you to set a field property for the event that Siebel CRM immediately runs on the Siebel Server.	Yes	Yes
BusComp_PreWriteRecord Event	Yes	No
BusComp_Query Event	Yes	No
BusComp_SetFieldValue Event	Yes	No
BusComp_WriteRecord Event	Yes	No

Business Service Events

Table 19 lists business service events. You can use these events only with Server Script or Browser Script.

Table 19.   Business Service Events

Event	Server Script	Browser Script
Service_InvokeMethod Event	Yes	Yes
Service_PreCanInvokeMethod Event	Yes	Yes
Service_PreInvokeMethod Event	Yes	Yes

Object Interfaces Reference

This topic describes reference information for Siebel object interfaces. It includes the following topics:

■ "Applet Methods" on page 103

■ "Applet Events" on page 109

■ "Application Methods" on page 124

■ "Application Events" on page 179

CAUTION: Oracle might modify or delete an undocumented method without notice. Use of an undocumented method is entirely at your own risk.

About Specialized and Custom Methods

A *specialized method* is a Siebel object interface method that references one of the following specialized class:

■ A specialized applet class

■ A specialized business component class

A *specialized applet class* or a *specialized business component class* is a class other than the CSSFrame class or the CSSBusComp class.

A *custom method* is a Siebel object interface method that you modify.

Applet Methods

This topic describes applet methods. It includes the following topics:

In these methods, the Applet variable represents an applet instance.

ActiveMode Method for an Applet

The ActiveMode method returns a string that contains the name of the current Web template mode.

Format
Applet.ActiveMode

No arguments are available.

Used With
Browser Script

Examples
The following example is in Browser Script:

```
function Applet_Load ()
{
    var currMode = this.ActiveMode();
    theApplication().SWEAlert("The active mode for the selected applet is: " +
        currMode);
}
```

BusComp Method for an Applet
The BusComp method when used in the context of an applet returns the current business component instance that this applet references.

Format
Applet.BusComp();

No arguments are available.

Used With
Browser Script, Server Script

BusObject Method for an Applet
The BusObject method returns the name of the business object that the business component references.

Format
Applet.BusObject()

No arguments are available.

Used With
Browser Script, Server Script

Examples

The following example is in Browser Script:

```
function Applet_Load ()
{
    var appletname = this.Name();
    var currBO = this.BusObject();
    var currBOName = currBO.Name();
    theApplication().SWEAlert("The active Business Object for the " + appletname +
        " is: " + currBOName);
}
```

The following example is in Siebel eScript:

```
function WebApplet_Load ()
{
    var busObj = this.BusObject();
}
```

The following example is in Siebel VB:

```
Sub WebApplet_Load
    Dim oBusObject As BusObject
    Set oBusObject = Me.BusObject

End Sub
```

FindActiveXControl Method for an Applet

The FindActiveXControl method returns the name of a control that is a Document Object Model element.

Format

Applet.FindActiveXControl(*controlName*)

Table 20 describes the arguments for the Browser Script format of the FindActiveXControl method.

Table 20.   Arguments for the Browser Script Format of the FindActiveXControl Method

Argument	Description
controlName	Literal string or string variable that contains the name of the control.

Usage

You can use the FindActiveXControl method to find a control on a form applet. It does not locate a list column on a list applet.

Used With

Browser Script

Examples

The following Browser Script example interacts with a Microsoft slide ActiveX control that resides on a Siebel applet:

```
// Get a reference to the control
var SlideCtrl = FindActiveXControl("SliderControl");

// Display some of the ActiveX Control's properties
theApplication().SWEAlert ("element id = " + SlideCtrl.id);
theApplication().SWEAlert ("Max ticks = " + SlideCtrl.Max);

SlideCtrl.SelStart = 2;  // Set a control property
SlideCtrl.Refresh();    // Call the control's Refresh method

var myCustomCtrl = FindActiveXControl("TestControl");
myCustomCtrl.TestProperty01 = "abc";
myCustomCtrl.Style.visibility = "hidden"; // Use a Style sheet property
```

FindControl Method for an Applet

The FindControl method returns the name of a control. This applet must be part of the view that Siebel CRM displays.

Format

Applet.FindControl(*controlName*)

The arguments you can use with this format are the same as the arguments described in .

Usage

The FindControl method does not do the following:

■ Locate a control in an MVG applet, pick applet, associate applet, or detail applet. In Siebel Tools, these applets do not appear in the child View Web Template Items list of the view.

■ Locate list columns in a list applet.

Used With

Browser Script

Examples of Using the FindControl Method

The following example is in Browser Script:

```
function Applet_PreInvokeMethod (name, inputPropSet)
   {
   // Code to modify the Font Size of the "Location" label
   if (name == "fontsize")
      {
      // Use FindControl() to get a reference to the control
      var ctl = this.FindControl("Location");
```

```
            ctl.SetLabelProperty("FontSize", "22"); // Set the font size
            return ("CancelOperation");
        }
    }
```

To use this example, see "SetLabelProperty Method for a Control" on page 299.

InvokeMethod Method for an Applet

The InvokeMethod method calls a specialized method. It returns the following:

■   In Server Script, returns a string that contains the result of the method.

■   In Browser Script, returns a property set.

Browser Script Format

Applet.InvokeMethod(*methodName, methodArgs_PropSet);*

Table 21 describes the arguments for the Browser Script format of the InvokeMethod method.

Table 21.   Arguments for the Browser Script Format of the InvokeMethod Method

Argument	Description
methodName	The name of the method.
methodArgs_PropSet	Property set that contains the method arguments.

Server Script Format

Applet.InvokeMethod(*methodName, methArg1, methArg2, methArgN);*

Table 22 describes the arguments for the Browser Script format of the InvokeMethod method.

Table 22.   Arguments for the Browser Script Format of the InvokeMethod Method

Argument	Description
methodName	The name of the method.
You can use the following arguments: ■ methArg1 ■ methArg2 ■ methArgN	One or more strings that contain arguments for the methodName argument.

Usage

Available with Server Script and Browser Script. Note the following:

■ If the method that the methodName argument identifies exists in the browser, then Siebel CRM runs this method in the browser.

■ If the method that the methodName argument identifies exists on the Siebel Server, then Siebel CRM runs this method on the Siebel Server.

Caution About Using the InvokeMethod Method

You must use InvokeMethod only to call a method that this book describes.

Used With

Browser Script, Server Script

Examples

The following example is in Siebel eScript:

```
function WebApplet_PreInvokeMethod (MethodName)
{
    //Call a Siebel SmartScript from a custom button
    //using the applet.InvokeMethod method
    //Note the InvokeSScriptFromButton is from a custom
    //method added to a button
    if (MethodName == "InvokeSScriptFromButton")
    {
        var iReturn = ContinueOperation;
        var sArgs = new Array(3);
        sArgs[0] = "Demo Opportunity Profile";
        sArgs[1] = "";
        sArgs[2] = "";
        this.InvokeMethod("RunCallScript", sArgs);
        iReturn = CancelOperation;
    }
    else
    {
        iReturn = ContinueOperation;
    }
    return(iReturn);
}
```

Name Method for an Applet

The Name method for an applet returns the name of an applet.

Format

Applet.Name()

No arguments are available.

Used With

Browser Script, Server Script

Examples

The following example is in Browser Script:

```
function Applet_Load ()
    {
    //Display the name of the applet if the applet loads using the
    //applet.Name() method that gets the name of the applet
    var appletName;
    appletName = this.Name();
    theApplication().SWEAlert("The name of the applet is: " + appletName);
}
```

The following example is in Siebel eScript:

```
function WebApplet_Load ()
{
    //Display the name of the applet if the applet loads using the
    //applet.Name() method that gets the name of the applet
    var appletName;
    appletName = this.Name();
    TheApplication().RaiseErrorText("The name of the applet is: " + appletName);
}
```

The following example is in Siebel VB:

```
Sub WebApplet_Load
' Display the name of the applet if the applet loads using the
' applet.Name() method that gets the name of the applet
Dim appletName As String
appletName = Me.Name
TheApplication.RaiseErrorText "The name of the applet is: " & appletName
End Sub
```

Applet Events

This topic describes applet events. It includes the following topics:

Overview of Applet Events

Siebel CRM calls an applet event in reply to a user interaction. You can manage each event for each applet. You can use an applet event only in high interactivity mode.

The format for an applet event that you use on the browser is Applet_*event*.

where:

■ *event* is the name of the event.

For example, Applet_ChangeFieldValue. If the event includes the Applet prefix, then you can use it only on the browser.

The format for an applet event that you use on the Siebel Server is WebApplet_*event*.

where:

■ *event* is the name of the event.

For example, WebApplet_InvokeMethod. If the event includes the WebApplet prefix, then you can use it only on the Siebel Server.

Applet_ChangeFieldValue Event

The Applet_ChangeFieldValue event starts if the user uses an applet to modify data in a field. It does not return any information. For more information, see "Applet_ChangeRecord Event" on page 111.

Format

Applet_ChangefieldValue(fieldname, fieldValue)

Table 23 describes the arguments for the Applet_ChangeFieldValue event.

Table 23.   Arguments for the Applet_ChangeFieldValue Event

Argument	Description
FieldName	A string that contains the name of the field that the user modified.
FieldValue	A string that contains the value that the user modified.

Usage

Note the following usage of the Applet_ChangeFieldValue event:

■ If the user moves to a different record but does not modify a value in the previous record, then the ChangeFieldValue event does not start.

■ If the user modifies the value of a field, and if Siebel CRM modifies the value in another field that depends on some way on the value that the user modified, such as a calculated field, then the event starts once for each field whose value Siebel CRM modifies.

■ If the user uses a pick applet or popup applet to modify the data that a field contains, then this event does not start.

Used With

Browser Script

Examples

The following example is in Browser Script:

```
function Applet_ChangeFieldValue (field, value)
{
    try
    {
        switch (field)
        {
            case "Primary Revenue Committed Flag":
            if (value == "Y")
            {
                var thisBC = this.BusComp();
                var sRev = thisBC.GetFieldValue("Primary Revenue Amount");
                var sUpside = thisBC.GetFieldValue("Primary Revenue Upside Amount");
                var total = sRev + sUpside;
                if (total < 500000)
                {
                    thisBC.SetFieldValue("Primary Revenue Committed Flag", "N");
                    theApplication().SWEAlert("Changing the Committed Flag to NO as
                        $500,000 in Revenue and Upside amount is required");
                }
            }
            break;
        }
    }
    catch(e)
    {
// error handling routine
    }
```

Applet_ChangeRecord Event

Siebel CRM calls the Applet_ChangeRecord event if the user moves to a different record or view. It does not return any information. For more information, see .

Format

Applet_ChangeRecord()

No arguments are available.

Used With

You use the Applet_ChangeRecord event with Browser Script. Note the following:

■ To return the value of the field the user navigates to, use the BusComp.GetFieldValue method.

■ To return the value of the field the user navigates away from, use the control.GetValue method.

Examples

The following example is in Browser Script:

```
function Applet_ChangeRecord ()
{
    try
    {
        var thisBC = this.BusComp();
        var sFlag = thisBC.GetFieldValue("Primary Revenue Committed Flag");
        if (sFlag == "Y")
        {
            theApplication().SWEAlert("This record cannot be updated because it has
            been committed");
        }
    }
    catch(e)
// error handling routine

}
```

Applet_InvokeMethod Event

The Applet_InvokeMethod event can start if any of the following items occur:

■ A call to applet.InvokeMethod occurs

■ A call to a specialized method occurs

■ A user chooses a menu item in a menu that the user defines

For more information, see "About Specialized and Custom Methods" on page 103.

This method does not return any information.

Format

Applet_InvokeMethod(*name*, *inputPropSet*)

Table 24 describes the arguments for the Applet_InvokeMethod event.

Table 24.　Arguments for the Applet_InvokeMethod Event

Argument	Description
name	The name of the method that Siebel CRM calls.
inputPropSet	A property set that identifies arguments that Siebel CRM sends to the event.

Usage

This method sends information you specify in the inputPropSet argument to the PreInvokeMethod event. You can use the Applet_InvokeMethod event to display or hide controls, or to set a search specification. To access a business component from this event handler, do the following:

■ Use this.BusComp.

■ Do not use TheApplication.ActiveBusComp.

Used With
Browser Script

Examples

Some methods can create, modify, or delete records. These actions might call an event at the applet or business component level. If you require Siebel CRM to perform a specific action before or after the method run, then you can use these events. The following example includes custom code in the PreInvokeMethod event and the InvokeMethod applet event. For more information, see "Applet_PreInvokeMethod Event" on page 115.

To set the fields, this code sets and resets the flag and uses the NewRecord server event:

```
function Applet_PreInvokeMethod (name, inputPropSet)
{
    if (name == "Quote")
    {
        //  Add code that Siebel CRM must run BEFORE the special method
        //  Set flag to "1"
        theApplication().SetProfileAttr("flag","1");
    }

    return ("ContinueOperation");
}



function Applet_InvokeMethod (name, inputPropSet)
{
    if (name == "Quote")
    {
        //  Add code that Siebel CRM must run AFTER the special method
```

```
        //  Reset the flag to "0"
        theApplication().SetProfileAttr("flag","0");
    }

}


    function BusComp_NewRecord ()
    {
        if (theApplication().GetProfileAttr("flag")== "1" )
        {
            this.SetFieldValue ("Field1", "Value1");
            this.SetFieldValue ("Field2", "Value2");
            . . . . .
        }

    }
```

Applet_Load Event

Siebel CRM calls the Applet_Load event after it loads an applet and displays the data for that applet.
It does not return any information.

Format

Applet_Load()

No arguments are available.

Usage

To hide or manipulate controls or to set properties on an ActiveX Control in a form applet, you can
use the Applet_Load event. You can manipulate the following types of controls:

■ CheckBox

■ ComboBox

■ TextBox

■ TextArea

■ Label

If you must display a dialog box, then do not use the SWEAlert method or the RaiseErrorText method
with the Applet_Load event. This technique can cause the browser to fail if Siebel CRM has not fully
rendered the Siebel application in the browser.

Used With

Browser Script

Examples

You can use the following example only with code on a form applet:

```
function Applet_Load ()
{
    // Get the control instance.
    var ctrl = this.FindControl("FirstName");

    // Hide the control
    ctrl.SetProperty("Visible","false");

    // Hide the label
    ctrl.SetLabelProperty("Visible", "hidden");
}
```

Applet_PreInvokeMethod Event

Siebel CRM calls the Applet_PreInvokeMethod event immediately before it calls a specialized method on an applet. The Applet_PreInvokeMethod event can start if any of the following items occur:

■ A call to the InvokeMethod method on an applet occurs.

■ A user chooses a custom menu item that you define in Siebel Tools.

This event returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

For more information, see "About Specialized and Custom Methods" on page 103.

Format

Applet_PreInvokeMethod(*name, inputPropSet*)

The arguments you use with this format are the same as the arguments described in Table 24 on page 113.

Used With

Browser Script

Examples

```
function Applet_PreInvokeMethod (name, inputPropSet)
{
    if(name == 'NewRecord')
    {
        if(confirm("Are you sure you want to create a new record?"))
            return ("ContinueOperation");
        else
            return ("CancelOperation");
        return ("ContinueOperation");
    }
}
```

WebApplet_InvokeMethod Event

Siebel CRM calls the WebApplet_InvokeMethod event after a specialized method on the Web applet runs. WebApplet_InvokeMethod starts only for a predefined method. It does not start for a custom method. For more information, see "About Specialized and Custom Methods" on page 103.

This method does not return any information.

Format

WebApplet_InvokeMethod(*methodName)*

Table 25 describes the arguments for the WebApplet_InvokeMethod event.

Table 25.   Arguments for the WebApplet_InvokeMethod Event

Argument	Description
methodName	String variable or literal that contains the name of the method that Siebel CRM calls.

Used With

Server Script

Examples

The following example is in Siebel eScript:

```
switch (MethodName)
{
    case "NewQuery":
        TheApplication().SetSharedGlobal("EnableButton", "N"); break;
    case "ExecuteQuery":
        TheApplication().SetSharedGlobal("EnableButton", ""); break;
    case "UndoQuery":
        TheApplication().SetSharedGlobal("EnableButton", "");
    break;
}
```

The following example is in Siebel VB:

```
Select Case MethodName
Case "NewQuery"
    TheApplication.SetSharedGlobal "EnableButton", "N"
Case "ExecuteQuery"
    TheApplication.SetSharedGlobal "EnableButton", ""
Case "UndoQuery"
    TheApplication.SetSharedGlobal "EnableButton", ""
End Select
```

Related Topics

For more information, see the following topics:

WebApplet_Load Event

Siebel CRM calls the WebApplet_Load event immediately after it loads an applet. It does not return any information.

Format

WebApplet_Load()

No arguments are available.

Usage

To avoid returning a null value, do not call TheApplication.ActiveBusObject from the WebApplet_Load event. Instead, you can use this.BusObject to get a reference to the current business object.

Used With

Server Script

Examples

The following example is in Siebel eScript:

```
function WebApplet_Load ()
{
    try
    {
        var currBC = this.BusComp();
        with (currBC)
        {
            SetViewMode(OrganizationView);
            ClearToQuery();
            SetSearchSpec("Last Name", "A*");
            ExecuteQuery(ForwardBackward);
        }
    }
    catch (e)
    {
        TheApplication().RaiseErrorText(e.errText);
    }
}
```

The following example is in Siebel VB:

```
Sub WebApplet_Load
    Dim iReturn As Integer
    Dim currBC As BusComp
```

```
    Set currBC = Me.BusComp
    With currBC
        .SetViewMode OrganizationView
        .ClearToQuery
        .SetSearchSpec "Last Name", "A*"
        .ExecuteQuery
    End With
End Sub
```

Related Topics

For more information, see the following topics:

- "Applet_InvokeMethod Event" on page 112

- "Application_InvokeMethod Event" on page 180

- "WebApplet_PreCanInvokeMethod Event" on page 118

WebApplet_PreCanInvokeMethod Event

The WebApplet_PreCanInvokeMethod event allows a script to determine if the user possesses the authority to call the applet method. Siebel CRM calls this method in the following situations:

- Before it calls the PreInvokeMethod event.

- If the user steps to a different record.

- If it loads an applet.

- If it calls a different method from Browser Script. For example, the GetProfileAttr method or the SetProfileAttr method.

This method returns CancelOperation or ContinueOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format

WebApplet_PreCanInvokeMethod(*MethodName, &CanInvoke*)

Table 26 describes the arguments for the WebApplet_PreCanInvokeMethod event.

Table 26.   Arguments for the WebApplet_PreCanInvokeMethod Event

Argument	Description
MethodName	A string that contains the name of the method that Siebel CRM must run.
&CanInvoke	A string that indicates if Siebel CRM call the applet method. You can use the following values: ■ **TRUE.** Siebel CRM can call the applet method. ■ **FALSE.** Siebel CRM cannot call the applet method.

Usage

Using the FirstSelected business component method with the PreCanInvokeMethod event can cause unexpected behavior in a pick applet that Siebel CRM calls from the applet where this event is called.

To enable and disable a method, it can be easier to use the CanInvokeMethod applet user property at the applet level. For an example, see "Using a MiniButton Control to Call a Custom Method" on page 78. For information about the CanInvokeMethod user property, see *Siebel Developer's Reference*.

Used With

Server Script

Examples

The following example is in Siebel eScript:

```
function WebApplet_PreCanInvokeMethod (MethodName, &CanInvoke)
{
    if ( MethodName == "CustomMethod" )
    {
        CanInvoke = "TRUE";
        return( CancelOperation );
    }
    return (ContinueOperation);
}
```

The following example is in Siebel VB:

```
Function WebApplet_PreCanInvokeMethod (MethodName As String, CanInvoke As String)
As Integer
    Dim iReturn As Integer
    iReturn = ContinueOperation
    If MethodName = "Test" Then
        CanInvoke = "TRUE"
        iReturn = CancelOperation
    End If
    WebApplet_PreCanInvokeMethod = iReturn
End Function
```

WebApplet_PreInvokeMethod Event

Siebel CRM calls the WebApplet_PreInvokeMethod event before it calls any of the following:

■ A specialized method for the Web applet.

■ A custom method that Siebel CRM calls through the oWebApplet object of the InvokeMethod method.

This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

For more information, see "About Specialized and Custom Methods" on page 103.

Format

WebApplet_PreInvokeMethod(*methodName*)

The arguments you can use with this format are the same as the arguments described in .

Used With

Server Script

Examples

The following example is in Siebel eScript:

```
function WebApplet_PreInvokeMethod (MethodName)
{
    switch (MethodName)
    {
    case "CustomMethod":
        var applet = this;
        var BC = applet.BusComp();
        var ConId = BC.GetFieldValue("Contact Id");
        var WshShell = COMCreateObject("WScript.Shell");
        WshShell.Popup("My Custom Method was called. Here is the ID " + ConId);
        return(CancelOperation);
        break;
    }
    return (ContinueOperation);
}
```

The following example is in Siebel VB:

```
Function WebApplet_PreInvokeMethod (MethodName As String) As Integer
    Dim iReturn As Integer
    iReturn = ContinueOperation
    Select Case MethodName
    Case "CustomMethod"
        Dim oBusComp As BusComp
        Set oBusComp = Me.BusComp
        Dim WshShell As Object
        ConId = oBusComp.GetFieldValue("Contact Id")
        Set WshShell = CreateObject("WScript.Shell")
        WshShell.Popup("My Custom Method was called. Here is the ID " & ConId)
        iReturn = CancelOperation
    End Select
    WebApplet_PreInvokeMethod = iReturn
End Function
```

WebApplet_ShowControl Event

The WebApplet_ShowControl event allows a script to modify the HTML that the Siebel Web Engine creates when it renders a control on a Web page in a Siebel application that runs in standard interactivity mode. You can use the WebApplet_ShowControl event only in a Siebel application that runs in standard interactivity mode. This event does not return any information.

Format

WebApplet_ShowControl (*controlName*, *property*, *mode, HTML*)

Table 27 describes the arguments for the WebApplet_ShowControl event.

Table 27.   Arguments for the WebApplet_ShowControl Event

Argument	Description
controlName	A string that indicates the name of the control that Siebel CRM must render.
property	A string that indicates the value of the property attribute of the swe:control tag or the swe:this tag that starts this event. If you do not specify the property attribute for the tag, then you can leave the property argument empty.
mode	The mode of the applet that Siebel CRM displays. You can use one of the following values: ■ Base ■ Edit ■ New ■ Query ■ Sort
HTML	The HTML that the Siebel Web Engine creates for the swe:control or swe:this tag that starts this event.

Usage of the WebApplet_ShowControl Event

The HTML that the Siebel Web Engine creates depends on the following items:

■ The control

■ The property displayed

■ The mode of the applet

The script can modify the value of the HTML argument. The Siebel Web Engine sends the modified value back to the Web browser.

To render the layout of an applet, Siebel CRM uses a Siebel Web Template (.swt) file. These files are HTML files that contain special variable tags that indicate where to render a control. To render the controls on the Web page, the Siebel Web Engine converts swe tags to HTML. Siebel CRM calls the WebApplet_ShowControl event for each swe tag after the Siebel Web Engine creates the HTML to render the control, but before it sends the created HTML back to the browser. This technique allows the scripts to modify this HTML before Siebel CRM displays it.

To display a control, you can include swe:control variable tags in the following ways:

- Use the swe:control tag by itself:

  `<swe:control id="1" property="DisplayName"/>`

  If the control ID is mapped to an actual control in the applet, then the Siebel Web Engine renders the DisplayName property of the control at the point where you place this tag in the template file. Siebel CRM starts the event only one time after the Siebel Web Engine creates the HTML for the swe:control tag.

- Use the swe:control tag and the swe:this tag:

  ```
  <swe:control id="1">
  .
  .
  .
  <swe:this property="DisplayName"/>
  .
  .
  .
  </swe:control>
  ```

  The Siebel Web Engine renders the DisplayName property of the control at the point where you place the swe:this tag in the template file. Siebel CRM uses the outer swe:control tag only to determine if the control ID is mapped to an actual control in the applet. Siebel CRM starts the event two times:

  - After the Siebel Web Engine creates the HTML for the swe:this tag.

  - After the Siebel Web Engine creates the HTML for the outer swe:control tag. This work occurs after Siebel CRM converts to HTML all objects that the code references between the swe:control tag and the /swe:control tag. This conversion includes objects in the swe:this tag.

To distinguish between these two event calls, the script can examine the value of the property attribute of the tag that Siebel CRM passes as an argument to the event.

Used With

Server Script

Examples

This Siebel eScript script displays negative amounts in red in a read-only form:

```
function WebApplet_ShowControl (ControlName, Property, Mode, &HTML)
{
    var BC = this.BusComp();
    if( ControlName == "Amount" && Mode == "Base" && Property == "FormattedHTML")
    {
        var amount = ToNumber(BC.GetFieldValue ("Transaction Amount"));
        if (amount < 0)
            HTML = "<FONT Color=Red> " + HTML + " </FONT>";
    }
}
```

WebApplet_ShowListColumn Event

This event allows a script to modify the HTML that the Siebel Web Engine creates when it renders a list column on a Web page in a Siebel application that runs in standard interactivity mode. You can use the WebApplet_ShowListColumn event only in a Siebel application that runs in standard interactivity. This event does not return any information.

Format

WebApplet_ShowListColumn (*columnName*, *property*, *mode*, *HTML*)

Table 28 describes the arguments for the WebApplet_ShowListColumn event.

Table 28.   Arguments for the WebApplet_ShowListColumn Event

Argument	Description
columnName	A string that indicates the name of the list column that Siebel CRM must render.
Other arguments: ■ Property ■ Mode ■ HTML	For more information, see Table 27 on page 121.

Usage

Usage for the WebApplet_ShowListColumn event is very similar to usage for the WebApplet_ShowControl event, except Siebel CRM uses a list column ID that is mapped to a list column in an applet. For more information, see "Usage of the WebApplet_ShowControl Event" on page 121.

Used With

Server Script

Example

This Siebel VB script displays negative amounts in a list in red font color:

```
Sub WebApplet_ShowListColumn (ColumnName As String, Property As String, Mode As
String, HTML As String)

Dim amount as Double

If ColumnName = "Amount" and Mode = "Base" and Property = "FormattedHTML" Then
   If HTML < 0 Then
       HTML = "<FONT Color=Red> " + HTML + " </FONT>"
   End If
End If
End Sub
```

The following example is in Siebel eScript:

```
function WebApplet_ShowListColumn (ColumnName, Property, Mode, &HTML)
{
    if ((ColumnName == 'Amount') && (Mode == "Base") && (Property == "FormattedHTML"))
    {
            var val - HTML.valueOf();
            if (val < 0)
                HTML = "<FONT Color=Red> " + HTML + " </FONT>";
            }
}
```

Application Methods

This topic describes application methods. It includes the following topics:

- "Overview of Application Methods" on page 125

- "ActiveApplet Method for an Application" on page 126

- "ActiveBusComp Method for an Application" on page 127

- "ActiveBusObject Method for an Application" on page 127

- "ActiveViewName Method for an Application" on page 129

- "Attach Method for an Application" on page 130

- "CurrencyCode Method for an Application" on page 133

- "Detach Method for an Application" on page 133

- "EnableExceptions Method for an Application" on page 134

- "FindApplet Method for an Application" on page 136

- "GetBusObject Method for an Application" on page 136

- "GetDataSource Method for an Application" on page 138

- "GetLastErrCode Method for an Application" on page 138

- "GetLastErrText Method for an Application" on page 139

- "GetProfileAttr Method for an Application" on page 140

Overview of Application Methods

An *application method* is a predefined methods that return the current Siebel application object instance:

■ TheApplication, if called from Siebel VB that resides in the Siebel repository file (SRF)

■ TheApplication(), if called from Siebel eScript that resides in the Siebel repository file (SRF)

■ theApplication(), if called from Browser Script that resides in the Siebel repository file (SRF)

Note the following:

■ If an application method applies to only one scripting language, then the Syntax definition in the method includes one of these methods.

■ If a method applies to an external interface or to more than one scripting language, then it must use more than one format. In this situation, the Syntax definition includes Application and results in the following situation:

■ If you use Siebel VB, Siebel eScript, or Browser Script, then Siebel CRM substitutes the applicable statement for Application

■ If you use an external interface, then Siebel CRM substitutes the name of an application instance for Application

Some examples in this chapter include an Application method that uses an external interface. These examples use SiebelApplication as the application instance. The examples assume that the script starts an instance of the Siebel application. This situation is true even if the example does not include the code that starts this instance.

ActiveApplet Method for an Application
The ActiveApplet method returns a reference to the applet that Siebel CRM displays.

Format
theApplication().ActiveApplet();

No arguments are available.

Usage
Use this method to identify the applet that Siebel CRM currently displays. This applet typically includes a blue border to indicate that it is active.

Used With
Browser Script

Examples
```
function Applet_PreInvokeMethod (name, inputPropSet)
{
    switch (name)
    {
        case "Drilldown":
            var activeapplet = theApplication().ActiveApplet();
            var activeappletname = activeapplet.Name();
            theApplication().SWEAlert("Here is the applet we are drilling down from "
            + activeappletname);
```

```
        break;
    }
    return ("ContinueOperation");
}
```

ActiveBusComp Method for an Application

The ActiveBusComp method returns the name of the business component that the active applet references.

Format

theApplication().ActiveBusComp();

No arguments are available.

Used With

Browser Script

Examples

```
function Applet_Load ()
{
    var activeBC = theApplication().ActiveBusComp();
    activeBC = activeBC.Name();
    theApplication().SWEAlert(activeBC);
}
```

ActiveBusObject Method for an Application

The ActiveBusObject method returns the name of the business object that the active view references.

Format

*Application.*ActiveBusObject

No arguments are available.

Usage for the ActiveBusObject Method

Do not use the ActiveBusObject method in an event handler that any of the following technologies can start:

■ COM Data Server

■ COM Data Control

■ Siebel Java Data Bean

Used With

Browser Script, Mobile Web Client Automation Server, Server Script

Example in Browser Script

The following example is in Browser Script:

```
function Applet_Load ()
{
    var oBusObj;
    oBusObj = theApplication().ActiveBusObject();
    theApplication().SWEAlert("The active business object is " + oBusObj.Name() +
".")
}
```

Example of Using the ActiveBusObject Method to Call from a Custom Button on a Child Applet

The following examples include script that runs on the Siebel Server that Siebel CRM can call from a custom button on a child applet in a view. This script does the following work:

1 Determines if the Contact business object is active. If it is active, then Siebel CRM returns the email address of the currently active parent Contact record.

2 Uses the contact email address to call the custom SendEmail function.

Objects that the script references are currently active in the Siebel client, so Siebel CRM does not delete these objects at the end of the script.

The following example is in Siebel eScript:

```
function WebApplet_PreInvokeMethod (MethodName)
{
    if (MethodName == "Send Email")
    {
        var oBO = TheApplication().ActiveBusObject();

        if (oBO.Name() == "Contact")
        {
          var oBC = oBO.GetBusComp("Contact");
           var sEmail = oBC.GetFieldValue("Email Address");

           SendMail(sEmail);

           sEmail ="";
        }
        return (CancelOperation);
    }
    return (ContinueOperation);
}
```

The following example is in Siebel VB:

```
Function WebApplet_PreInvokeMethod (MethodName As String) As Integer

    Dim iRtn As Integer
    iRtn = ContinueOperation
```

```
If MethodName = "Send Email" Then

   Dim oBO As BusObject
    Set oBO = TheApplication.ActiveBusObject()

   If oBO.Name() = "Contact" Then

       Dim oBC As BusComp
       Dim sEmail As String

       Set oBC = oBO.GetBusComp("Contact")

       sEmail = oBC.GetFieldValue("Email Address")

       SendEmail(sEmail)

       sEmail =""

   End If

   iRtn = CancelOperation

End If

WebApplet_PreInvokeMethod = iRtn
End Function
```

ActiveViewName Method for an Application

The ActiveViewName method returns the name of the active view.

Format

*Application.*ActiveViewName

No arguments are available.

Usage

Usage for the ActiveViewName method is very similar to usage for the ActiveViewName method. For more information, see .

Used With

Browser Script, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel eScript:

```
function BusComp_PreSetFieldValue (FieldName, FieldValue)
{
    switch(FieldName)
    {
    case "Name":
```

```
    case "Location":
    case "Account Status":
    case "Alias":
    case "City":
    case "Country":
    case "Currency Code":
    case "Current Volume":
    case "DUNS Number":
    case "Expertise":
    case "Freight Terms":
    case "Freight Terms Info":
    case "Home Page":
    case "Industry":
    case "Location":
    case "Main Phone Number":
    case "Main Fax Number":
    case "Sales Rep":
    var sActiveViewName = TheApplication().ActiveViewName();
    if (sActiveViewName == "All Accounts across Organizations")
    {
        TheApplication().RaiseErrorText("You cannot update the " + FieldName +
            " on the " + sActiveViewName + " View");
    }
    break;
    }
    return (ContinueOperation);
}
```

Attach Method for an Application

The Attach method allows an external application to reconnect to an existing Siebel session. It returns a Boolean value that indicates if Siebel CRM successfully ran the method.

Format

Application.Attach(*sessionString*)

Table 29 describes the arguments for the Attach method.

Table 29.   Arguments for the Attach Method

Argument	Description
sessionString	A string that contains the Siebel Session Id. This argument is typically the output of the Detach method.

Used With

COM Data Control, Siebel Java Data Bean

Examples

The examples in this topic do the following work:

1 Start an instance of COM Data Control.

2 Log in to a Siebel Server.

3 Detach the instance.

4 Determine the session string.

5 Start another instance of COM Data Control.

The script does not log in again. Instead, it uses the session string to access the existing session. This technique reuses the connection that the first instance created.

The following example uses COM Data Control and is written in native Visual Basic:

```
Dim SiebelApplication_first As SiebelDataControl
Dim SiebelApplication_second As SiebelDataControl
Dim errCode As Integer
Dim sessionString As String
Dim attachResult As Boolean
Dim errText As String

' Instantiate the first instance
Set SiebelApplication_first = CreateObject("SiebelDataControl.SiebelDataControl.1")

' Login to Siebel
SiebelApplication_first.Login "host=""Siebel.tcpip.none.none://virtual ip:port/
enterprise/object manager""", "user id", "password"

errCode = SiebelApplication_first.GetLastErrCode
If errCode <> 0 Then
    errText = SiebelApplication_first.GetLastErrText
    MsgBox errText
    Exit Sub
End If

' Detach this instance from Siebel and get session id
sessionString = SiebelApplication_first.Detach
MsgBox "The session string is: " & sessionString

' Instantiate the second instance
Set SiebelApplication_second =
CreateObject("SiebelDataControl.SiebelDataControl.1")

' Attach the existing session to this instance
attachResult = SiebelApplication_second.Attach(sessionString)
If (attachResult = True) Then
    MsgBox "Session attached!"
Else
    MsgBox "Session attach failed"
End If
```

```
SiebelApplication_second.LogOff
Set SiebelApplication_second = Nothing
Set SiebelApplication_first = Nothing
```

The following example uses the Siebel Java Data Bean:

```java
import com.siebel.data.*;
import com.siebel.data.SiebelException;

public class JDBAttachDetachDemo
{
    private SiebelDataBean m_dataBean_first = null;
    private SiebelDataBean m_dataBean_second = null;

    public static void main(String[] args)
    {
        JDBAttachDetachDemo demo = new JDBAttachDetachDemo();
    }

    public JDBAttachDetachDemo()
    {
      try
      {
          // Instantiate the Siebel Java Data Bean
          m_dataBean_first = new SiebelDataBean();

          // Login to the Siebel Servers
          m_dataBean_first.login("siebel.tcpip.none.none://virtualip:2320/
           enterprise/object manager name","user id","password");

          System.out.println("Logged in to the Siebel Server ");

          //Get the Detach Handle
          String detachHandle = m_dataBean_first.detach();
          System.out.println("The session id is: " + detachHandle);

          // Instantiate another Siebel Java Data Bean
          SiebelDataBean m_dataBean_second = new SiebelDataBean();

          // Do Attach
          System.out.println("Attaching in to the Siebel Server ");
          m_dataBean_second.attach(detachHandle);
          System.out.println("Attach Done ");

          // Logoff
          m_dataBean_second.logoff();
      }

      catch (SiebelException e)
      {
          System.out.println(e.getErrorMessage());
      }
    }
}
```

CurrencyCode Method for an Application

The CurrencyCode method returns the currency code that is associated with the division of the user position. For example, USD for U.S. dollars, EUR for the euro, or JPY for the Japanese yen.

Format

Application.CurrencyCode

No arguments are available.

Used With

Browser Script, COM Data Control, COM Data Server, Web Client Automation Server, and Server Script

Examples

The following example is in Siebel eScript:

```
function WebApplet_Load ()
{
    var currencycode;
    currencycode = TheApplication().CurrencyCode();
    var WshShell = COMCreateObject("WScript.Shell");
    WshShell.Popup(currencycode);
}
```

Detach Method for an Application

The Detach method returns a string that contains the Siebel session Id.

Format

Application.Detach

No arguments are available.

Usage

Use the string that the Detach method returns only with the Attach method.

Used With

COM Data Control, Siebel Java Data Bean

Examples

For a Siebel Java Data Bean example and a native VB example that uses COM Data Control, see .

EnableExceptions Method for an Application

The EnableExceptions method enables or disables native Component Object Model (COM) error handling. This method does not return any information.

Format

*Application.*EnableExceptions(*bEnable*)

Table 30 describes the arguments for the EnableExceptions method.

Table 30.   Arguments for the EnableExceptions Method

Argument	Description
bEnable	You can one of the following values: ■ TRUE ■ FALSE

Usage

Setting the argument to TRUE enables native error handling. This allows Siebel CRM to intercept and display the exception ID and description. Native COM error handling is disabled by default.

Used With

COM Data Control, Mobile Web Client Automation Server

Example of Using the EnableExceptions Method with Siebel ActiveX Data Control

The native Visual Basic script in this example does the following work:

■ Uses the Siebel ActiveX Data Control to connect to the Siebel application and to create an instance of a business object.

■ Prompts the user to use or not use the native error handling.

■ If the user answers yes, and if the script encounters an error, then it issues the error immediately.

■ If the user answers no, then the script suppresses errors.

You can detect errors only with the GetLastErrCode method.

The following code is an example of using the EnableExceptions method with Siebel ActiveX Data Control:

```
Dim SiebelApplication As SiebelDataControl
Dim errCode As Integer
Dim wrongBO As SiebelBusObject

Dim nativeHandle As String

Set SiebelApplication = CreateObject("SiebelDataControl.SiebelDataControl.1")
```

```
' Login to Siebel

SiebelApplication_first.Login "host=""Siebel.tcpip.none.none://virtual ip:port/
enterprise/object manager""", "user id", "password"

nativeHandle = InputBox("Use native error handling?", "", "Yes")

If nativeHandle = "Yes" Then
    SiebelApplication.EnableExceptions (True)
Else
    SiebelApplication.EnableExceptions (False)
End If

Set wrongBO = SiebelApplication.GetBusObject("No Such One") 'intended to create an
error at this line by instantiating a nonexisting Business Object

errCode = SiebelApplication.GetLastErrCode()
If errCode <> 0 Then 'if native error handle is disabled, this block detects it
    ErrText = SiebelApplication.GetLastErrText
    MsgBox ErrText
    Exit Sub
End If
```

Example of Using the EnableExceptions Method with Siebel Mobile Automation Server

The script in this example performs the same work that is described in "Example of Using the
EnableExceptions Method with Siebel ActiveX Data Control" on page 134, except it uses the Siebel
Mobile Automation Server:

```
Dim SiebelApp As SiebelWebApplication
Dim errCode As Integer
Dim wrongBO As SiebelBusObject

Set SiebelApp = CreateObject("TWSiebel.SiebelWebApplication.1")

Dim nativeHandle As String
nativeHandle = InputBox("Use native error handle?", "", "Yes")

If nativeHandle = "Yes" Then
    SiebelApp.EnableExceptions (True)
Else
    SiebelApp.EnableExceptions (False)
End If

Set wrongBO = SiebelApp.GetBusObject("No Such One") 'intended to create an error at
this line by instantiating a nonexisting Business Object

errCode = SiebelApp.GetLastErrCode()
If errCode <> 0 Then 'if native error handle is disabled, this block detects it
    ErrText = SiebelApp.GetLastErrText
    MsgBox ErrText
    Exit Sub
End If
```

FindApplet Method for an Application

The FindApplet method returns the name of an applet.

Format

theApplication().FindApplet(*appletName*)

Table 31 describes the arguments for the FindApplet method.

Table 31.   Arguments for the FindApplet Method

Argument	Description
appletName	String variable or literal that contains the name of an applet.

Usage

The only applets available are applets that are visible in the active view.

Used With

Browser Script

Examples

The following example is in Browser Script:

```
function Applet_ChangeFieldValue (field, value)
{
   if (theApplication().ActiveViewName() == "Account List View")
   {
      var newapplet = theApplication().FindApplet("Account Entry Applet");
      var entryappletcontrol = newapplet.FindControl("Name");
      var entryappletvalue = entryappletcontrol.GetValue();
      theApplication().SWEAlert(entryappletvalue);
   }
}
```

GetBusObject Method for an Application

The GetBusObject method creates a new instance of a business object. It returns the name of this new business object instance.

Format

Application.GetBusObject(*busObjectName*)

Table 32 describes the arguments for the GetBusObject method.

Table 32. Arguments for the GetBusObject Method

Argument	Description
busObjectName	String variable or literal that contains the name of the business object.

Usage

To delete the business object instance after it is no longer needed, you can set the business object to Nothing.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The examples in this topic create a new instance of the Account business object and returns the name of the instance of the Account business object.

The following example is in Siebel eScript:

```
var oBusObject = TheApplication().GetBusObject("Account");
var oBusComp = oBusObject.GetBusComp("Account");

Your custom code

oBusComp = null;
oBusObject = null;
```

The following example is in Siebel VB:

```
Dim AccntBO as BusObject
Dim AccntBC as BusComp
Dim AddrBC as BusComp
Set AccntBO = TheApplication.GetBusObject("Account")
Set AccntBC = AccntBO.GetBusComp("Account")

Your custom code

Set AccntBO = Nothing
Set AccntBC = Nothing
```

Examples of Using the GetBusObject Method to Refer to the Business Object That Is Currently Active

The name of the business object instance that Siebel CRM returns might vary depending on the location where it calls the code, such as a Web applet event. The examples in this topic are useful if you must refer to the business object instance that is currently active.

The following example is for Siebel Java Data Bean:

```
private SiebelDataBean m_dataBean = null;
private SiebelBusObject   m_busObject = null;
m_busObject = m_dataBean.getBusObject("Opportunity");
```

The following example is in Siebel eScript:

```
var oBO = TheApplication().GetBusObject(this.BusObject.Name);
```

The following example is in Siebel VB:

```
Dim oBO as BusObject
Dim oBC as BusComp
Set oBO = TheApplication.GetBusObject(Me.BusObject.Name)
```

GetDataSource Method for an Application

The GetDataSource method returns the name of the data source that Siebel CRM defines in the DataSource server parameter for the session. The default value is ServerDataSrc.

Format

dataSrc = *Application*.InvokeMethod("GetDataSource")

No arguments are available.

Used With

To use this method, you can use an *Application*.InvokeMethod call with the following interfaces:

- COM Data Control

- Siebel Java Data Bean

- Mobile Web Client Automation Server

- Server Script

Examples

The following Siebel eScript code detects the data source and displays the name of the data source in a dialog box:

```
var dataSrc = TheApplication().InvokeMethod("GetDataSource");
TheApplication().RaiseErrorText(dataSrc);
```

The following example is in Siebel VB:

```
Dim dataSrc As String
dataSrc = TheApplication.InvokeMethod("GetDataSource")
TheApplication.RaiseErrorText(dataSrc)
```

GetLastErrCode Method for an Application

The GetLastErrCode method returns the error code for the error that Siebel CRM logged most recently. This code is a short integer. 0 (zero) indicates no error.

Format
Application.GetLastErrCode

No arguments are available.

Usage for the GetLastErrCode Method
After you run an object interface method, you can call the GetLastErrCode method to determine if Siebel CRM returned an error from the previous operation. You can use the GetLastErrText method to return the text of the error message. Each call to a method resets the run status. For more information, see "GetLastErrText Method for an Application" on page 139.

Used With
COM Data Control, Mobile Web Client Automation Server, Web Client Automation Server

Examples
The following example is for COM Data Control:

```
errcode = SiebelApplication.GetLastErrCode
If errcode <> 0 Then
    ErrText = SiebelApplication.GetLastErrText
    MsgBox ErrText
    Exit Sub
End If
```

GetLastErrText Method for an Application
The GetLastErrText method returns a string that contains the text message for the error that Siebel CRM logged most recently.

Format
Application.GetLastErrText

No arguments are available.

Usage for the GetLastErrText Method
The text that the GetLastErrText method returns includes a Siebel error code that you can use to investigate the error. For more information, see "GetLastErrCode Method for an Application" on page 138. For more information about a specific error, see My Oracle Support.

Used With
COM Data Control, COM Data Server, Mobile Web Client Automation Server, Web Client Automation Server

Examples

The following example is for COM Data Control:

```
errcode = SiebelApplication.GetLastErrCode
If errcode <> 0 Then
    ErrText = SiebelApplication.GetLastErrText
    MsgBox ErrText
    Exit Sub
End If
```

GetProfileAttr Method for an Application

The GetProfileAttr method returns the name of an attribute in a user profile. For more information, see "SetProfileAttr Method for an Application" on page 166.

Format

Application.GetProfileAttr(*name*)

Table 33 describes the arguments for the GetProfileAttr method.

Table 33.   Arguments for the GetProfileAttr Method

Argument	Description
name	A string that indicates the name of the attribute.

Usage

For more information, see "Using System Fields with the SetProfileAttr Method" on page 167.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Browser Script:

```
var myprofile = theApplication().GetProfileAttr("Hobby");
```

The following example is in Siebel eScript:

```
var myprofile = TheApplication().GetProfileAttr("Hobby");
```

The following example is in Siebel VB:

```
Dim myprofile As String
myprofile = TheApplication.GetProfileAttr("Hobby")
```

GetService Method for an Application

The GetService method locates a business service. If this business service is not already running, then Siebel CRM starts it. This method returns the name of the business service.

Format

*Application.*GetService(*serviceName*)

Table 34 describes the arguments for the GetService method.

Table 34.   Arguments for the GetService Method

Argument	Description
serviceName	The name of the business service to start.

Usage

The GetService method searches through the predefined services that are stored in the Siebel repository file (SRF). If it does not find the business service that you specify in the serviceName argument, then it searches the business services defined in the run-time Business Services table.

Siebel CRM normally deletes a business service from memory as soon as it clears all references to this business service. The act of setting the business service to another value usually clears these references. If you set the Cache property on the business service to TRUE, then Siebel CRM keeps this business service in memory as long as the Siebel application is running.

Registering a Business Service with a Siebel Application

Starting with Siebel CRM version 8, if you use the Web Client Automation Server or Browser Script to call a business service, then you must register that business service with the Siebel application. You must do this to prevent a Service Not Found error. It is not necessary to specify this business service in the CFG file. This requirement does not apply to Server Script.

To register a business service with a Siebel application

1   In Siebel Tools, in the Object Explorer, click Application.

2   In the Applications list, locate the Siebel application you must modify.

For example, Siebel Universal Agent.

3   In the Object Explorer, expand the Application tree, and then click Application User Prop.

4 In the Application User Props list, create new application user properties using values from the following table.

Name	Value
ClientBusinessService0	XML Converter
ClientBusinessService1	My Business Service

You must enter the ClientBusinessService records sequentially, starting with ClientBusinessService0 and incrementing by 1 for each new ClientBusinessService user property you add.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

The following examples start a new instance of a business service named Workflow Process Manager.

The following example is in Browser Script:

```
function Applet_PreInvokeMethod (name, inputPropSet)
{
    if (name == "MyCustomMethod")
    {
        var oBS;
        var inpPS;
        var outPS;
        inpPS = theApplication().NewPropertySet();
        outPS = theApplication().NewPropertySet();
        oBS = theApplication().GetService("Workflow Process Manager");
        outPS = oBS.InvokeMethod("RunProcess", inpPS);
        inpPS = null;
        outPS = null;
        return ("CancelOperation");
    }
    else
    {
        return ("ContinueOperation");
    }
}
```

The following example is in Siebel eScript:

```
function WebApplet_PreInvokeMethod (MethodName)
{
    if (MethodName == "MyCustomMethod")
    {
        var oBS;
        var inpPS;
```

```
        var outPS;
        inpPS = TheApplication().NewPropertySet();
        outPS = TheApplication().NewPropertySet();
        oBS = TheApplication().GetService("Workflow Process Manager");
        oBS.InvokeMethod("RunProcess", inpPS, outPS);
        inpPS = null;
        outPS = null;
        oBS = null;
        return (CancelOperation);
    }
    else
    {
        return (ContinueOperation);
    }
}
```

The following example is in Siebel VB:

```
Function WebApplet_PreInvokeMethod (MethodName As String) As Integer
If MethodName = "MyCustomMethod" Then
    Dim oBS As Service
    Dim inpPS As PropertySet
    Dim outPS As PropertySet
    Set inpPS = TheApplication.NewPropertySet
    Set outPS = TheApplication.NewPropertySet
    Set oBS = TheApplication.GetService("Workflow Process Manager")
    oBS.InvokeMethod "RunProcess", inpPS, outPS
    Set inpPS = Nothing
    Set outPS = Nothing
    Set oBS = Nothing
    WebApplet_PreInvokeMethod = CancelOperation
Else
    WebApplet_PreInvokeMethod = ContinueOperation
End If
End Function
```

GetSharedGlobal Method for an Application

The GetSharedGlobal method returns the shared global variables. A *shared variable* is a type of variable that any script in the user session can access. It is shared among all scripts.

A shared global variable is unique to the user and the user session. A global variable for a given user is not visible to any other user. A global variable is visible only to the current user and user session. You can access the global variable from any event.

Format

Application.GetSharedGlobal(*varName*)

Table 35 describes the arguments for the GetSharedGlobal method.

Table 35.   Arguments for the GetSharedGlobal Method

Argument	Description
varName	String literal or variable that contains the name of the global variable.

Usage
Consider the following code:

```
GetSharedGlobal ("varName")
```

This code returns the string that the following code sets:

```
SetSharedGlobal  "varName",  "stringValue".
```

Used With
COM Data Control, COM Data Server, Mobile Web Client Automation Server, Server Script

Example of Using the GetSharedGlobal Method
To get the myGlobalVar global variable, the examples in this topic call the GetSharedGlobal method in the BusComp_WriteRecord event. This global variable is set through the SetSharedGlobal method in the Application_Start event. For more information, see "SetSharedGlobal Method for an Application" on page 169.

The following example is for the Component Object Model (COM):

```
Dim sReturn as String
oleVar = SiebelApplication.GetSharedGlobal("myGlobalVar", errCode)
SiebelApplication.SetSharedGlobal  "myGlobalVar", " helloworld", errCode
```

The following example is in Siebel eScript:

```
function Application_Start (CommandLine)
{
    TheApplication().SetSharedGlobal("myGlobalVar", "helloworld");
}

function BusComp_WriteRecord ()
{
    var myVar;
    myVar = TheApplication().GetSharedGlobal("myGlobalVar");
}
```

The following example is in Siebel VB:

```
Sub Application_Start (CommandLine As String)
    TheApplication.SetSharedGlobal  "myGlobalVar", "helloworld"
End Sub
```

```
Sub BusComp_WriteRecord
    Dim myVar as String
    myVar = TheApplication.GetSharedGlobal("myGlobalVar")
End Sub
```

GotoView Method for an Application

The GotoView method does the following work:

1 Deactivates any business object, business component, applet, or control that is active.

2 Activates a view.

3 Creates an instance of the business object that the view references. This business object instance becomes the active business object.

4 Activates the primary applet of the view and the business component that this applet references.

5 Activates the first tab sequence control of the primary applet.

This method does not return any information.

Format

Application.GotoView(*ViewName*[, *BusinessObjectName*])

Table 36 describes the arguments for the GotoView method.

Table 36.   Arguments for the GotoView Method

Argument	Description
ViewName	The name of the view that the Siebel application must display.
BusinessObjectName	Optional. The business object that Siebel CRM uses to display the view. You cannot specify the current active business object. If you do not provide this argument, or if you specify Nothing in this argument, then Siebel CRM activates a new business object in the normal way.

Usage

If an instance of the business object does not exist, then you must set the value for the BusinessObjectName argument to Nothing.

You cannot use the GotoView method in the following events:

■ Application_Navigate

■ Application_PreNavigate

■ Application_Start

■ Navigate

■ PreNavigate

■ WebApplet_Load

The following Siebel VB script uses GotoView to programmatically navigate to the Opportunity List view:

```
TheApplication.GotoView "Opportunity List View", Nothing
```

If your Siebel application already started an instance of an Opportunity object with the object reference of objOppty, then the following usage in Siebel VB is acceptable:

```
TheApplication.GotoView "Opportunity List View", objOppty
```

If you use the GotoView method in a Siebel VB or Siebel eScript script, then Siebel CRM runs the method last. This situation is true regardless of where you use this method in the script.

If script on a control uses the GotoView method, then do not set the Show Popup property on this control to TRUE. If you set the Show Popup to TRUE in this situation, then Siebel CRM opens the view in a new browser window. You cannot use a Multiple Document Interface (MDI) with the Siebel client, so you cannot use this configuration.

Used With
Server Script

Examples
The following examples use the GoToView method with and without the optional business object parameter.

The following example is in Siebel eScript:

```
function BusComp_WriteRecord ()
{
    var leadQuality;
    var actName;
    var actBO;
    var actBC;

    //Get the lead quality for this opportunity
    leadQuality = this.GetFieldValue("Quality");
    if(leadQuality == "1-Excellent")
    {

        //If it is a excellent lead,
        //go to the account for this opportunity
        actName = this.GetFieldValue("Account");
        actBO = TheApplication().GetBusObject("Account");
        actBC = actBO.GetBusComp("Account");

        with (actBC)
        {
            SetViewMode(AllView);
            ClearToQuery();
            SetSearchSpec("Name", actName);
            ExecuteQuery(ForwardBackward);
        }
```

```
        TheApplication().GotoView("All Account List View",actBO);

    }
    else
    {
        TheApplication().GotoView("Opportunity Detail - Activities View");
    }

    actBC = null;
    actBO = null;

}
```

The following example is in Siebel VB:

```
Sub BusComp_WriteRecord

    Dim leadQuality As String
    Dim actName As String
    Dim actBO As BusObject
    Dim actBC As BusComp

    'Get the lead quality For this opportunity
    leadQuality = Me.GetFieldValue("Quality")
    If (leadQuality = "1-Excellent") Then

        'If it is an excellent lead
        'go to the account For this opportunity
        actName = Me.GetFieldValue("Account")
        Set actBO = TheApplication.GetBusObject("Account")
        Set actBC = actBO.GetBusComp("Account")

        With actBC
            .SetViewMode AllView
            .ClearToQuery
            .SetSearchSpec "Name", actName
            .ExecuteQuery
        End With

        TheApplication.GotoView "All Account List View",actBO

    Else
        TheApplication.GotoView "Opportunity Detail - Activities View"
    End If

    Set actBC = Nothing
    Set actBO = Nothing

End Sub
```

InvokeMethod Method for an Application

The InvokeMethod method calls a method. It returns the following values:

■   In Server Script, it returns a string that contains the result of the method.

■ In Browser Script, it returns a Boolean value.

For more information, see "About Specialized and Custom Methods" on page 103.

Browser Script Format
theApplication().InvokeMethod(*methodName, methArg1, methArg2, methArgN*);

Table 37 describes the arguments for the InvokeMethod method.

Table 37. Arguments for the InvokeMethod Method

Argument	Description
methodName	The name of the method.
You can use the following arguments: ■ methArg1 ■ methArg2 ■ methArgN	One or more strings that contain arguments for the methodName argument.

Server Script Format
Application.InvokeMethod(*methodName, methArg1, methArg2, methArgN*);

The arguments you can use with this format are the same as the arguments described in Table 37 on page 148.

Usage
The InvokeMethod method allows you to call a method on an application object that is made available directly through the Siebel application interface. For more information, see "Caution About Using the InvokeMethod Method" on page 108 and "LoadObjects Method for an Application" on page 150.

Used With
Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples
For an example, see "Examples of Using the FindControl Method" on page 106.

IsViewReadOnly Method for an Application
You can use the IsViewReadOnly method to determine if a view is read-only. This method returns the following information:

■ TRUE if the view is read-only

■ FALSE if the view is not read-only

If this method does not return TRUE or FALSE, then an error occurred. If this method does not return TRUE or FALSE, then your script must provide a handler.

Format

*Application.*InvokeMethod("IsViewReadOnly",*viewName*)

Table 38 describes the arguments for the IsViewReadOnly method.

Table 38. Arguments for the IsViewReadOnly Method

Argument	Description
viewName	The name of a view. You can include the name of this view in double quotes or in a variable that contains the name of the view.

Usage

You can set a view as read-only for a particular responsibility in the Responsibility Administration view. You can use the IsViewReadOnly method to determine if a view is read-only for the current responsibility before you attempt to edit a field.

Siebel CRM does not automatically set a button to read-only when that button resides in a view that is read-only. You can use the IsViewReadOnly method to set a button to read-only in a view where IsViewReadOnly returns TRUE.

Used With

To use this method, you can use an *Application.*InvokeMethod call with the following interfaces:

■ Browser Script

■ COM Data Control

■ COM Data Server

■ Siebel Java Data Bean

■ Mobile Web Client Automation Server

■ Server Script

Examples

The following example for Siebel eScript determines if the active view is read only:

```
function ShowViewROStatus()

{

var sActive = TheApplication().ActiveViewName();

if (TheApplication().InvokeMethod("IsViewReadOnly",sActive) == "TRUE")
```

```
        TheApplication().RaiseErrorText(sActive + "is read only.");

    else

        TheApplication().RaiseErrorText(sActive + "is not read only.");

    }
```

Language Method for an Application

The Language method returns the language code of the language that the active Siebel application is running. For example, ENU.

Format

Application.InvokeMethod("Language");

No arguments are available.

Used With

To use this method, you can use an *Application*.InvokeMethod call with Server Script.

Examples

The following example uses Siebel VB:

```
    Dim curLang As String

    curLang = TheApplication.InvokeMethod("Language")
```

The following example uses Siebel eScript:

```
    var curLang;

    curLang = TheApplication().InvokeMethod("Language");
```

LoadObjects Method for an Application

The LoadObjects method starts the COM Data Server. This method must be the first call to the COM Data Server. This method returns the following information:

■ If the COM Data Server starts successfully, then the LoadObjects method returns nothing.

■ If the COM Data Server does not start successfully, then the LoadObjects method returns an error.

Format

Application.LoadObjects(*absoluteCFGfileName*)

Table 39 describes the arguments for the LoadObjects method.

Table 39.   Arguments for the LoadObjects Method

Argument	Description

absoluteCFGfileName	The path and name of the Siebel application configuration (CFG) file to open. For example:
	
	    C:\Siebel\8.1\Server\BIN\ENU
	
	As an option, to identify the data source you can append the CFG file string with the data source, separated by a comma. For example:
	
	    C:\Siebel\8.1\Server\BIN\ENU\siebel.cfg,ServerDataSrc
	
	If you do not specify the data source, then the LoadObjects method assumes the data source is local.

Usage

Prior to calling the LoadObjects method, you must modify the current folder to the Siebel\bin folder.

If you use the COM Data Server, then the COM client cannot create multiple connections to the COM Server. For example, a second attempt to call the LoadObjects method causes an error message that is similar to the following:

    The object definition manager has already been initialized.

You must restart the COM client before you can make another successful connection. Use COM Data Control instead.

Used With

COM Data Server

Examples

The following example uses COM Data Server:

```
Private Sub LoadConfig_Click()
    Dim errCode As Integer
    LoadConfig.Enabled = False
    SiebelApplication.LoadObjects "C:\Siebel\8.1\Client_2\BIN\ENU\uagent.cfg", _
        errCode

    If errCode = 0 Then
        ConfigOK = 1
    End If

    Status.Text = SiebelApplication.GetLastErrText
End Sub
```

LoadUserAttributes Method for an Application

The LoadUserAttributes method loads a user profile to the session. This method does not return any information.

Format

LoadUserAttributes(*row_id*)

Table 40 describes the arguments for the LoadUserAttributes method.

Table 40.   Arguments for the LoadUserAttributes Method

Argument	Description
row_id	The row ID of the user whose profile Siebel CRM must load.

Usage

To access the user profile, you can use the You profile from personalization rules, with the following exception: if the row ID is the row ID of the current user, then Siebel CRM loads the profile to the Me profile.

If you call this function with no argument, then it unloads the loaded user profile.

For information about user profiles, see *Siebel Personalization Administration Guide*.

Used With

Server Script

Examples

The following Siebel VB example loads a user profile to the session. The function is made available on the Siebel application object:

```
Function LoadUserProfile As Integer
TheApplication.InvokeMethod ("LoadUserAttributes","0-10N07")
End Function
```

The following Siebel VB example unloads the loaded user profile:

```
Function LoadUserProfile As Integer
TheApplication.InvokeMethod ("LoadUserAttributes", "")
End Function
```

Login Method for an Application

The Login method allows an external application to do the following:

1   Log in to the COM Data Server, COM Data Control, or Siebel Java Data Bean.

2   Access Siebel objects.

The Login method allows the end user to call the Siebel application without being prompted for a login and password. The Login method determines the privileges granted, and the role and responsibility of the end user for that session.

This method returns a string that contains the error code.

Format

Application.Login([*connectString*,] *username*, *password*)

Table 41 describes the arguments for the Login method.

Table 41.   Arguments for the Login Method

Argument	Description
connectString	Connect string that uses a token.
username	Username for the login.
password	User password for the login.

Usage

Verify that the Siebel \bin folder is the current folder. To access Data Control, you must do the following work:

■ Make sure the default Data Source references the Siebel database that you must access. For more information, see "Setting the Connect String" on page 30.

■ In the Siebel application configuration (CFG) file, make sure the EnableOLEAutomation parameter is TRUE.

Used With

COM Data Control, COM Data Server, Mobile Web Client Automation Server, Siebel Java Data Bean

Examples

The connect string for COM Data Control uses a token. For example:

    host = "Siebel://*my_computer*/SIEBEL/objsrvr/*my_computer*" lang = "ENU"

Most languages use quotes to enclose a text string, so you must use quotes in parentheses. For example:

■ To use COM Data Control in Visual Basic:

    m_dataBean.login("siebel.tcpip.none.none://gateway:gatewayport/
    enterpriseserver/SCCObjMgr", "username", "password");

■ To use COM Data Control in C++:

    Login("host=\"siebel://my_computer/SIEBEL/objsvr/my_computer\" lang =
    \"ENU\"","user","password");

The following example logs in to the Siebel Server and determines if errors exist:

```
    Call SiebelAppControl.Login("host=""siebel://gtwy/enterprise/ObjMgr""",
"SADMIN", "SADMIN")

    //Check for errors
        If SiebelAppControl.GetLastErrCode <> 0 Then
            frmMain.txtStatus.Text = SiebelAppControl.GetLasErrText
        Else
            frmMain.txtStatus.Text = "Connected successfully..."
        End If
```

The following is a Siebel Java Data Bean example that logs in to a Siebel Server and then logs off:

```
import com.siebel.data.*;
import com.siebel.data.SiebelException;

public class JDBLoginLogoffDemo
{
    private SiebelDataBean m_dataBean = null;
    public static void main(String[] args)
    {
        JDBLoginLogoffDemo demo = new JDBLoginLogoffDemo();
    }

    public JDBLoginLogoffDemo()
    {
        try
        {

            // instantiate the Siebel Java Data Bean
            m_dataBean = new SiebelDataBean();

            // login to the Siebel Servers
            m_dataBean.login("siebel.tcpip.none.none://gateway:port/enterprise/
            object manager","userid","password");
            System.out.println("Logged in to the Siebel Server ");

            //perform function code

            //release the business object

            // logoff
            m_dataBean.logoff();
            System.out.println("Logged off the Siebel Server ");
        }

        catch (SiebelException e)
        {
            System.out.println(e.getErrorMessage());
        }
    }
}
```

LoginId Method for an Application

The LoginId method returns the login ID of the user who started the Siebel application.

Format

*Application.*LoginId

No arguments are available.

Usage

The login ID is the value of the ROW_ID column in the user login record in the S_USER table. You can use the login ID as a search specification.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

In this Siebel VB example in the BusComp_PreSetFieldValue event, the LoginId method determines if the user possesses the rights to modify a record:

```
Function BusComp_PreSetFieldValue (FieldName As String,
    FieldValue As String) As Integer
    Select Case FieldName
        Case "Account Status"
            if Me.GetFieldValue("Created By") <> _
                TheApplication.LoginId then
                TheApplication.RaiseErrorText("You cannot modify Account Status " & _
                "because you did not create the record.")
            end if
    End Select
    BusComp_PreSetFieldValue = ContinueOperation
End Function
```

LoginName Method for an Application

The LoginName method returns the login name of the user who started the Siebel application. This login name is the name that the user types in the login dialog box. For more information, see "Login Method for an Application" on page 152.

Format

*Application.*LoginName

No arguments are available.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

For examples, see "ExecuteQuery Method for a Business Component" on page 197 and "TheApplication Method" on page 324.

Logoff Method for an Application

The Logoff method disconnects the Siebel client from the Siebel Server. This method does not return any information.

Format

*Application.*Logoff

No arguments are available.

Usage

For clients that include a user interface, the Logoff method removes every window except for the topmost window. Logoff also removes every object, except for the topmost object, on the Siebel client and Siebel Server.

If you remove the main object, then Siebel CRM automatically calls the Logoff method.

Used With

COM Data Control, Siebel Java Data Bean, Mobile Web Client Automation Server

LookupMessage Method for an Application

The LookupMessage method returns message text for a key. It returns this information in the current language.

Format

*Application.*LookupMessage (*category, key, [arg1], [arg2],…., [argN]*)

Table 42 describes the arguments for the LookupMessage method.

Table 42.   Arguments for the LookupMessage Method

Argument	Description
category	Name of the Message Category object that is the parent of the Key value. You can define this value in Siebel Tools.
key	Name of the Message object whose text contains the value that Siebel CRM must format. You can define this value in Siebel Tools.
Other arguments: ■ arg1 ■ arg2 ■ argN	If the error message contains a substitution argument, such as %1, then Siebel CRM uses these optional arguments to format the error message.

Usage
Useful for retrieving locale specific custom error messages.

Used With
Server Script

Examples
The following Siebel eScript example returns the following text:

> Enter Account Title before stepping off.

To test this code in the User Defined Errors message category, create a new record with the following text:

> Enter %1 before stepping off.

Siebel CRM uses the Account Title parameter to substitute the %1 variable:

```
var sVal = TheApplication().LookupMessage("User Defined Errors", "Test", "Account
Title");
```

LookupValue Method for an Application
If all of the following items are true, then the LookupValue method locates a row in the
S_LST_OF_VAL table:

■ The value in the TYPE column matches the value in the type argument.

■ The value in the CODE column matches the value in the lang_ind_code argument.

■ The value in the LANG_ID column matches the language code of the currently active language.

You can use this method to get the translation of the untranslated value in the LOV to the language that is currently active.

The LookupValue method returns a string that contains the display value from the VAL column for the row. If it does not find the display value, then it returns the language independent code as the value.

Format

val = *Application.*InvokeMethod("LookupValue", *type*, *lang_ind_cd*)

Table 43 describes the arguments for the LookupValue method.

Table 43.   Arguments for the LookupValue Method

Argument	Description
type	The type that is specified in the List of Values administration view.
lang_ind_cd	Value for the language independent code that is specified in the List of Values administration view.

Used With

To use the LookupValue method, you can use an *Application.*InvokeMethod call with the following interfaces:

- COM Data Control
- Siebel Java Data Bean
- Mobile Web Client Automation Server
- Server Script

Examples

The following example is in Siebel eScript:

```
var LOVText = TheApplication().InvokeMethod("LookupValue", "SR_AREA", "Network");
```

Name Method for an Application

The Name method returns the name of the Siebel application.

Format

*Application.*Name

No arguments are available.

Used With

Browser Script, Web Client Automation Server

NewPropertySet Method for an Application

The NewPropertySet method creates a new property set. It returns a property set.

Format

*Application.*NewPropertySet

No arguments are available.

Usage

You can use the NewPropertySet method to create input and output arguments for a business service.

If you use the NewPropertySet method on an existing PropertySet object, then old references to this PropertySet are lost. If you reuse a PropertySet, then use the Reset method on this PropertySet.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

This example creates a new property set. It uses Browser Script:

```
function Applet_PreInvokeMethod (name, inputPropSet)
{
   if (name == "MyCustomMethod")
   {
      var oBS;
      var inpPS;
      var outPS;
      inpPS = theApplication().NewPropertySet();
      outPS = theApplication().NewPropertySet();
      oBS = theApplication().GetService("New Value Business Service");
      outPS = oBS.InvokeMethod("New Value Method", inpPS);
      inpPS = null;
      outPS = null;
      oBS = null;
      return ("CancelOperation");
   }

   else
   {
      return ("ContinueOperation");
   }
}
```

The following example is for the Component Object Model (COM):

```
Dim oBS As SiebelService
Dim inpPS As SiebelPropertySet
Dim outPS As SiebelPropertySet
Dim errCode as integer

Set inpPS = SiebelApplication.NewPropertySet(errCode)
Set outPS = SiebelApplication.NewPropertySet(errCode)
Set oBS = SiebelApplication.GetService("New Value Business Service", errCode)
oBS.InvokeMethod "New Value Method", inpPS, outPS, errCode
Set inpPS = Nothing
Set outPS = Nothing
Set oBS = Nothing
```

The following example is in Siebel eScript:

```
function WebApplet_PreInvokeMethod (MethodName)
{
    if (MethodName == "MyCustomMethod")
    {
        var oBS;
        var inpPS;
        var outPS;
        inpPS = TheApplication().NewPropertySet();
        outPS = TheApplication().NewPropertySet();
        oBS = TheApplication().GetService("New Value Business Service");
        oBS.InvokeMethod("New Value Method", inpPS, outPS);
        inpPS = null;
        outPS = null;
        oBS = null;
        return (CancelOperation);
    }

    else
    {
        return (ContinueOperation);
    }

}
```

The following example is in Siebel VB:

```
Function WebApplet_PreInvokeMethod (MethodName As String) As Integer
    If MethodName = "MyCustomMethod" Then
        Dim oBS As Service
        Dim inpPS As PropertySet
        Dim outPS As PropertySet
        Set inpPS = TheApplication.NewPropertySet
        Set outPS = TheApplication.NewPropertySet
        Set oBS = TheApplication.GetService("New Value Business Service")
        oBS.InvokeMethod "New Value Method", inpPS, outPS
        Set inpPS = Nothing
        Set outPS = Nothing
        Set oBS = Nothing
        WebApplet_PreInvokeMethod = CancelOperation
```

```
        Else
            WebApplet_PreInvokeMethod = ContinueOperation
        End If

    End Function
```

PositionId Method for an Application

The PositionId method returns the position ID of the user position. This position ID is the ROW_ID from the S_POSTN table. Siebel CRM sets this value by default when the Siebel application starts. To modify this value, the user can use the Edit menu, and then the Change Position menu item.

Format

Application.PositionId

No arguments are available.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

PositionName Method for an Application

The PositionName method returns the name of the current user position. Siebel CRM sets this value by default when it starts the Siebel application.

Format

Application.PositionName

No arguments are available.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following Siebel VB example determines the position of a user who is attempting to modify the sales stage. If the position does not allow this modification, then this code prevents the modification:

```
    Function BusComp_PreSetFieldValue (FieldName As String, FieldValue As String) As
    Integer

    Dim sPosName As String sMsgText As String
    Select Case FieldName
        Case "Sales Stage"
            If FieldValue = "Approved" Then
                ' Do not allow the sales cycle to be modified to
```

```
            ' this value if the User is not a manager or VP.
            sPosName = TheApplication.PositionName
            If NOT ((sPosName="Manager") OR (sPosName="VP"))Then
                TheApplication.RaiseErrorText("Only a Manager or Vice President can
                approve a Pipeline Item. Please notify your Manager that you _
                want to have this Pipeline Item approved.")
            End If
      BusComp_PreSetFieldValue = ContinueOperation
   End Select

   End Function
```

RaiseError Method for an Application

The RaiseError method sends a scripting error message to the browser. The error code is a standard number.

To determine the error text, Siebel CRM uses the key to look up the current language from the User-Defined Errors category. To define these errors in Siebel Tools, you can use the Message Category object. You can use the optional arguments to format the string if it contains a substitution argument, such as %1 or %2. This method does not return any information.

Format

*Application.*RaiseError(*key*, [*arg1*], [*arg2*],…., [*argN*])

The arguments you can use in this format are the same as the arguments that are described in Table 42 on page 157 except the RaiseError Method does not include a category argument.

Usage for the RaiseError Method

The RaiseError method causes Siebel CRM to terminate the script and send a notification to the browser. Therefore, you are not required to use CancelOperation after you use the RaiseError method. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

The RaiseError method and the RaiseErrorText method create a Server Script exception. If you use error handling in your script, then the error handling can suppress RaiseError and RaiseErrorText functionality.

If you use On Error Goto error handling in Siebel VB, and if you use the RaiseError method or the result from the RaiseErrorText method, then Siebel CRM transfers the script run to the error handler. If you use On Error Resume Next error handling, then Siebel CRM suppresses the RaiseError method and the RaiseErrorText method.

CAUTION: Be careful if you use RaiseError because it cancels operations. For example, if you use it in the BusComp_PreWriteRecord event, then the user or code cannot step off the current record until Siebel CRM addresses the condition that causes the call to the RaiseError method.

Used With

Server Script

Examples

In the following Siebel eScript example, the RaiseError method results in Siebel CRM raising a scripting exception and transferring control to the Catch statement. To display the error message, you must use the Throw statement:

```
function BusComp_PreDeleteRecord ()
{
    try {
        var status = this.GetFieldValue("Account Status");

        if (status == "Gold") {
            TheApplication().RaiseError(user defined error name);
        }
        else {
            return (ContinueOperation);
        }
    }
    catch (e) {
        throw e;
    }

}
```

In the following Siebel eScript example, if the user deletes an opportunity that includes the Pipeline revenue class, then Siebel CRM sends an error message:

```
function BusComp_PreDeleteRecord ()
{
    try
    {
        var revClass = this.GetFieldValue("Primary Revenue Class");
        if (revClass == "1-Pipeline")
        {
            TheApplication().RaiseError("user-defined test error1", "PreDelete",
            "RaiseError Method" );
        }

        else
        {
        return (ContinueOperation);
        }

    }
    catch (e)
    {
        throw e;
    }

}
```

Siebel CRM sends the following error message:

This user-defined test error is used in PreDelete, as an example for RaiseError Method

Note the following key:

> user-defined test error1

This key is predefined as the following:

> This user-defined test error is used in %1, as an example for %2.

When the script runs, Siebel CRM does the following:

■ Substitutes PreDelete for %1

■ Substitutes Raise Error Method for %2

RaiseErrorText Method for an Application

The RaiseErrorText method sends a scripting error message to the browser. This method does not return any information.

Format

Application.RaiseErrorText(*value*, [*arg1*], [*arg2*],…., [*argN*])

Table 44 describes the arguments for the RaiseErrorText method.

Table 44.   Arguments for the RaiseErrorText Method

Argument	Description
value	The error text message.
Other arguments: ■ arg1 ■ arg2 ■ argN	If the error message contains a substitution argument, such as %1, then Siebel CRM uses these optional arguments to format the error message.

Usage

Usage for the RaiseErrorText method is very similar to usage for the RaiseError method. For more information, see "Usage for the RaiseError Method" on page 162.

Used With

Server Script

Examples

In the following Siebel eScript example, the RaiseErrorText method causes Siebel CRM to raise a scripting exception and then transfer control to the Catch statement. To display the error message, you must use the Throw statement.

```
function BusComp_PreDeleteRecord ()
{
    try {
        var status = this.GetFieldValue("Account Status");

        if (status == "Gold") {
            TheApplication().RaiseErrorText("Unable to delete Gold Account");
        }
        else {
            return (ContinueOperation);
        }
    }
    catch (e) {
        throw e;
    }

}
```

In the following Siebel eScript example, if the user deletes an opportunity that includes Pipeline as the revenue class, then Siebel CRM sends an error:

```
function BusComp_PreDeleteRecord ()
{
    try
    {
        var revClass = this.GetFieldValue("Primary Revenue Class");
        if (revClass == "1-Pipeline")
        {
            TheApplication().RaiseErrorText("Exception occurred in %1. Unable to
            delete Opportunity with %2 revenue class.", "PreDeleteRecord", revClass);
        }
        else
        {
            return (ContinueOperation);
        }

    }
    catch (e)
    {
        throw e;
    }
}
```

SetPositionId Method for an Application

The SetPositionId method sets the active position to a Position Id. This method returns a Boolean value that indicates if Siebel CRM successfully completed the operation.

Format

Application.SetPositionId(*positionId*)

Table 45 describes the arguments for the SetPositionId method.

Table 45.   Arguments for the SetPositionId Method

Argument	Description
positionId	A string that contains the Position Id.

Usage

The positionId argument must contain the Position Id that is associated with the user who is currently logged in to the Siebel application.

Used With

COM Data Server, COM Data Control, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

SetPositionName Method for an Application

The SetPositionName method sets the active position to a position name. The method returns a Boolean value that indicates if the method succeeded.

Format

*Application.*SetPositionName*(positionName)*

Table 46 describes the arguments for the SetPositionName method.

Table 46.   Arguments for the SetPositionName Method

Argument	Description
positionName	A string that contains the name of the position.

Usage

The positionName argument must contain the Position name that is associated with the user who is currently logged in to the Siebel application.

Used With

COM Data Server, COM Data Control, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

SetProfileAttr Method for an Application

Personalization uses the SetProfileAttr method to set a value for an attribute in a user profile. This method does not return any information.

Format

Application.SetProfileAttr(*name*, *value*)

Table 47 describes the arguments for the SetProfileAttr method.

Table 47.   Arguments for the SetProfileAttr Method

Argument	Description
name	A string that contains the name of the attribute.
value	The value of the attribute.

Usage

The SetProfileAttr method sets the value of the value argument to an attribute in the user profile that the name argument contains. Siebel CRM does the following work:

■ If this attribute already exists, then Siebel CRM updates the corresponding persistent profile attribute in the Siebel application. This value is defined in the Personalization Profile business component.

■ If the profile attribute does not exist in the list of persistent profile attributes, then Siebel CRM creates it as a dynamic profile attribute. It does not include quotation marks at the beginning or end of the name.

■ If you use the SetProfileAttr method in Browser Script, then Siebel CRM performs a round trip to the Siebel Server and back to the browser each time it uses this method. This processing creates a performance overhead.

For more information about user profile attributes, see *Siebel Applications Administration Guide*.

Using System Fields with the SetProfileAttr Method

You cannot use the SetProfileAttr method with a system field. These fields are not explicitly defined in the Personalization Profile business component. You cannot use the SetProfileAttr method with the Id field because attempting to modify the ROW_ID column of a table creates an error. For more information about system fields, see *Configuring Siebel Business Applications*.

Personalization uses the GetProfileAttr method. Siebel CRM does not explicitly define system fields in the Personalization Profile business component, so you cannot use this method with a system field, except for the Id field. For more information, see *Siebel Personalization Administration Guide*.

Used With

Browser Script, COM Data Control, COM Data Server, Server Script, Siebel Java Data Bean, Mobile Web Client Automation Server

Examples

The following example is in Browser Script:

```
function Applet_PreInvokeMethod (name, inputPropSet)
```

```
{
    if (name == "hobbyReq") {
        var hobby = theApplication().GetProfileAttr("Hobby");


        if (hobby == "") {
            hobby = prompt("Please enter your favorite hobby");
            theApplication().SetProfileAttr("Hobby", hobby);
        }
        return ("CancelOperation");
    }
    else
        return ("ContinueOperation");
}
```

This following examples exchange information between an applet Server Script and an applet Browser Script:

■ In the applet Server Script, Siebel CRM uses the SetProfileAttr method to set a customer profile attribute named MyProAttr to Hello World.

■ In the applet Browser Script, you can use the GetProfileAttr method to return the profile attribute.

The following example is in Siebel eScript:

```
function WebApplet_PreInvokeMethod (MethodName)
{

    if (MethodName == "MyCustomMethod") {

        TheApplication().SetProfileAttr("MyProAttr", "Hello World Siebel eScript");
        return (CancelOperation);

    }
    return (ContinueOperation);

}
```

The following example is in Siebel VB:

```
Function WebApplet_PreInvokeMethod (MethodName As String) As Integer

If MethodName = "MyCustomMethod" Then

    TheApplication.SetProfileAttr "MyProAttr", "Hello World VB"
    WebApplet_PreInvokeMethod = CancelOperation
Else
    WebApplet_PreInvokeMethod = ContinueOperation
End If

End Function
```

SetSharedGlobal Method for an Application

The SetSharedGlobal method sets a shared global variable that your code can access with the GetSharedGlobal method. The SetSharedGlobal method does not return any information.

Format

*Application.*SetSharedGlobal(*varName*, *value*)

Table 48 describes the arguments for the SetSharedGlobal method.

Table 48.   Arguments for the SetSharedGlobal Method

Argument	Description
varName	String variable or literal that contains the name of the shared global variable that Siebel CRM must set.
value	String variable or literal that contains the value of the shared global variable.

Used With

COM Data Control, COM Data Server, Mobile Web Client Automation Server, Server Script

Examples

The following example is for the Component Object Model (COM):

```
comVar = SiebelApplication.GetSharedGlobal("myVar", errCode)
SiebelApplication.SetSharedGlobal  "myVar", "BLAH", errCode
```

The following example is in Siebel VB:

```
TheApplication.SetSharedGlobal  "myVar", "FOO"
myVar = TheApplication.GetSharedGlobal("myVar")
```

The remaining examples for using the SetSharedGlobal method are the same as the examples for using the GetSharedGlobal method. For more information, see "Example of Using the GetSharedGlobal Method" on page 144.

ShowModalDialog Method for an Application

The ShowModalDialog method allows you to display a dialog box with the cursor in the default state. This application object method calls the equivalent object method in Microsoft Windows. This method returns the value of the returnValue property. The window of the document specified in the url argument sets this property.

Format

theApplication().ShowModalDialog (*url[, argin][, options]*)

Table 49 describes the arguments for the ShowModalDialog method.

Table 49.   Arguments for the ShowModalDialog Method

Argument	Description
url	The URL of the document that Siebel CRM finished loading and displaying.
argin	Passes arguments to use if Siebel CRM displays the document. This argument can be a value of any type, including an array of values. For more information, see the window.dialogArguments property of the object in the Document Object Model. For example: ■ See the window.showModalDialog property at http://developer.mozilla.org. ■ See the showModalDialog method at http://msdn.microsoft.com.
options	String that specifies the attributes for the dialog box. For more information, see "Values for the Options Argument" on page 170.

Values for the Options Argument

Table 50 describes values you can use for the options argument of the ShowModalDialog method. You must use a semicolon to separate these values.

Table 50.   Values for the Options Argument of the ShowModalDialog Method

Option	Description
dialogHeight	Sets the height of the dialog box. You must use an integer or floating-point number followed by one of the following items: ■ **An absolute units designator.** For example, cm, mm, in, pt, pc, or px. ■ **A relative units designator.** For em or ex. The default value is em. For consistent results, specify the dialogHeight and dialogWidth in pixels. The minimum height is 100 pixels.
dialogLeft	Sets the left position of the dialog box relative to the upper-left corner of the desktop.
dialogTop	Sets the top position of the dialog box relative to the upper-left corner of the desktop.
dialogWidth	Sets the width of the dialog box.

Table 50.   Values for the Options Argument of the ShowModalDialog Method

Option	Description
center	Sets centering for the dialog box. You can use one of the following values: ■ yes ■ no ■ 1 ■ 0 ■ on ■ off The default value is yes.
dialogHide	Specifies how to hide the dialog box if the user prints or uses print preview. This option is available only if the user opens the dialog box from a trusted application. You can use the same values that you use with the center option. The default value is no.
edge	Specifies the edge style of the dialog box. You can use one of the following values: ■ sunken ■ raised The default value is raised.
help	Specifies how to display the dialog box with the context-sensitive Help icon. You can use the same values that you use with the center option. The default value is yes.
resizable	Specifies if the dialog box dimensions are fixed. You can use the same values that you use with the center option. The default value is no.
scroll	Specifies if the dialog box displays scrollbars. You can use the same values that you use with the center option. The default value is yes.

Table 50.   Values for the Options Argument of the ShowModalDialog Method

Option	Description
status	Specifies how the dialog box displays a status bar.
	You can use the same values that you use with the center option. The default value is one of the following:
	■ yes for an untrusted dialog box
	■ no for a trusted dialog box
unadorned	Specifies how the dialog box displays the border window chrome. This feature is available only if the user opens the dialog box from a trusted application. A *trusted application* is an application that includes a trust certificate.
	You can use the same values that you use with the center option. The default value is no.

Used With

Browser Script

Examples

This example uses Browser Script to display a dialog box that includes a URL:

```
function Applet_Load ()
{
   var sOptions="dialogHeight:  1000px;edge:sunken;resizable;yes";
   theApplication().ShowModalDialog("http://www.yahoo.com",  "",  sOptions)
}
```

SWEAlert Method for an Application

The SWEAlert method displays a modal dialog box that includes a message. This method does not return any information.

Format

theApplication().SWEAlert(message)

Usage

Use the SWEAlert method instead of alert. Note the following:

■   If you use the SWEAlert method, then Siebel CRM does not send the parent applet to the background.

■   If you use alert, then Siebel CRM sends pop-up applets to the background. MVGs and pick applets are examples of pop-up applets. If a browser event sends a JavaScript alert, then Siebel CRM hides the pop-up applet.

Used With

Browser Script

Examples

The following Browser Script example displays a status message:

```
function BusComp_PreSetFieldValue (fieldName, value) {

    if (fieldName == "Account Status") {
        var cVolume = this.GetFieldValue("Current Volume");
        if ((value == "Inactive") && (cVolume > 0)) {
            theApplication().SWEAlert("Unable to inactivate an account that has a
            current volume greater than 0");

            return ("CancelOperation");
        }
        else
            return ("ContinueOperation");
    }
    else
        return ("ContinueOperation");
}
```

Trace Method for an Application

The Trace method appends a message to the trace file. Trace helps to debug an SQL query and to monitor how Siebel CRM allocates objects. This method does not return any information.

This tracing is not the same as the tracing that you can activate in the Siebel application configuration (CFG) file. For more information, see "Tracing a Script" on page 81.

It is recommended that you do not use the Trace method or the TraceOn method in a production environment. For more information, see "TraceOn Method for an Application" on page 176.

Format

Application.Trace(*message*)

Table 51 describes the arguments for the Trace method.

Table 51.   Arguments for the Trace Method

Argument	Description
message	String variable or literal that contains message text that Siebel CRM appends to the trace file.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is for COM Data Server:

```
Private Sub TraceOn_Click()
    Dim ErrCode As Integer
    SiebelApplication.TraceOn "c:\temp\trace.txt", "allocation", _
        "all",        ErrCode
    If (ErrCode = 0) Then SiebelApplication.TraceOn
        "c:\temp\trace.txt",       "SQL", "",ErrCode
    If (ErrCode = 0) Then SiebelApplication.Trace
        "Start of Tracing!",
        ErrCode
End Sub
```

The following example is in Siebel VB:

```
Sub Button2_Click
    TheApplication.TraceOn "C:\temp\trace.txt", "allocation",    "all"
    TheApplication.TraceOn "C:\temp\trace.txt", "sql", ""
    TheApplication.Trace "start of tracing!"
End Sub
```

Example Trace Output

The following is example output of an Allocation trace section:

```
03/05/98, 17: 27: 47, START, 4. 0. 4 [1425_P3] ENU
03/05/98, 17: 27: 47, ALLOC, 1, BusObject, Account, Basic
03/05/98, 17: 27: 48, ALLOC, 2, BusComp, Account, Basic
03/05/98, 17: 27: 48, RELEASE, 1
03/05/98, 17: 27: 48, RELEASE, 2
```

The following is example output of an SQL trace section:

```
01/22/98, 21: 03: 49, START, 4. 0. 2 [1416] ENU
01/22/98, 21: 04: 02, COMMENT, Start of Tracing!
01/22/98, 21: 04: 10, SQLSTMT, 1, SELECT, "SELECT
    T1. ROW_ID,
    T1. MODIFICATION_NUM,
    T1. CREATED_BY,
    T1. LAST_UPD_BY,
    T1. CREATED,
    T1. LAST_UPD,
    T1. CONFLICT_ID,
    T1. NAME,
    T1. DESC_TEXT,
    T1. PRIV_FLG,
    T1. QUERY_STRING
FROM
    DEV32. S_APP_QUERY T1
WHERE
    (T1. CREATED_BY = :1 OR T1. PRIV_FLG = :2) AND
    ((T1. NAME LIKE :3 OR T1. NAME LIKE :4 OR T1. NAME LIKE :5 OR
        T1. NAME LIKE :6) AND UPPER(T1. NAME) = UPPER(:7))
    ORDER BY
```

```
        T1.NAME, T1.DESC_TEXT"
01/22/98,21:04:10,SQLBIND,1,1,1-6NF
01/22/98,21:04:10,SQLBIND,1,2,N
01/22/98,21:04:10,SQLBIND,1,3,ac%
01/22/98,21:04:10,SQLBIND,1,4,Ac%
01/22/98,21:04:10,SQLBIND,1,5,aC%
01/22/98,21:04:10,SQLBIND,1,6,AC%
01/22/98,21:04:10,SQLBIND,1,7,Account
```

Related Topics

For more information, see the following topics:

-

-

TraceOff Method for an Application

The TraceOff method turns off tracing that the TraceOn method starts. This method does not return any information.

Format

Application.TraceOff

No arguments are available.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

This following example in Siebel VB sets the value in the Sales Stage field to the first value in the drop-down list for the field. It uses tracing to track the result:

```
Sub BusComp_NewRecord
    TheApplication.TraceOn "C:\lvpick.doc", "SQL", ""
    Dim oBC as BusComp
    set oBC = me.GetPickListBusComp("Sales Stage")

    With oBC
        .SetViewMode AllView
        .ActivateField "Sales Stage Order"
        .ClearToQuery
        .SetSortSpec "Sales Stage Order"
        .ExecuteQuery ForwardOnly
        if .FirstRecord then
            .Pick
        end if
    End With
```

```
        set oBC = Nothing

        TheApplication.TraceOff

    End Sub
```

TraceOn Method for an Application

The TraceOn method turns on tracing for allocations and deallocations of Siebel objects and SQL statements that Siebel CRM creates. This method does not return any information.

Format

*Application.*TraceOn(*filename*, *type*, *selection*)

Table 52 describes the arguments for the TraceOn method.

Table 52.   Arguments for the TraceOn Method

Argument	Description
filename	Output filename for trace messages. If you do not use this argument, then Siebel CRM logs tracing information to the Object Manager log file. For more information, see "Filename Argument of the TraceOn Method" on page 176.
type	The type of tracing to start. You can use the following values: ■ **Allocation.** Traces allocations and deallocations of Siebel objects. This feature is useful if you suspect a memory leak exists in your code. ■ **SQL.** Traces SQL statements that the Siebel application creates.
selection	Identifies the Siebel objects that Siebel CRM must trace for the Allocation trace type. This argument is "" if the trace type is SQL: ■ **Script.** Traces Siebel VB and Siebel eScript objects. ■ **OLE.** Traces allocations for data server or automation server programs. ■ **All.** Traces all objects that Siebel CRM creates as a result of scripting. This value does not trace Siebel objects that are defined through Siebel Tools.

Filename Argument of the TraceOn Method

You can use the following values for the filename argument:

■ $p. Substitutes the process Id for the filename.

■ $t. Substitutes the thread Id for the file name.

For example:

```
    TheApplication().TraceOn("C:\\temp\\trace_$p_$t.txt", "Allocation", "All");
```

This code causes Siebel CRM to log trace files to the trace_1496_1412.txt file in the C:\temp\trace folder.

To make sure the filename argument is unique, you must place a separator between the $p and $t values. For example, assume you do not use a separator and the following items are true:

■ The process id for user A is 1 and the thread id is 12.

■ The process id for user B is 11 and the thread id is 2.

In this situation, the file name is trace_112.txt for user A and for user B, so Siebel CRM logs trace information for each user to the same file.

If you add a separator between the process id and the thread id, then the file names are unique and Siebel CRM logs trace information to a separate file for each user. For example:

■ trace_1_12.txt

■ trace_11_2.txt

Usage

To turn off tracing, you must call the TraceOff method. If you attempt to call the TraceOn method with a different filename without first calling TraceOff, then Siebel CRM writes trace information to the new trace file name. The old file remains open and is locked. You can issue multiple TraceOn statements to the same trace file.

It is recommended that you do not use the Trace method or the TraceOn method in a production environment. For more information, see "Trace Method for an Application" on page 173.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is for COM Data Server:

```
Private Sub TraceOn_Click()
    Dim ErrCode As Integer
    SiebelApplication.TraceOn "c:\temp\trace.txt", "allocation",
        "all",       ErrCode
    If (ErrCode = 0) Then SiebelApplication.TraceOn
        "c:\temp\trace.txt",      "SQL", "",ErrCode
    If (ErrCode = 0) Then SiebelApplication.Trace
        "Start of Tracing!",
        ErrCode
End Sub
```

The following example is in Siebel eScript:

```
function BusComp_PreSetFieldValue (FieldName, FieldValue)

{
TheApplication().TraceOn("C:\\temp\\trace.txt", "Allocation", "All");
TheApplication().TraceOn("C:\\temp\\trace.txt", "SQL", "");
TheApplication().Trace("start tracing!");
```

```
    return (ContinueOperation);
    }
```

The following example is in Siebel VB:

```
    Sub Button2_Click
        TheApplication.TraceOn "C:\temp\trace.txt", "allocation",
            "all"
        TheApplication.TraceOn "C:\temp\trace.txt", "sql", ""
        TheApplication.Trace "start of tracing!"
    End Sub
```

For example trace output, see .

The following examples use Trace, Traceoff, and TraceOn methods to create a trace file with SQL statements issues by the scripting query.

The following example is in Siebel eScript:

```
    function BusComp_NewRecord ()
    {
        TheApplication().TraceOn("C:\\trace_output.txt", "SQL", "");
        TheApplication().Trace("Start of tracing!");
        var oBC = this.GetPickListBusComp("Sales Stage");

        with (oBC)
        {
            SetViewMode(AllView);
            ClearToQuery();
            SetSortSpec("Sales Stage Order(ASCENDING)");
            ExecuteQuery(ForwardOnly);
            if (FirstRecord())
            {
                Pick();
            }
        }

        oBC = null;
        TheApplication().Trace("End of tracing!");
        TheApplication().TraceOff();
    }
```

The following example is in Siebel VB:

```
    Sub BusComp_NewRecord

        TheApplication.TraceOn "C:\trace_output.txt", "SQL", ""
        TheApplication.Trace "Start of tracing!"
        Dim oBC as BusComp
        Set oBC = Me.GetPickListBusComp("Sales Stage")

        With oBC
            .SetViewMode AllView
            .ClearToQuery
            .SetSortSpec "Sales Stage Order(ASCENDING)"
            .ExecuteQuery ForwardOnly
```

```
        If .FirstRecord Then
            .Pick
        End If
    End With

    Set oBC = Nothing
    TheApplication.Trace "End of tracing!"
    TheApplication.TraceOff
End Sub
```

Related Topics

For more information, see the following topics:

- ■ "Trace Method for an Application" on page 173

- ■ "TraceOff Method for an Application" on page 175

Application Events

This topic describes application events. It includes the following topics:

- ■ "Application_Close Event" on page 179

- ■ "Application_InvokeMethod Event" on page 180

- ■ "Application_Navigate Event" on page 180

- ■ "Application_PreInvokeMethod Event" on page 181

- ■ "Application_PreNavigate Event" on page 182

- ■ "Application_Start Event" on page 183

You can use these events only on the Siebel Server, except for the following events that you can use on the Siebel Server or on the browser:

- ■ Application_InvokeMethod Event

- ■ Application_PreInvokeMethod Event

Application_Close Event

You can call the Application_Close event before the Siebel application exits. This technique allows scripts to perform cleanup, such as closing a connection to a COM server. Note the following:

- ■ If Windows notifies the Siebel application that it must close, then Siebel CRM calls this event.

- ■ If the process is terminated directly, then Siebel CRM does not call this event. For example, a direct termination occurs if the user clicks the close (X) icon at the top right of a window.

This event does not return any information.

Format
Application_Close

No arguments are available.

Used With
Server Script

Siebel Business Processes call this event. For more information, see *Siebel Business Process Framework: Workflow Guide*.

Application_InvokeMethod Event
Siebel CRM calls the Application_InvokeMethod event after a specialized method is called. This method returns TRUE if the call succeeds or FALSE if the call does not succeed. For more information, see "About Specialized and Custom Methods" on page 103.

Browser Script Format
Application_InvokeMethod(*name, inputPropSet*)

The arguments you use with this format are the same as the arguments described in Table 24 on page 113.

This method sends the values you enter in the inputPropSet argument to the InvokeMethod event.

Server Script Format
Application_InvokeMethod(*methodName*)

The arguments you use with this format are the same as the arguments described in Table 24 on page 113 except there is no inputPropSet argument.

Used With
Browser Script, Server Script

Related Topics
For more information, see the following topics:

■ "Customizing the Outcome of an Object Interface Event" on page 58

■ "Application_PreInvokeMethod Event" on page 181

Application_Navigate Event
Siebel CRM calls the Application_Navigate event after the user navigates to a view. This event does not return any information.

Format
Application_Navigate

No arguments are available.

Used With
Server Script

Application_PreInvokeMethod Event

Siebel CRM calls the Application_PreInvokeMethod event before one of the following items calls a specialized method:

■ A custom applet menu that you define

■ The InvokeMethod method

This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

For more information about this method, see "About Specialized and Custom Methods" on page 103 and "Customizing the Outcome of an Object Interface Event" on page 58.

Browser Script Format
Application_PreInvokeMethod (*methodName*, *inputPropSet*)

The arguments you use with this format are the same as the arguments described in Table 24 on page 113.

Server Script Format
Application_PreInvokeMethod(*methodName*)

The arguments you use with this format are the same as the arguments described in Table 24 on page 113, except there is no inputPropSet argument.

Usage
If the method you instruct Siebel CRM to call is part of an If statement, then you must set the return value for the PreInvokeMethod before the End If statement. The following code is an example of this usage:

```
If MethodName = "ResetQuery" then
    Application_PreInvokeMethod = CancelOperation
End If
```

Used With
Browser Script, Server Script

Examples

The following example is in Siebel VB:

```
Function Application_PreInvokeMethod (MethodName _
        As String) As Integer

    Dim i As Integer
    Dim iReturn As Integer
    iReturn = ContinueOperation

    Select Case MethodName
        Case "LaunchWord"
            i = Shell("C:\Program Files\Microsoft Office\Office\WINWORD.EXE",1)
            iReturn = CancelOperation

        Case "LaunchExcel"
            i = Shell("C:\Program Files\Microsoft Office\Office\EXCEL.EXE",1)
            iReturn = CancelOperation
    End Select

    Application_PreInvokeMethod = iReturn

End Function
```

The following is the equivalent example in Siebel eScript. Note that for this script to run, the entire Clib.system statement must reside on a single line in the editor:

```
function Application_PreInvokeMethod (MethodName)

    var iReturn = ContinueOperation;

    switch (MethodName)
    {
      case "LaunchWord":
        Clib.system("C:\\Program Files\\Microsoft Office\\Office\\WINWORD.EXE",1);
        iReturn = CancelOperation;
         break;

      case "LaunchExcel":
        Clib.system("C:\\Program Files\\Microsoft Office\\Office\\EXCEL.EXE",1);
        iReturn = CancelOperation;
    }

    return (iReturn);
}
```

Application_PreNavigate Event

Siebel CRM calls the Application_PreNavigate event before it displays the view where the user navigates. This event returns CancelOperation or ContinueOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format

Application_PreNavigate(*DestViewName*, *DestBusObjName*)

Table 53 describes the arguments for the Application_PreNavigate event.

Table 53.   Arguments for the Application_PreNavigate Event

Argument	Description
DestViewName	Name of the view where the user navigates.
DestBusObjName	Business object that the destination view references.

Used With

Server Script

Examples

In the following Siebel eScript example, the script Identifies the current business object and sets the current contact Id as a global variable. You can use this variable to keep context:

```
function Application_PreNavigate (DestViewName, DestBusObjName)
{
   try
   {
      var currentView = this.ActiveViewName();
      var BO = this.ActiveBusObject();
      if(BO.Name() == "Contact")
      {
         var BC = BO.GetBusComp("Contact");
         var id = BC.GetFieldValue("Id");
         TheApplication().SetSharedGlobal("ContactId", id);
      }
   }
   catch (e)
   {
      this.Trace("Exception caught: "+e.toString());
   }
   return (ContinueOperation);
}
```

Application_Start Event

Siebel CRM calls the Application_Start event when the Siebel client starts and again when it displays the client interface for the first time. This event does not return any information.

CAUTION: Do not use the RaiseErrorText method in the Application_Start event. The RaiseErrorText method does not work in the Application_Start event, and can cause the Application Object Manager to abort.

Format

Application_Start(*commandline*)

describes the arguments for the Application_Start event.

Table 54.   Arguments for the Application_Start Event

Argument	Description
commandline	Text of the command line that starts the Siebel application.

Siebel Business Processes call this event. For more information, see *Siebel Business Process Framework: Workflow Guide*.

Used With

Server Script

Examples

This example Siebel VB code returns the first and last name of the user who logs in to the Siebel application:

```
Sub Application_Start(CommandLine As String)
Dim oEmpBusObj as BusObject
Dim oEmpBusComp as BusComp
Dim oEmpBusComp as BusComp
Dim sLoginName as String
Dim sUserName as String

sLoginName = TheApplication.LoginName
Set oEmpBusObj = TheApplication.GetBusObject("Employee")
Set oEmpBusComp = oEmpBusObj.GetBusComp("Employee")
With oEmpBusComp
   .ActivateField "First Name"
   .ActivateField "Last Name"
   .ClearToQuery
   .SetSearchSpec "Login Name", sLoginName
   .ExecuteQuery
   If (.FirstRecord = 1) Then
      sUserName = .GetFieldValue("First Name")
      sUserName = sUserName + " " + .GetFieldValue("Last Name")
   End If
End With

Set oEmpBusComp = Nothing
Set oEmpBusObj = Nothing
End Sub
```

Business Component Methods

This topic describes business component methods. It includes the following topics:

The oBusComp and BusComp variables that this topic describes refer to an instance of a business component.

ActivateField Method for a Business Component

The ActivateField method activates a field. This method does not return any information. You must use the ActivateField method to activate a field before you can perform a query for the business component. For more information, see "DeactivateFields Method for a Business Component" on page 194.

CAUTION: Do not use the ActivateField method to activate a field in a UI context business component. This technique might cause unexpected Siebel application behavior. For more information about UI context objects, see Doc ID 477419.1 on My Oracle Support.

Format for the ActivateField Method
BusComp.ActivateField(*FieldName*)

Table 55 describes the arguments for the ActivateField method.

Table 55.   Arguments for the ActivateField Method

Argument	Description
FieldName	String variable or literal that contains the name of the field.

You must enclose the FieldName argument in double quotes. The value you enter for the FieldName argument must match exactly the field name that displays in Siebel Tools, including the same case. For example:

```
ActivateField("ActivityCreatedByName")
```

Usage for the ActivateField Method

By default, a field is inactive except in the following situations:

■ The field is a system field, such as Id, Created, Created By, Updated, or Updated By.

■ The Force Active property of the field is TRUE.

  If you write an event handler on a business component, then you must use the ForceActive user property on the control to make sure the field is active. For more information, see *Siebel Developer's Reference*.

■ The Link Specification property of the field is TRUE.

■ The field is included in an applet, and this applet references a business component that is active. For a field in a list applet, the Show In List list column property is TRUE.

■ Siebel CRM calls the ActivateField method on the field, and then runs the ExecuteQuery method.

Note the following:

■ If Siebel CRM activates a field after it queries a business component, then it must requery the business component before the user can access the value in that field. If Siebel CRM does not requery the business component, then it returns a value of 0.

■ If Siebel CRM calls the ActivateField method after it calls the ExecuteQuery method, then the ActivateField method deletes the query context.

■ The ActivateField method causes Siebel CRM to include the field in the SQL statement that the ExecuteQuery method starts. If Siebel CRM activates a field, and then if a statement in the GetFieldValue method or the SetFieldValue method references the file before Siebel CRM performs a statement from the ExecuteQuery method, then the activation has no effect. The query contains an empty value because Siebel CRM does not return the activated field through this query.

■ Siebel CRM does not restrict the maximum number of fields that the ActivateField method can activate. This number depends on the SQL query limitations of the database that your deployment uses.

Avoiding a Corrupted Database

If Siebel CRM does not activate a field before it performs a WriteRecord command, then it writes data to the Siebel database, but a corruption problem might occur if a mobile user synchronizes. This situation applies only to mobile users.

To avoid a corrupted database

1   Use the ActivateField method to call a field.

2   Call the ExecuteQuery method.

3   Call the WriteRecord method.

Using this sequence makes sure Siebel CRM writes the field correctly to the transaction log. During synchronization, it saves any modifications that the mobile user makes back to the Siebel database correctly.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel VB. For an equivalent Siebel eScript example, see "ClearToQuery Method for a Business Component" on page 193:

```
Dim oEmpBusObj As BusObject
Dim oEmpBusComp As BusComp
Dim sLoginName As String

Set oEmpBusObj = TheApplication.ActiveBusObject
Set oEmpBusComp = oEmpBusObj.GetBusComp("Employee")
oEmpBusComp.SetViewMode AllView
oEmpBusComp.ClearToQuery
oEmpBusComp.SetSearchSpec "Login Name", sLoginName
oEmpBusComp.ExecuteQuery ForwardBackward
Set oEmpBusComp = Nothing
Set oEmpBusObj = Nothing
```

ActivateMultipleFields Method for a Business Component

The ActivateMultipleFields method activates multiple fields. This method returns one of the following values:

■   TRUE if the activation is successful

■   FALSE if the activation is not successful

Format

*BusComp.*ActivateMultipleFields(*SiebelPropertySet*)

Table 56 describes the arguments for the ActivateMultipleFields method.

Table 56.   Arguments for the ActivateMultipleFields Method

Argument	Description
SiebelPropertySet	Property set that identifies a collection of properties. These properties identify the fields that Siebel CRM must activate.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is for Siebel Java Data Bean:

```
import com.siebel.data.*;
...
//Create Siebel Java Data Bean.
//log in to Siebel Java Data Bean
...
//Create Siebel Bus Object.
//Get the Bus Object from SiebelDataBean
...
//Create Siebel Bus Comp siebBusComp
//Get the business component using SiebelBusObject

SiebelPropertySet ps = new mdata_bean.NewPropertySet();
ps.setProperty("Account Products","");
ps.setProperty("Agreement Name","");
ps.setProperty("Project Name","");
ps.setProperty("Description","");
ps.setProperty("Name","");
siebBusComp.ActivateMultipleFields(ps);
...
```

The following Siebel eScript example queries the Contact business component and returns the First Name and Last Name of the first contact that it finds:

```
var ContactBO = TheApplication().GetBusObject("Contact");
var ContactBC = ContactBO.GetBusComp("Contact");
with (ContactBC)
{
    SetViewMode(AllView);
    var fieldsPS = TheApplication().NewPropertySet();
    var valuesPS = TheApplication().NewPropertySet();
    fieldsPS.SetProperty("Last Name", "");
    fieldsPS.SetProperty("First Name", "");
    ActivateMultipleFields(fieldsPS);
    ClearToQuery();
    ExecuteQuery(ForwardBackward);
```

```
        if (FirstRecord())
        {
            GetMultipleFieldValues(fieldsPS, valuesPS);
            var slName = valuesPS.GetProperty("Last Name");
            var sfName = valuesPS.GetProperty("First Name");
        }
    }
```

Related Topics

For more information, see the following topics:

■ "SetMultipleFieldValues Method for a Business Component" on page 232

■ "GetMultipleFieldValues Method for a Business Component" on page 209

Associate Method for a Business Component

The Associate method creates a new many-to-many relationship for the parent object through an association business component. This method does not return any information. For more information, see "GetAssocBusComp Method for a Business Component" on page 204.

Format

BusComp.Associate(*whereIndicator*)

Table 57 describes the arguments for the Associate method.

Table 57.   Arguments for the Associate Method

Argument	Description
whereIndicator	You must use one of the following predefined constants: ■ NewBefore ■ NewAfter For more information, see "Use Constants to Standardize Code" on page 68.

Usage

To set field values on a child record that is associated with a parent record, use the context of the multivalue group business component.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following VB example updates the Opportunity Assignment Type field. The parent business component can be any business component that includes the Sales Rep multivalue group:

```
Dim oParentBC as BusComp
Dim oMvgBC as BusComp
Dim oAssocBC as BusComp

Set oParentBC = me.BusComp
Set oMvgBC = OpBC.GetMVGBusComp("Sales Rep")
Set oAssocBC = oMvgBC.GetAssocBusComp
With oAssocBC
    .SetSearchSpec "Id", newPosId
    .ExecuteQuery
    .Associate NewAfter
End With

oMvgBC.SetFieldValue "Opportunity Assignment Type", "NewType"
oMvgBC.WriteRecord
Set oAssocBC = Nothing
Set oMvgBC = Nothing
Set oParentBC = Nothing
```

The following Siebel eScript example finds a contact when the Last Name is Abanilla, and then adds a new organization named CKS Software to the Organization multivalue group:

```
var ok = 0;
var ContactBO= TheApplication().GetBusObject("Contact");
var ContactBC = ContactBO.GetBusComp("Contact");
with (ContactBC)
{
    ClearToQuery();
    SetViewMode(AllView);

    // Searches by Last Name
    SetSearchSpec ("Last Name", "Abanilla");
    ExecuteQuery(ForwardOnly);
    if (FirstRecord())
    {

        // Instantiates Organization MVG
        var oMvgBC = GetMVGBusComp("Organization");
        var oAssocBC = oMvgBC.GetAssocBusComp();
        oAssocBC.ClearToQuery();
        oAssocBC.SetSearchSpec("Name", "CKS Software");
        oAssocBC.ExecuteQuery ();

        // Checks if the Organization was found
        if (oAssocBC.FirstRecord())
        {

            // Organization was found
            try
            {
```

```
            oAssocBC.Associate(NewAfter);
            ok = 1;
        }

        catch (e)
        {
            ok = 0;
            TheApplication().RaiseErrorText("Error Associating new Organization");
        }

    } // if oAssocBC.FirstRecord

} // if FirstRecord

oAssocBC = null;
oMvgBC = null;

} // With ContactBC

ContactBC = null;
ContactBO = null;
```

Related Topics

For more information, see the following topics:

■ "NewRecord Method for a Business Component" on page 220

■ "FirstSelected Method for a Business Component" on page 202

■ "GetMVGBusComp Method for a Business Component" on page 211

BusObject Method for a Business Component

The BusObject method returns the name of the business object that the business component references. For more information, see "ActiveBusObject Method for an Application" on page 127.

Format

BusComp.BusObject

No arguments are available.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

For an example, see "SetViewMode Method for a Business Component" on page 246.

ClearToQuery Method for a Business Component

The ClearToQuery method clears the current query but does not clear sort specifications on a business component. This method does not return any information. For more information, see "RefineQuery Method for a Business Component" on page 226.

Format

BusComp.ClearToQuery

No arguments are available.

Usage

You must use the ActivateField method to activate a field before you can use the ClearToQuery method. For more information, see "ActivateField Method for a Business Component" on page 186.

Search and sort specifications sent to a business component are cumulative. The business component retains and logically performs an AND operation for the queries that accumulate since the last time Siebel CRM performed the ClearToQuery method. This situation is true except if there is a new search specification on a field, and if that field already included a search specification. In this situation, the new search specification replaces the old search specification.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel eScript.

```
var oEmpBusObj  = TheApplication().ActiveBusObject();
var oEmpBusComp = oEmpBusObj ().GetBusComp("Employee");
var sLoginName;

oEmpBusComp.ClearToQuery();
oEmpBusComp.SetSearchSpec("Login Name", sLoginName);
oEmpBusComp.ExecuteQuery(ForwardBackward);

oEmpBusComp = null;
oEmpBusObj  = null;
```

For more examples, see the following:

■ For Siebel VB examples, see the following topics:

  ■ "Applet_PreInvokeMethod Event" on page 115

  ■ "ActivateField Method for a Business Component" on page 186

  ■ "ExecuteQuery Method for a Business Component" on page 197.

■ For another Siebel eScript example, see "GotoView Method for an Application" on page 145.

CountRecords Method for a Business Component

The CountRecords method returns the number of records that the most recent call to the ExecuteQuery method returned.

Format

*BusComp.*CountRecords()

No arguments are available.

Used With

Server Script

Examples

The following example is in Siebel eScript:

```
function Service_PreInvokeMethod (MethodName, Inputs, Outputs)
{
    if (MethodName == "Call_eScript")
    {
        var bo = TheApplication().GetBusObject("Opportunity");
        var bc = bo.GetBusComp("Opportunity");
        with (bc)
        {
            ClearToQuery();
            SetSearchSpec ("Name", "A*");
            ExecuteQuery(ForwardBackward);
            var count = CountRecords();
        }

        // other code..

        bc = null;
        bo = null;

        return (CancelOperation);
    }

    return (ContinueOperation);
}
```

DeactivateFields Method for a Business Component

The DeactivateFields method deactivates fields from the SQL query statement of a business component. It deactivates fields that are currently active. This situation is true except in the following situations:

■ The Force Active property is TRUE

■ A link requires the field to remain active.

■ A business component class requires the field to remain active.

This method does not return any information.

Format

*BusComp.*DeactivateFields

No arguments are available.

Usage

You must use the ActivateField method to activate a field before you perform a query for a business component. For more information, see "ActivateField Method for a Business Component" on page 186.

After you deactivate a field, you must query the business component again or the Siebel application fails.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is for the Component Object Model (COM):

```
Dim oBO As BusObject
Dim OBC As BusComp
Dim errCode

Set oBO = SiebelApplication.GetBusObject("Account", errCode)
Set oBC = oBO.GetBusComp("Account", errCode)
oBC.ActivateField "Name", errCode
oBC.ActivateField "Location", errCode
oBC.ClearToQuery errCode
oBC.ExecuteQuery ForwardOnly, errCode

' Manipulate the data

oBC.DeactivateFields errCode
Set oBC = Nothing
Set oBO = Nothing
```

The following example is in Siebel eScript:

```
var oBC;
var oBO;

oBO = TheApplication().GetBusObject("Account");
oBC = oBO.GetBusComp("Account");
oBC.ActivateField("Name");
oBC.ActivateField("Location");
oBC.ClearToQuery();
oBC.ExecuteQuery(ForwardOnly);

// Manipulate the data
```

```
oBC.DeactivateFields();
oBC = null;
oBO = null;
```

The following example is in Siebel VB:

```
Dim oBO As BusObject
Dim oBC As BusComp

Set oBO = TheApplication.GetBusObject("Account")
Set oBC = oBO.GetBusComp("Account")
oBC.ActivateField "Name"
oBC.ActivateField "Location"
oBC.ClearToQuery
oBC.ExecuteQuery ForwardOnly

' Manipulate the data

oBC.DeactivateFields
Set oBC = Nothing
Set oBO = Nothing
```

DeleteRecord Method for a Business Component

The DeleteRecord method removes the current record from a business component. This method does not return any information.

Format

BusComp.DeleteRecord

No arguments are available.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

This Siebel VB example deletes accounts with a status of Inactive:

```
Sub DeleteInactiveAccounts()
    Dim objBO as BusObject
    Dim objBC as BusComp

    Set objBO = TheApplication.GetBusObject("Account")
    Set objBC = objBO.GetBusComp("Account")
    With objBC
        .ClearToQuery
        .SetSearchSpec "Status", "Inactive"
        .ExecuteQuery ForwardBackward
        Do While .FirstRecord
            .DeleteRecord
```

```
        Loop
    End With
    Set objBC = Nothing
    Set objBO = Nothing
End Sub
```

Siebel CRM moves the cursor to the next record after it runs the DeleteRecord method. Do not use the NextRecord method after you use the DeleteRecord method in a loop because this configuration causes Siebel CRM to skip deleting the last record in the loop. If you use the DeleteRecord method on the last record, then the cursor points to nothing.

ExecuteQuery Method for a Business Component

The ExecuteQuery method uses criteria form another method, such as the SetSearchSpec method, to return a set of business component records. This method allows you to specify the order that Siebel CRM uses to process records.

Format

BusComp.ExecuteQuery ([*cursorMode*])

Table 58 describes the arguments for the ExecuteQuery method.

Table 58.   Arguments for the ExecuteQuery Method

Argument	Description
cursorMode	An integer. You must use one of the following constants: ■ **ForwardBackward.** Siebel CRM processes records from first to last or from last to first. If you do not provide a value for the cursorMode argument, then Siebel CRM uses ForwardBackward. ■ **ForwardOnly.** Siebel CRM processes records only from the first record to the last record. Siebel CRM does return to a prior record. For more information, see "Use Constants to Standardize Code" on page 68.

Usage

To achieve maximum performance, use ForwardOnly. If you use ForwardOnly, make sure that your Siebel application code does not use PreviousRecord or FirstRecord to navigate backward without a requery. Do not use ForwardOnly with a UI business component unless the Siebel application code performs a requery with the cursorMode argument set to ForwardBackward.

A *UI business component* is a type of business component that Siebel CRM is actively using in the Siebel client. You can write a script that creates a UI business component that does not reference the data the user manipulates. A user might scroll up and down a record set, so you must use ForwardBackward.

You Must Activate Fields Before You Can Query Them

Before you can query a business component, you must use the ActivateField method to activate all fields that are involved in the query. If you write an event handler on a business component, then you must use the ForceActive user property on the control to make sure the field is activate.

Reducing a Large Query Set

If you use ForwardBackward, and if the query matches over 10,000 records, then the object manager returns an error message that is similar to the following:

There were more rows than could be returned. Refine your query to bring back fewer rows.

To reduce the number of queries, you can use a parent-child relationship between business components that the business object establishes. For example, the Opportunity business object establishes a parent-child relationship between the Opportunity business component and the Contact business component. If you instruct Siebel CRM to query the Opportunity business component, then it can read values from the corresponding records in the Contact business component without performing another query. You must instruct Siebel CRM to query the parent business component first, and then to query the child business component. If you query the child business component first, then Siebel CRM returns no records.

How Siebel CRM Handles Duplicate Records with the ExecuteQuery Method

A faulty join configuration or duplicate data in joined tables might cause a business component to return duplicate records. If Siebel CRM detects duplicate records when it executes the ExecuteQuery method, then it does the following work depending on the value of the cursorMode argument:

■ **ForwardBackward.** It automatically filters duplicate records to make sure each record is unique.

■ **ForwardOnly.** It does not filter records. It returns all records that match the criteria, including duplicate records. If you update all records that Siebel CRM returns, then it displays an error that is similar to the following:

The selected record has been modified by another user since it was retrieved. Please continue.

This error can occur if the code attempts to update the duplicate of a record that it already updated.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

This Siebel VB example sets up and runs a query that locates the primary on the account team. Only the primary can modify the primary address.

```
(general)
(declarations)
Option Explicit
```

```
Function BusComp_PreSetFieldValue (FieldName As String,
FieldValue As String) As Integer
Dim i As Integer
Dim iFoundP As Integer ' 1 = found (TRUE), 0 = not found (FALSE)
Dim oMVGBC as BusComp

iFoundP = FALSE
Select Case FieldName
Case "SSA Primary Field"
    Set oMVGBC = me.ParentBusComp.GetMVGBusComp("Sales Rep")
    With oMVGBC ' this is the position BC
        .ActivateField "Active Login Name"
        .ActivateField "SSA Primary Field"
        .ClearToQuery
        .ExecuteQuery ForwardBackward
        i = .FirstRecord
        Do While i <> 0
            If .GetFieldValue("SSA Primary Field") = "Y" Then
                iFoundP = TRUE 'mark that found a primary
                If .GetFieldValue("Active Login Name") <> TheApplication.LoginName Then
                    TheApplication.RaiseErrorText"You cannot modify the Primary address
                    because you are not the Primary on the Account Team")
            End If
        Exit Do
        Loop
    If iFoundP = FALSE Then
        .FirstRecord
        TheApplication.RaiseErrorText("No Primary Found - Contact an Administrator")
    End If
    End With
End Select

Set oMVGBC = Nothing
BusComp_PreSetFieldValue = ContinueOperation

End Function
```

For other examples, see the following topics:

■ "Applet_PreInvokeMethod Event" on page 115

■ "GotoView Method for an Application" on page 145

■ "ClearToQuery Method for a Business Component" on page 193:

Related Topics

For more information, see the following topics:

■ "ActivateField Method for a Business Component" on page 186

■ "ClearToQuery Method for a Business Component" on page 193

■ "SetSearchSpec Method for a Business Component" on page 238

ExecuteQuery2 Method for a Business Component

The ExecuteQuery2 method uses criteria form another method, such as SetSearchSpec, to return a set of business component records. Allows you to control the number of records Siebel CRM returns.

Format

*BusComp.*ExecuteQuery2 ([*cursorMode]*, *ignoreMaxCursorSize*)

Table 59 describes the ignoreMaxCursorSize argument for the ExecuteQuery2 method. For the cursorMode argument, see Table 58 on page 197.

Table 59. Arguments for the ExecuteQuery2 Method

Argument	Description
ignoreMaxCursorSize	You can use one of the following values: ■ **TRUE.** Returns every record from a business component. This value might result in lower performance. ■ **FALSE.** Returns the number of records according to the value in the MaxCursorSize argument. You can define the MaxCursorSize argument in the Siebel application configuration (CFG) file.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

FirstRecord Method for a Business Component

The FirstRecord method moves the record pointer to the first record in a business component, making that record the current record. It also calls any associated script events. This method returns the following information:

■ An integer in Siebel VB. It returns 1 or nonzero if it finds at least one record. It returns 0 (zero) if it does not find any records.

■ a Boolean value in Siebel eScript, COM, or ActiveX.

If you issue a query on a business component, then Siebel CRM creates SQL for any child business component that is active. Calling the FirstRecord method starts the BusComp_ChangeRecord event and causes Siebel CRM to run the same SQL for the child business component again.

For more information, see "NextRecord Method for a Business Component" on page 222.

Format

*BusComp.*FirstRecord

No arguments are available.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

To determine if an account displayed in a child applet includes a service request, the following examples use the FirstRecord method. The outcome of this query can determine if Siebel CRM must run other code for this account record. In this example, the Account List Applet is a child applet in the Contact Detail - Accounts View.

The following example is in Siebel eScript:

```
function BusComp_PreInvokeMethod (MethodName)
{
    // 'CheckSR' method called from a custom button on 'Account List Applet - child'
applet.
    if (MethodName == "CheckSR")
    {
        var oBO = TheApplication().ActiveBusObject();
        var oBC = oBO.GetBusComp("Service Request");
        var strAccntId = this.GetFieldValue("Id");

        with (oBC)
        {
            SetViewMode(AllView);
            ClearToQuery();
            SetSearchSpec("Account Id", strAccntId);
            ExecuteQuery(ForwardOnly);
            if (FirstRecord())
            {

                // more code placed here
            }

            else
            {
                TheApplication().RaiseErrorText("No Service Requests Associated To This
Account.")
            }

        }

        return (CancelOperation);
    }

    return (ContinueOperation);
}
```

The following example is in Siebel VB:

```
Function BusComp_PreInvokeMethod (MethodName As String) As Integer

    Dim iRtn As Integer
```

```
        iRtn = ContinueOperation

        ''CheckSR' method called from a custom button On 'Account List Applet - child'
    Applet.
    If MethodName = "CheckSR" Then
        Dim oBO As BusObject
        Dim oBC As BusComp
        Dim strAccntId As String

        Set oBO = TheApplication.ActiveBusObject
        Set oBC = oBO.GetBusComp("Service Request")
        strAccntId = me.GetFieldValue("Id")

        With oBC
            .SetViewMode AllView
            .ClearToQuery
            .SetSearchSpec "Account Id", strAccntId
            .ExecuteQuery ForwardOnly
            If .FirstRecord Then
                '[more code placed here]
            Else
                TheApplication.RaiseErrorText("No Service Requests Associated To This
    Account.")
            End If

        End With

        Set oBC = Nothing
        Set oBO = Nothing

        iRtn = CancelOperation
    End If

    BusComp_PreInvokeMethod = iRtn
End Function
```

FirstSelected Method for a Business Component

The FirstSelected method makes the first record of the multiple selection in a business component active. It also calls any associated events. It returns the same information as the FirstRecord method. For more information, see "FirstRecord Method for a Business Component" on page 200.

Format

*BusComp.*FirstSelected

No arguments are available.

Used With

COM Data Server, Server Script

Examples

The following examples use the FirstSelected method and the NextSelected method to allow you to customize multirecord deletion. If the user clicks a custom button in an applet, then Siebel CRM can call this code and it can call the Delete Selected custom method.

The following example is in Siebel eScript:

```
function BusComp_PreInvokeMethod (MethodName)
{
    if (MethodName == "Delete Selected")
    {
        with (this)
        {
            var iRecord = FirstSelected();

            while (iRecord)
            {
                DeleteRecord();
                iRecord = NextSelected();
            }

        }

        return (CancelOperation);
    }

    return (ContinueOperation);
}
```

The following example is in Siebel VB:

```
Function BusComp_PreInvokeMethod (MethodName As String) As Integer

    Dim iRtn As Integer

    iRtn = ContinueOperation
    If MethodName = "Delete Selected" Then

        With me
            Dim iRecord As Integer

            iRecord = .FirstSelected

            While iRecord
                .DeleteRecord
                iRecord = .NextSelected
            Wend

        End With

        iRtn = CancelOperation

    End If

    BusComp_PreInvokeMethod = iRtn
End Function
```

GetAssocBusComp Method for a Business Component

The GetAssocBusComp method returns a string that contains the name of the association business component. You can use the association business component to manipulate the association.

Format

BusComp.GetAssocBusComp

No arguments are available.

Usage for the GetAssocBusComp Method

It is appropriate to use the GetAssocBusComp method and the Associate method only with a many-to-many relationship that uses an intersection table. For example, account and industry. In the context of a many-to-many relationship, you can use Siebel VB to do the following:

■ To associate a new record, add it to the child business component. To add a record, you use the GetAssocBusComp method and the Associate method. You set the GetAssocBusComp method to Nothing in Siebel VB or null in Siebel eScript.

■ To insert a record, create a new record in the child business component. To insert a record, you use the GetMVGBusComp method and the NewRecord method.

If a many-to-many link exists, and if an association applet is defined for the child applet, then you can use the GetAssocBusComp method with the child business component of a parent-child view. You can use this technique instead of modifying the multivalue group business component.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel VB. It uses the GetAssocBusComp method to add a new industry to an account:

```
Dim oAssocBC As BusComp

Set oAssocBC = oMainBc.GetMVGBusComp("Industry").GetAssocBusComp
With oAssocBC
    .ClearToQuery
    .SetSearchExpr "[SIC Code] = ""5734"""
    .ExecuteQuery ForwardOnly

    If .FirstRecord Then .Associate NewBefore
End With
Set oAssocBC = Nothing
```

The following is the same example in Siebel eScript:

```
//get the business Object and the business component
var oAssocBC = oMainBc.GetMVGBusComp("Industry").GetAssocBusComp();
with (oAssocBC)
```

```
{
    ClearToQuery;
    SetSearchExpr("[SIC Code] = '5734'");
    ExecuteQuery(ForwardOnly)
    if (FirstRecord())
        Associate(NewBefore);
}
oAssocBC = null;
```

Related Topics

For more information, see the following topics:

■ “GetMVGBusComp Method for a Business Component” on page 211

■ “GetPicklistBusComp Method for a Business Component” on page 213

GetFieldValue Method for a Business Component

The GetFieldValue method returns one of the following items:

■ A string that contains the value of a field from the current record of a business component.

■ An empty string if the field is empty.

■ An error message if the field is inactive. To avoid this situation, activate the field before you use the GetFieldValue method. For more information, see “ActivateField Method for a Business Component” on page 186.

The GetFieldValue method uses the MM/DD/YYYY format when it returns a date field regardless of what format the local date uses. To return the date in the same format that the local date uses, you can use the GetFormattedFieldValue method. For more information, see “GetFormattedFieldValue Method for a Business Component” on page 206.

In Browser Script, you can use the GetFieldValue method only if the field is available in the applet and for system fields.

Format

BusComp.GetFieldValue(*FieldName*)

The arguments you can use in this format are the same as the arguments that are described in Table 55 on page 187.

Usage for the GetFieldValue Method

If you require a value from a business component that is a parent of the current business component, then you must make sure the Link Specification property for that field is set to TRUE in Siebel Tools. If it is not, then the child business component cannot access the value in the parent business component. For more information, see *Siebel Object Types Reference.*

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client
Automation Server, Server Script

Examples

The following example is in Siebel VB:

```
Function BusComp_PreSetFieldValue (FieldName As String, FieldValue As String) As
Integer

    Dim bcOppty As BusComp
    Dim boBusObj As BusObject
    Dim srowid As String

    srowid = GetFieldValue("Id")
    Set boBusObj = TheApplication.GetBusObject("Opportunity")
    Set bcOppty = boBusObj.GetBusComp("Opportunity")
    With bcOppty
        .SetViewMode SalesRepView
        .ActivateField "Sales Stage"
        .SetSearchSpec "Id", srowid
        .ExecuteQuery ForwardOnly
    End With

    Set bcOppty = Nothing
    Set boBusObj = Nothing

End Function
```

The following example is in Siebel eScript:

```
function BusComp_PreSetFieldValue (FieldName, FieldValue)

    var boBusObj = TheApplication().GetBusObject("Opportunity");
    var bcOppty = boBusObj.GetBusComp("Opportunity");
    var srowid = GetFieldValue("Id");

    with (bcOppty)
    {
        SetViewMode(SalesRepView);
        ActivateField("Sales Stage");
        SetSearchSpec("Id", srowid);
        ExecuteQuery(ForwardOnly);
    }

    bcOppty = null;
    boBusObj = null;
}
```

GetFormattedFieldValue Method for a Business Component

The GetFormattedFieldValue method returns the following information:

■ A string that contains a field value that is in the same format that the Siebel client uses.

■ An empty string if the field is inactive or empty.

Format

*BusComp.*GetFormattedFieldValue(*FieldName*)

The arguments you can use in this format are the same as the arguments that are described in .

Usage

You can use the GetFormattedFieldValue method with code that your implementation uses in multiple countries that use different formats for currency, date, or numbers.

Usage with Phone Data and Date Data

The following behavior exists for phone data and date data:

■ **DTYPE_PHONE.** If you use the GetFormattedFieldValue method with a field whose Type property is DTYPE_PHONE, then this method returns a formatted phone number.

Example 1:

```
phone = bc.GetFieldValue("Main Phone Number")
TheApplication.Trace "The number is " & phone
```

Result:

```
The number is 8869629123
```

Example 2:

```
phone = bc.GetFormattedFieldValue("Main Phone Number")
TheApplication.Trace "The number is " & phone
```

Result:

```
The number is (886) 962-9123
```

■ **DTYPE_DATE.** If you use the GetFormattedFieldValue method with a field whose Type property is DTYPE_DATE, then the result is the same as the GetFieldValue method or the SetFieldValue method except that the GetFormattedFieldValue method returns the value in the same format as the Regional Setting.

Table 60 describes the formats that the GetFieldValue method and the SetFieldValue method use.

Table 60.   Date and Time Formats That the GetFieldValue Method and SetFieldValue Method Use

Type of Data	Format
Dates	mm/dd/yyyy
Times	hh:nn:ss
Date-times	mm/dd/yyyy hh:nn:ss

If you attempt to use the SetFieldValue method, and if the Regional Setting format is different, the Siebel CRM displays an error that is similar to the following:

Error: The value '31-Dec-99' can not be converted to a date time value.

To avoid this error, use the GetFormattedFieldValue format or the SetFormattedFieldValue method.

Used With
Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples
The following Siebel VB example uses the GetFormattedFieldValue method and calculates the number of days between two dates:

```
Sub Button_Click
    Dim DateDiff as Integer
    Dim oBC as BusComp
    Set oBC= me.BusComp
    x = oBC.GetFormattedFieldValue("Start Date")
    y = oBC.GetFormattedFieldValue("Done")
    dx = DateValue(x)
    dy = DateValue(y)
    DateDiff = dy - dx
End Sub
```

Related Topics
For more information, see the following topics:

■ "ActivateField Method for a Business Component" on page 186

■ "GetFieldValue Method for a Business Component" on page 205

■ "SetFieldValue Method for a Business Component" on page 229

■ "SetFormattedFieldValue Method for a Business Component" on page 230

GetLastErrCode Method for a Business Component
The GetLastErrCode method returns the error code for the error that Siebel CRM logged most recently. This code is a short integer. 0 (zero) indicates no error.

Format
BusComp.GetLastErrCode

No arguments are available.

Usage
For more information, see "Usage for the GetLastErrCode Method" on page 139.

Used With

COM Data Control, Mobile Web Client Automation Server

GetLastErrText Method for a Business Component

The GetLastErrText method returns a string that contains the text message for the error that Siebel CRM logged most recently.

Format

*BusComp.*GetLastErrText

No arguments are available.

Usage

For more information, see .

Used With

COM Data Control, Mobile Web Client Automation Server

GetMultipleFieldValues Method for a Business Component

The GetMultipleFieldValues method returns a value for each field specified in a property set. It also returns the following information:

■   TRUE if it finds the fields.

■   FALSE if it does not find the fields.

For more information, see .

Format

*BusComp.*GetMultipleFieldValues(*fieldNamesPropSet*, *fieldValuesPropSet*)

Table 61 describes the arguments for the GetMultipleFieldValues method.

Table 61.   Arguments for the GetMultipleFieldValues Method

Argument	Description
fieldNamesPropSet	A property set that identifies a collection of fields.
fieldValuesPropSet	A property set that provides values for the fields specified in the fieldNamesPropSet argument.

Usage

You cannot use the same instance of a property set for the fieldNamesPropSet argument and for the fieldValuesPropSet argument.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel eScript:

```
try {

  var oPsDR_Header: PropertySet = TheApplication().NewPropertySet();

  // Cannot use the same property set in GetMultipleFieldValues, must use a different
  // one for the values. The process will not error, but the values will not be placed
  // in the property set.

  var IPS_values: PropertySet = TheApplication().NewPropertySet();

  oPsDR_Header.SetProperty("Last Name","");

  oPsDR_Header.SetProperty("First Name","");

  oPsDR_Header.SetProperty("Middle Name","");

  var boContact = TheApplication().GetBusObject("Contact");

  var bcContact = boContact.GetBusComp("Contact");

      with (bcContact) {

          ClearToQuery();

          SetViewMode(AllView);

          ActivateMultipleFields(oPsDR_Header);

          SetSearchSpec("Last Name", "Mead*");

          ExecuteQuery(ForwardOnly);

          var isParent = FirstRecord();

          do {

            // Use a different property set for the values. If you use the same one

            // for arguments you get no values back.

            GetMultipleFieldValues(oPsDR_Header, IPS_values);

            // Get the value from the output property set.

            TheApplication().Trace("Last Name = " +
            IPS_values.GetProperty("Last Name"));

            } while (NextRecord());

          } //end with
```

```
    } //end try

  catch(e) {

    TheApplication().Trace(e.toString());

    }//end catch
```

GetMVGBusComp Method for a Business Component

The GetMVGBusComp method returns the multivalue group business component that is associated with a business component field.

Format

*BusComp.*GetMVGBusComp(*FieldName*)

The arguments you can use in this format are the same as the arguments that are described in Table 55 on page 187 except the GetMVGBusComp method uses the FieldName argument to identify the multivalue group business component.

Usage

A *multivalue group* is a set of detail records attached to the current record in a business component that holds the corresponding multivalue field. After you run the GetMVGBusComp method, it is recommended that you set the multivalue group business component to one of the following:

■ Nothing in Siebel VB

■ Null in Siebel eScript

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example Siebel VB code uses the GetMVGBusComp method to add a new address to the Hong Kong Flower Shop account:

```
Dim AccntBO as BusObject
Dim AccntBC as BusComp
Dim AddrBC as BusComp
Set AccntBO = TheApplication.GetBusObject "Account"
Set AccntBC = AccntBO.GetBusComp "Account"

With AccntBC
    .SetViewMode SalesRepView
    .ClearToQuery
    .SetSearchSpec "Name", "Hong Kong Flower Shop"
    .ExecuteQuery
    If (.FirstRecord) Then Set AddrBC = .GetMVGBusComp ("Street Address")
```

```
        With AddrBC
            .NewRecord NewAfter
            .SetFieldValue "City", "Denver"
            .SetFieldValue "Street Address", "123 Main Street"
            .WriteRecord
        End With

        End If

    End With

    Set AddrBC = Nothing
    Set AccntBC = Nothing
    Set AccntBO = Nothing
```

For more examples, see the following topics:

■ "ExecuteQuery Method for a Business Component" on page 197

■ "FirstSelected Method for a Business Component" on page 202.

For more information about inserting records, see "Usage for the GetAssocBusComp Method" on page 204:

GetNamedSearch Method for a Business Component

The GetNamedSearch method returns a string that contains the name of a search specification.

Format

*BusComp.*GetNamedSearch(*searchName*)

Table 62 describes the arguments for the GetNamedSearch method.

Table 62.   Arguments for the GetNamedSearch Method

Argument	Description
searchName	Name of the search specification that references the search string.

Usage

The search specification uses the same format that a predefined query uses.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Related Topics

For more information, see the following topics:

■ "GetSearchSpec Method for a Business Component" on page 215

■ "SetNamedSearch Method for a Business Component" on page 234

GetPicklistBusComp Method for a Business Component

The GetPicklistBusComp method returns the name of the pick business component that is associated with a field in the current business component. If there is no picklist associated with this field, then this method returns an error.

Format

*BusComp.*GetPicklistBusComp(*FieldName*)

The arguments you can use in this format are the same as the arguments that are described in Table 55 on page 187, except the GetPicklistBusComp method uses the FieldName argument to identify the pick business component.

Usage

To manipulate a picklist, you can use the name of the pick business component that the GetPicklistBusComp method returns.

After you run the GetPickListBusComp method, it is recommended that you set the pick business component to one of the following:

■ Nothing in Siebel VB

■ Null in Siebel eScript

Picking a Record on a Constrained Picklist

If Siebel CRM uses the GetPickListBusComp method or the Pick method to pick a record on a constrained picklist, then the constraint is active. The pick business component that these methods return contains only those records that fulfill the constraint.

To Pick a Value From a Picklist in Siebel VB

You can pick a value from a picklist in Siebel VB.

To pick a value from a picklist in Siebel VB

1 Use the GetPicklistBusComp method to create an instance of the picklist business component.

2 Navigate in the pick business component to the record you must pick.

3 Use Pick to pick the value.

4 To explicitly delete this instance of the pick business component, use the following code:

```
Set objBCPickList = Nothing.
```

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel eScript:

```
if (this.GetFieldValue("City") == "San Mateo")
{
    var oBCPick = this.GetPicklistBusComp("State");
    with (oBCPick)
    {
        ClearToQuery();
        SetSearchSpec("Value", "CA");
        ExecuteQuery(ForwardOnly);
        if (FirstRecord())
            Pick();
    }
    oBCPick = null;
}
```

The following example is for Siebel Java Data Bean. It chooses a product from a picklist:

```
Sieb_busObject = Sieb_dataBean.getBusObject("Service Request");
Sieb_busComp = Sieb_busObject.getBusComp("Service Request");
Sieb_busComp.newRecord(false);

.   .   .

SiebelBusComp productBusComp = Sieb_busComp.getPicklistBusComp("Product");
productBusComp.clearToQuery();
productBusComp.setSearchSpec("Name", "ATM Card");
productBusComp.executeQuery(false);
isRecord =productBusComp.firstRecord();
try
{
    if (isRecord)
    productBusComp.pick();
    Sieb_busComp.writeRecord();
}

catch (SiebelException e)
{
    System.out.println("Error in Pick " + e.getErrorMessage());
}
```

The following example is in Siebel VB:

```
If Me.GetFieldValue("City") = "San Mateo" Then
    Set oBCPick = Me.GetPicklistBusComp("State")
    With oBCPick
        .ClearToQuery
        .SetSearchSpec "Value", "CA"
        .ExecuteQuery ForwardOnly
```

```
        If .FirstRecord Then .Pick
    End With
    Set oBCPick = Nothing
End If
```

Related Topics

For more information, see the following topics:

■ "FirstSelected Method for a Business Component" on page 202

■ "GetMVGBusComp Method for a Business Component" on page 211

GetSearchExpr Method for a Business Component

The GetSearchExpr method returns a string that contains the current search expression that is defined for a business component. For example:

    [Revenue] > 10000 AND [Probability] > .5

If an instance of the business component does not exist, then the GetSearchExpr method returns nothing. If you use the GetSearchExpr method in Browser Script with the Applet_PreInvokeMethod event, then it returns a null value even if you add a query filter.

Format

*BusComp.*GetSearchExpr

No arguments are available.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Related Topics

For more information, see the following topics:

■ "GetNamedSearch Method for a Business Component" on page 212

■ "GetSearchSpec Method for a Business Component" on page 215

■ "SetSearchExpr Method for a Business Component" on page 236

GetSearchSpec Method for a Business Component

The GetSearchSpec method returns a string that contains the search specification that is defined for a business component. For example, > 10000.

Format

*BusComp.*GetSearchSpec(*FieldName*)

The arguments you can use in this format are the same as the arguments that are described in Table 55 on page 187, except the GetSearchSpec method uses the FieldName argument to identify the search specification.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Related Topics

For more information, see the following topics:

■ "GetNamedSearch Method for a Business Component" on page 212

■ "GetSearchExpr Method for a Business Component" on page 215

■ "GetSortSpec Method for a Business Component" on page 216

■ "SetSearchSpec Method for a Business Component" on page 238

GetSortSpec Method for a Business Component

The GetSortSpec method returns the sort specification for a business component.

Format

this.GetSortSpec();

No arguments are available.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Related Topics

For more information, see the following topics:

■ "GetSearchSpec Method for a Business Component" on page 215

■ "SetSortSpec Method for a Business Component" on page 243

GetUserProperty Method for a Business Component

The GetUserProperty method returns the value of a user property.

Format

*BusComp.*GetUserProperty(*propertyName*)

Table 63 describes the arguments for the GetUserProperty method.

Table 63.   Arguments for the GetUserProperty Method

Argument	Description
propertyName	The name of the user property.

Usage for the GetUserProperty Method

A user property is similar to an instance variable of a business component. You can use the GetUserProperty method to access a user property from anywhere in the code, even from another application through COM.

An *instance variable* is a type of variable that is defined at the top level of the business component in the general declarations section. You can access an instance variable only in Siebel VB, and in the same object where you declare the instance variable. For more information, see "SetUserProperty Method for a Business Component" on page 245.

Siebel CRM resets the value of a user property every time you create a business component instance.

The GetUserProperty method does not interact directly with user properties that you define in Siebel Tools.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

GetViewMode Method for a Business Component

The GetViewMode returns a Siebel ViewMode constant or the corresponding integer value for this constant. This constant identifies the current visibility mode for a business component. This mode effects the records that queries return according to the visibility rules. For more information, see "SetViewMode Method for a Business Component" on page 246 and "Use Constants to Standardize Code" on page 68.

Format

*BusComp.*GetViewMode

No arguments are available.

Usage

The GetViewMode method returns NoneSetView mode until one of the following situations is true:

■ Siebel CRM queries a business component.

■ The SetViewMode method sets the view mode for the business component.

The NoneSetViewMode value indicates that no visibility rules are applied to the business component. If Siebel CRM creates a business component through a call to the GetBusComp method, then the value for that business component is NoneSetViewMode. If you require a specific view mode, then you must use the SetViewMode method to set this view mode. If you do not use the SetViewMode method, then Siebel CRM sets the view mode according to the most restrictive visibility mode that is defined for that business component. It does this the first time that it creates a business component instance.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

InvokeMethod Method for a Business Component

The InvokeMethod method calls a method. It returns a string that contains the result of the method. For more information, see "About Specialized and Custom Methods" on page 103.

Siebel VB Format

*BusComp.*InvokeMethod *methodName, methArg1, methArg2, methArgN*

Table 64 describes the arguments for the Siebel VB format of the InvokeMethod method.

Table 64.   Arguments for the Siebel VB Format of the InvokeMethod Method

Argument	Description
methodName	The name of the method. For information about the values you can enter for this argument, see "Business Component Invoke Methods" on page 252.
You can use the following arguments: ■ methArg1 ■ methArg2 ■ methArgN	A single string that contains arguments for the methodName argument. You can also pass this string in an array that contains the method parameters.

Siebel eScript Format

*BusComp.*InvokeMethod(*methodName, methArg1, methArg2, …, methArg*n);

The arguments you can use in this format are the same as the arguments that are described in Table 22 on page 107.

Usage

You can use the InvokeMethod method to call a method on a business component object that is not available directly through the object interface. For more information, see "Caution About Using the InvokeMethod Method" on page 108.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

For examples of using the InvokeMethod method, see the following topics:

■ "ClearLOVCache Method for a Business Component" on page 253

■ "CreateFile Method for a Business Component" on page 254

■ "GetFile Method for a Business Component" on page 257

■ "PutFile Method for a Business Component" on page 259

LastRecord Method for a Business Component

The LastRecord method moves the record pointer to the last record in a business component. It returns one of the following items:

■ An integer in Siebel VB

■ A Boolean value in ActiveX, COM, Siebel Java Data Bean, or Siebel eScript

For more information, see "FirstRecord Method for a Business Component" on page 200 and "NextRecord Method for a Business Component" on page 222.

Format

*BusComp.*LastRecord

No arguments are available.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is for the Mobile Web Client Automation Server:

```
Private Sub LastRecord_Click()

    Dim errCode As Integer
    Dim oBusComp as SiebelBusComp
    FieldValue.Text = ""
```

```
    oBusComp.ClearToQuery
    oBusComp.ExecuteQuery ForwardBackward
    oBusComp.LastRecord errCode
    If errCode = 0 Then
        FieldValue.Text = oBusComp.GetFieldValue(FieldName.Text, _
            errCode)
    End If

    Status.Text = SiebelApplication.GetLastErrText
End Sub
```

Name Method for a Business Component

The Name method returns a string that contains the name of a business component.

Format

BusComp.Name()

No arguments are available.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Browser Script:

```
function BusComp_PreSetFieldValue (fieldName, value)
{
    theApplication().SWEAlert(this.Name());
}
```

NewRecord Method for a Business Component

The NewRecord method adds a new record to a business component. This method does not return any information.

Format

BusComp.NewRecord(*whereIndicator*)

Table 65 describes the arguments for the NewRecord method.

Table 65.   Arguments for the NewRecord Method

Argument	Description
whereIndicator	Predefined constant that configures where Siebel CRM must add the new record. You can use one of the following values: ■ NewBefore ■ NewAfter ■ NewBeforeCopy ■ NewAfterCopy For more information, see "Use Constants to Standardize Code" on page 68. If you use Siebel Java Data Bean, then you can use one of the following values: ■ **FALSE.** This value is equivalent to the NewBefore constant. ■ **TRUE.** This value is equivalent to the NewAfter constant.

Usage

If you use the NewRecord method to add a new record, then Siebel CRM does the following:

1 Places the new record before or after the current record, depending on the value you enter for the WhereIndicator argument.

2 Sets this new record as the current record.

You can use the NewRecord method to copy a record. To place the copy before the original record, you use the following command:

   Object.NewRecord NewBeforeCopy

To place the copy after the original record, you use the following command:

   Object.NewRecord NewAfterCopy

Performance with the NewRecord Method

In some situations, using the NewRecord method in a Server Script can result in this method performing slowly. In this situation, Siebel CRM does not display an error message. It creates the record but the reply time is not optimal. This situation is due to the expected behavior of the Siebel application when it creates a new record.

To position the new record in the record set, Siebel CRM gets the cursor for the record set. This record set must include data before Siebel CRM creates the new record. In the context of a script, Siebel CRM must run a query on the business component before it calls the NewRecord method. If the script does not explicitly run the query, then Siebel CRM runs a full table query. This situation can cause suboptimal performance. For more information, see Doc ID 477556.1 on My Oracle Support.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel VB:

```
Dim oBusObj as BusObject
Dim oBC as BusComp

Set oBusObj = TheApplication.ActiveBusObject
Set oBC = oBusObj.GetBusComp("Action")
oBC.NewRecord NewAfter
oBC.SetFieldValue "Type", "To Do"
oBC.SetFieldValue "Description", "Find Decision Makers"
oBC.WriteRecord

set oBC = Nothing
set oBusObj = Nothing
```

NextRecord Method for a Business Component

The NextRecord method moves the record pointer to the next record in a business component, making that record the current record. This method returns the following information:

- In Siebel VB, an integer that includes one of the following values:

  - **1.** Indicates the method successfully moved the record pointer to the next record.

  - **0 (zero).** Indicates the method did not move the record pointer because it points to the last record.

- In Siebel eScript and COM, a Boolean value.

Format

*BusComp.*NextRecord

No arguments are available.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Browser Script

Examples

The following example is in Siebel eScript:

```
var isRecord;

with (this)
{

    ClearToQuery();
    SetSearchSpec("Name", "A*");
    ExecuteQuery(ForwardBackward);
    isRecord = FirstRecord();
    while (isRecord)
    {

        // do some record manipulation
        isRecord = NextRecord();

    }

}
```

For a similar Siebel VB example, see "FirstRecord Method for a Business Component" on page 200.

NextSelected Method for a Business Component

The NextSelected method makes the next record of the current multiple selection the active record. It returns the same information as the NextRecord method. For more information, see "NextRecord Method for a Business Component" on page 222.

Format

BusComp.NextSelected

No arguments are available.

Used With

Server Script

Examples

For examples, see "FirstSelected Method for a Business Component" on page 202.

ParentBusComp Method for a Business Component

The ParentBusComp method returns the name of the parent business component of a link.

Format

*BusComp.*ParentBusComp

No arguments are available.

Usage

The ParentBusComp method allows you to write code in the child business component that can access a field value or perform actions in the parent business component. To use this method, it might be necessary to set the Link Specification property. For more information, see "Usage for the GetFieldValue Method" on page 205.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel VB. For another example, see "ExecuteQuery Method for a Business Component" on page 197:

```
Dim strParentName as String
...
strParentName = Me.ParentBusComp.GetFieldValue("Name")
```

Pick Method for a Business Component

The Pick method places the currently chosen record in a pick business component into the appropriate fields of the parent business component. This method does not return any information.

You cannot use the Pick method to modify the record in a read-only picklist field.

Format

*BusComp.*Pick

No arguments are available.

Usage

For more information, see "Picking a Record on a Constrained Picklist" on page 213.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following Siebel VB example sorts the values in the Sales Stage field:

```
Sub BusComp_NewRecord
    Dim oBC as BusComp
    set oBC = me.GetPickListBusComp("Sales Stage")
```

```
    With oBC
        .ClearToQuery
        .SetSearchSpec "Sales Stage", "2 - Qualified"
        .ExecuteQuery ForwardOnly
        if .FirstRecord then .Pick
    End With

    set oBC = Nothing
End Sub
```

The following is the equivalent example in Siebel eScript:

```
function BusComp_NewRecord ()
{
    var oBC = this.GetPickListBusComp("Sales Stage");
    with (oBC)
    {
        ClearToQuery();
        SetSearchSpec("Sales Stage", "2 - Qualified");
        ExecuteQuery(ForwardOnly);
        if (FirstRecord())
            Pick();
    }
    oBC = null;
}
```

PreviousRecord Method for a Business Component

The PreviousRecord method moves the record pointer to the previous record in a business component, making that record the current record. This method returns one of the following values:

■ An integer in Siebel VB that includes one of the following values:

 ■ **1.** Indicates the method successfully moved the record pointer to the next record.

 ■ **0 (zero).** Indicates the method did not move the record pointer because it points to the last record.

■ A Boolean value in Siebel eScript and COM.

Format

*BusComp.*PreviousRecord

No arguments are available.

Usage

You can use the PreviousRecord method only on a business component that Siebel CRM has queried with the CursorMode mode argument set to ForwardBackward. For more information, see "ExecuteQuery Method for a Business Component" on page 197.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following Siebel eScript example locates the next-to-last record in a query and then manipulates it:

```
with (this)

{

    ActivateField("Name")

    ClearToQuery();

    SetSearchSpec("Name", "A*");

    ExecuteQuery(ForwardBackward);

    isRecord = FirstRecord();

    while (isRecord)

    {

        // do some record manipulation

        isRecord = NextRecord();

        } // end while loop

    nextToLastRecord = PreviousRecord();

    if (nextToLastRecord)      // verify that there is a penultimate record

    {

        // do some more record manipulation that applies only to next-to-last record

    } // end if

} // end with
```

For more information, see "ExecuteQuery Method for a Business Component" on page 197.

RefineQuery Method for a Business Component

The RefineQuery method refines a query. This method does not return any information.

Format

*BusComp.*RefineQuery

No arguments are available.

Usage

Unlike the ClearToQuery method, the RefineQuery method retains the existing query specification and allows you to add search conditions that include those fields that Siebel CRM has not set through a previous search expression. The RefineQuery method is most useful if you use it with the GetNamedSearch method. For more information, see "ClearToQuery Method for a Business Component" on page 193 and "GetNamedSearch Method for a Business Component" on page 212.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following Siebel VB code uses RefineQuery:

```
me.SetSearchSpec "Status", "Open"
me.ClearToQuery
me.ExecuteQuery
me.RefineQuery
me.SetSearchSpec "Substatus", "Assigned"
me.ExecuteQuery
```

Release Method for a Business Component

The Release method releases a business component and the resources for this business component that exist on the Siebel Server. This method does not return any information.

Format

*BusComp.*release()

No arguments are available.

Used With

Siebel Java Data Bean

Examples

The following example is for Siebel Java Data Bean:

```
import com.siebel.data.*;
{
    …
// create Siebel Java Data Bean
    // log in to Siebel Java Data Bean
…
    // Create Siebel Bus Object.
// get the Bus Object from SiebelDataBean
    …
    // Create Siebel Bus Comp siebBusComp
```

```
// Get the business component using SiebelBusObject
    …
// Use the bus. Component
…
// make sure to release the business component and its resources on the Siebel Server
    siebBusComp.release();
// release the resources occupied by Siebel Bus Object and Siebel Java Data Bean
after their use.
}
```

The following example logs in to a Siebel Server. It then creates an instance for each of the following items:

■ Business object

■ Business component

■ Business service

It then releases each of these items in reverse order:

```
import com.siebel.data.*;
import com.siebel.data.SiebelException;

public class JDBReleaseDemo
{
    private SiebelDataBean m_dataBean = null;
    private SiebelBusObject m_busObject = null;
    private SiebelBusComp     m_busComp = null;
    private SiebelService  m_busServ = null;

    public static void main(String[] args)
    {
        JDBReleaseDemo demo = new JDBReleaseDemo();
    }

    public JDBReleaseDemo()
    {
        try
        {

            // instantiate the Siebel Java Data Bean
            m_dataBean = new SiebelDataBean();

            // login to the Siebel Servers
            m_dataBean.login("siebel.tcpip.none.none://gateway:port/enterprise/
            object manager","userid","password");
            System.out.println("Logged in to the Siebel Server ");

            // get the business object
            m_busObject = m_dataBean.getBusObject("Account");

            // get the business component
            m_busComp = m_busObject.getBusComp("Account");

            // get the business service
            m_busServ = m_dataBean.getService("Workflow Process Manager");
```

```
            //release the business service
            m_busServ.release();
            System.out.println("BS released ");

            //release the business component
            m_busComp.release();

            System.out.println("BC released ");

            //release the business object
            m_busObject.release();
            System.out.println("BO released ");

            // logoff
            m_dataBean.logoff();
            System.out.println("Logged off the Siebel Server ");
        }

        catch (SiebelException e)
        {
            System.out.println(e.getErrorMessage());
        }

    }

}
```

For more information, see "Logoff Method for an Application" on page 156.

SetFieldValue Method for a Business Component

The SetFieldValue method sets a new value for a field in the current record of a business component.
This method does not return any information.

Format

*BusComp.*SetFieldValue *FieldName, FieldValue*

Table 66 describes the arguments for the SetFieldValue method.

Table 66.   Arguments for the SetFieldValue Method

Argument	Description
FieldName	String that contains the name of the field.
FieldValue	String that contains the value to set.

The format for the FieldName argument uses the same format that is described in "Format for the
ActivateField Method" on page 186.

The length of the FieldValue argument must not exceed the length of the field. For example, if you pass a 20 character string to a field that is defined as 16 characters in length, then Siebel CRM creates a run-time error that is similar to the following:

Value too long for field 'xxxxx' (maximum size nnn).

You must make sure the length of the string you pass is no longer than the length of the destination field.

Usage

You can use the SetFieldValue method only on a field that is active. For more information, see "ActivateField Method for a Business Component" on page 186.

If the Siebel application runs in standard interactivity mode, then call the WriteRecord method and write the record immediately after you use the SetFieldValue method.

You cannot use the SetFieldValue method with a calculated field. You cannot use the SetFieldValue method recursively.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel VB:

```
If Val(Me.GetFieldValue("Rep %")) < 75 Then
    Me.SetFieldValue "Rep %", "75"
    Me.WriteRecord
End If
```

The following is the equivalent example in Siebel eScript:

```
if (ToInteger(this.GetFieldValue("Rep %")) < 75)

    {
    this.SetFieldValue("Rep %", "75");
    this.WriteRecord();
    }
```

The following Siebel VB example sets a field to null:

```
oBC.SetFieldValue "FieldName", ""
```

SetFormattedFieldValue Method for a Business Component

The SetFormattedFieldValue method sets a new value in a field in the current record of a business component. It accepts the field value in the current local format. This method does not return any information.

Format
*BusComp.*SetFormattedFieldValue *FieldName, FieldValue*

The arguments you can use this format are the same as the arguments described in Table 66 on page 229.

Usage
The SetFormattedFieldValue method is useful if you write code for a Siebel application that you deploy in multiple countries that use different currency, date, and number formats.

You can use the SetFormattedFieldValue method only on a field that is active. For more information, see "ActivateField Method for a Business Component" on page 186.

Used With
Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples
The following Siebel VB example is a fragment from a program that tracks the progress of an opportunity through sales stages:

```
Function BusComp_PreWriteRecord As Integer

Dim OpportunityBO as BusObject, StageBC as BusComp
Dim OppStageId as String, SalesRep as String, Stage as String
Dim StagePrev As String, StageDate as String, StageDatePrev as String
Dim Dx as Double, Dy as Double, Diff as Double, DiffStr as String
Dim OppID As String, OppStageId as String, StageID As String
Dim SalesStageBO as BusObject, SalesStageBC as BusComp

Set OpportunityBO = TheApplication.GetBusObject ("Opportunity")
Set SalesStageBO = TheApplication.GetBusObject ("Sales Cycle Def")
Set SalesStageBC = SalesStageBO.GetBusComp("Sales Cycle Def")

With SalesStageBC
    .SetViewMode AllView
    .ClearToQuery
    .SetSearchSpec "Sales Cycle Stage", StagePrev
    .ExecuteQuery ForwardOnly
    If (.FirstRecord) Then
        StageId = .GetFieldValue("Id")
End With

'Instantiate stage BC
Set StageBC = OpportunityBO.GetBusComp("Opportunity Stage")

'Check that we do not already have a record for the stage

    With StageBC
        .SetViewMode AllView
        .ClearToQuery
```

```
            .SetSearchSpec "Sales Stage Id", StageId
            .ExecuteQuery ForwardOnly

    'Proceed further only if we do not already have record
    'opportunity sales stage

        If (.FirstRecord = 0) Then
            'Create a new stage record and write it out
                .NewRecord NewAfter
                'Record Id for future use
                OppStageId = .GetFieldValue("Id")
                .SetFieldValue "Opportunity Id", OppId
                .SetFieldValue "Sales Stage Id", StageId
                .SetFieldValue "Sales Rep", SalesRep
                .SetFormattedFieldValue "Entered Date", StageDatePrev
                .SetFormattedFieldValue "Left Date", StageDate
                Dx = DateValue (StageDatePrev)
                Dy = DateValue (StageDate)
                Diff = Dy - Dx
                DiffStr = Str(Diff)
                .SetFieldValue "Days In Stage", DiffStr
                .WriteRecord
        End If
    End With

    Set SalesStageBC = Nothing
    Set SalesStageBO = Nothing
    Set StageBC = Nothing
    Set OpportunityBO = Nothing

    End Function
```

SetMultipleFieldValues Method for a Business Component

The SetMultipleFieldValues method sets new values in the fields of the current record of a business component. This method does not return any information.

Format

BusComp.SetMultipleFieldValues oPropertySet

Table 67 describes the arguments for the SetMultipleFieldValues method.

Table 67.   Arguments for the SetMultipleFieldValues Method

Argument	Description
oPropertySet	Property set that identifies a collection of properties. This argument identifies the fields to set and the value to set for each field.

The FieldName argument in the property set must match exactly the field name in Siebel Tools, including the correct case. In the following example, the FieldName is Name and the FieldValue is Acme:

```
oPropertySet.SetProperty "Name","Acme"
```

Usage

You can use the SetMultipleFieldValues method only on a field that is active.

Do not use the SetMultipleFieldValues method on a field that uses a picklist.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel eScript:

```
var bo = TheApplication().GetBusObject("Opportunity");
var bc = bo.GetBusComp("Opportunity");
var ps = TheApplication().NewPropertySet();

with (ps)
{
    SetProperty ("Name", "Call Center Opportunity");
    SetProperty ("Account", "Marriott International");
    SetProperty ("Sales Stage", "2-Qualified");
}

bc.ActivateMultipleFields(ps);
bc.NewRecord(NewBefore);
bc.SetMultipleFieldValues(ps);
bc.WriteRecord;

ps = null;
bc = null;
bo = null;
```

The following Siebel Java Data Bean example sets multiple fields:

```
SiebelDataBean        Sieb_dataBean     = null;
SiebelBusObject       Sieb_busObject    = null;
SiebelBusComp         Sieb_busComp      = null;
SiebelPropertySet     ps                = null;

try {

    Sieb_dataBean = new SiebelDataBean();
    ...
    Sieb_busObject = Sieb_dataBean.getBusObject("Account");
    Sieb_busComp = Sieb_busObject.getBusComp("Account");
    ps = Sieb_dataBean.newPropertySet();
```

```
with(ps) {

    setProperty("Name", "Frank Williams Inc");
    setProperty("Location", "10 Main St");
    setProperty("Account Status", "Active");
    setProperty("Type", "Customer");

}

Sieb_busComp.activateField ("Name");
Sieb_busComp.activateField ("Location");
Sieb_busComp.activateField ("Account Status");
Sieb_busComp.activateField ("Type");

Sieb_busComp.newRecord(true);
Sieb_busComp.setMultipleFieldValues(ps);
Sieb_busComp.writeRecord();

}

catch (SiebelException e) {

    system.out.println("Error : " + e.getErrorMessage());

}

ps.release();
Sieb_busComp.release();
Sieb_busObject.release();
Sieb_dataBean.release();
```

Related Topics

For more information, see the following topics:

- ■ "ActivateMultipleFields Method for a Business Component" on page 188

- ■ "GetMultipleFieldValues Method for a Business Component" on page 209

SetNamedSearch Method for a Business Component

The SetNamedSearch method sets the named search specification on a business component. This method does not return any information.

Format

*BusComp.*SetNamedSearch *searchName*, *searchSpec*

Table 68 describes the arguments for the SetNamedSearch method.

Table 68.   Arguments for the SetNamedSearch Method

Argument	Description
searchName	String that identifies the name of the search specification.
searchSpec	String that contains the search specification.

The searchSpec argument works in the same way as the argument you use after the equal sign in a predefined query. For more information, see “SetSearchExpr Method for a Business Component” on page 236 and “SetSearchSpec Method for a Business Component” on page 238.

Usage

A *named search specification* is a type of search specification that Siebel CRM applies in conjunction with the existing search specification. It applies the named search specification every time it calls the ExecuteQuery method. For example, with a predefined query or with the search specification on a business component.

You can only modify a named search specification programmatically. You cannot use the administrative interface to modify a named search specification.

The ClearToQuery method does not clear the named search specification. To clear it, you must explicitly set the searchSpec argument to "". If Siebel CRM creates a new instance of a business component, then it clears the named search specification.

Using the SetNamedSearch method to define a search does not create a predefined query. You specify this search only in script. To return this search specification, you can use the GetNamedSearch method. To return the values of an attribute in a user profile, Personalization uses the GetProfileAttr method.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The examples in this topic set a named search specification for a business component depending on the position of the current user.

The following example is in Siebel eScript:

```
function BusComp_PreQuery ()
{
    if (TheApplication().GetProfileAttr("Position") == "Siebel Administrator");
    {
        this.SetNamedSearch ("Candidates", "[Status] LIKE 'Candidate'")
    }
```

```
    return (ContinueOperation);
}
```

The following example is in Siebel VB:

```
Function BusComp_PreQuery () As Integer
    If TheApplication.GetProfileAttr("Position") = "Siebel Administrator" Then
        Me.SetNamedSearch "Candidates", "[Status] LIKE 'Candidate'"
End If

BusComp_PreQuery = ContinueOperation
End Function
```

SetSearchExpr Method for a Business Component

The SetSearchExpr method sets a search expression for a business component. This method does not return any information.

Format

*BusComp.*SetSearchExpr *searchSpec*

Table 69 describes the arguments for the SetSearchExpr method.

Table 69.   Arguments for the SetSearchExpr Method

Argument	Description
searchSpec	String that identifies the search specification.

Usage

You can call the SetSearchExpr method after you call the ClearToQuery method and before you call the ExecuteQuery method. It is not necessary to use the ActivateField method on a field that you specify in the SetSearchExpr method.

The maximum length of a predefined query is 2000 characters.

The searchSpec argument works in the same way as the argument you use after the equal sign in a predefined query. For example, consider the following predefined query:

```
'Account'.Search = "[Name] ~ LIKE ""A. C. Parker"" "
```

You can use the following equivalent search specification in various interface methods:

```
BC.SetSearchExpr "[Name] ~ LIKE ""A. C. Parker"" "
```

In this example, Name is a field in a business component. You must enclose it in square brackets, [ ].

To create a query that includes a sort specification, use the SetSortSpec method. You cannot use the SetSearchExpr method to set a sort specification. Do not use the SetSearchExpr method and the SetSearchSpec method together. These methods are mutually exclusive.

Any date you use with the SetSearchExpr method must use the MM/DD/YYYY format, regardless of the Regional control panel settings on the Siebel Server or the Siebel client.

Using the SetSearchExpr Method with a Keyword

If a field value contains a search keyword, then you must use two pairs of double quotes around the field value. Example keywords include NOT, AND, or OR. For example, if the Sub-Status field includes the string Not an Issue as a field value, then you must use the following Siebel VB format to avoid an SQL error:

```
substatus = GetFieldValue("Sub-Status")
searchst = "[Value] = """ & substatus & """"
BC.SetSearchExpr searchst
```

The following Siebel VB format creates an SQL error:

```
substatus = GetFieldValue("Sub-Status")
searchst = "[Value] = " & substatus
BC.SetSearchExpr searchst
```

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example in Siebel eScript demonstrates how to log the current search specification to a file:

```
var Ob = TheApplication().ActiveBusObject();
var BC = Ob.GetBusComp("Opportunity");
var Account = "Turston Steel";
var Oppty = "CAD/CAM implementation";
var searchst = "[Name] = '" + Oppty + "' AND [Account] = '" + Account + "'";

TheApplication().TraceOn("c:\\temp\\trace.txt", "Allocation", "All");
TheApplication().Trace("the search expression is: " + searchst);
BC.ClearToQuery();
BC.SetSearchExpr(searchst);
BC.ExecuteQuery(ForwardBackward);
```

Related Topics

For more information, see the following topics:

- "ClearToQuery Method for a Business Component" on page 193

- "ExecuteQuery Method for a Business Component" on page 197

- "SetSearchSpec Method for a Business Component" on page 238

- "SetSortSpec Method for a Business Component" on page 243

SetSearchSpec Method for a Business Component

The SetSearchSpec method sets the search specification for a business component. This method does not return any information.

CAUTION: Do not use the SetSearchExpr method and the SetSearchSpec method together. They are mutually exclusive.

Format

BusComp.SetSearchSpec *FieldName*, *searchSpec*

Table 70 describes the arguments for the SetSearchSpec method.

Table 70.   Arguments for the SetSearchSpec Method

Argument	Description
FieldName	String that identifies the name of the field where Siebel CRM sets the search specification.
searchSpec	String that contains the search specification.

Usage

You must call the SetSearchSpec method before you call the ExecuteQuery method.

To avoid an unexpected compound search specification on a business component, it is recommended that you call the ClearToQuery method before you call the SetSearchSpec method. It is not necessary to use the ActivateField method on a field that you reference in the SetSearchSpec method.

Making Multiple Calls to the SetSearchSpec Method

If you instruct Siebel CRM to make multiple calls to the SetSearchSpec method for a business component, then it handles the multiple search specifications in the following ways:

■   If the existing search specification is on the same field as the new search specification, then Siebel CRM replaces the existing search specification with the new search specification. For example, consider the following code:

```
myBusComp.SetSearchSpec("Status", "<> 'Renewal'");
myBusComp.SetSearchSpec("Status", "<> 'Dropped'");
```

This code results in the following WHERE clause:

```
WHERE Status <> 'Dropped'
```

■   If the existing search specification is not on the same field as the new search specification, then Siebel CRM creates a search specification that is a logical AND of the existing and the new search specifications. For example:

```
myBusComp.SetSearchSpec("Type", "<> 'Renewal'");
myBusComp.SetSearchSpec("Status", "<> 'Sold' AND [Status] <> 'Cancelled' AND
[Status] <> 'Renewed'");
```

This code results in the following WHERE clause:

```
WHERE Type <> 'Renewal' AND (Status<> 'Sold' AND Status <> 'Cancelled' AND Status
<> 'Renewed')
```

■ If the existing search specification includes one or more of the same fields as the new search specification, then Siebel CRM replaces only that part of the existing search specification that includes fields that the new search specification also includes. For example:

```
myBusComp.SetSearchSpec("Status", "<> 'In Progress'")
```

This code results in the following WHERE clause:

```
WHERE Type <> 'Renewal' AND Status <> 'In Progress'
```

Siebel CRM replaces the search specification only on the Status field.

Combining Declarative and Scripted Search Specifications

If you define a search specification declaratively in Siebel Tools, and if you use the SetSearchSpec method to define another search specification in script, then Siebel CRM creates a search specification that is a logical AND of the declarative search specification and the scripted search specification. For example, consider the following scripted search specification:

```
myBusComp.SetSearchSpec("Status", "<>  'Cancelled'")
```

Consider the following declarative search specification:

```
[Type] <> 'Renewal' AND [Status] <> 'Sold'
```

When Siebel CRM creates a logical AND between these search specifications, the following WHERE clause results:

```
WHERE Type <> 'Renewal' AND (Status <> 'Sold' AND Status <> 'Cancelled')
```

Using Logical and Comparison Operators in a Search Specification

You can use logical operators and comparison operators. Consider the following example, in Siebel VB:

```
BC.SetSearchSpec "Status", "<> 'Closed' AND ([Owner] = LoginName () OR [Refer To] =
LoginName ()) OR ([Owner] IS NULL AND [Support Group] = 'TS-AE')"
```

Using Special Characters in a Search Specification

The search specification can contain any of the following special characters:

■ " (double quote)

■ ' (single quote)

■ = (equal sign)

- ■ > (greater than symbol)

- ■ < (less than symbol)

- ■ ((opening parenthesis)

- ■) (closing parenthesis)

- ■ [(opening square bracket)

- ■ [(closing square bracket)

- ■ , (comma)

- ■ ~ (tilde)

You must enclose each of these special characters in quotes. This rule applies to operators that are part of the search expression and to the search text.

Using Quotes and Other Characters in a Search Specification

If the search expression contains quotes or another special character, then you must enclose the entire search specification in double quotes. An apostrophe is an example of a special character.

If the search object includes a special character, then you must double that character. For example, assume your specification must search for text that contains a single double quote:

"We must

In this situation, you must do the following work:

1 Use two double quotes before the word We:

""We must

2 Enclose the string you created in Step 1 with single quotes:

'""We must'

3 Enclose the entire expression in double quotes:

"'""We must' "

4 Add the expression to the search specification:

SetSearchSpec "Comments", "'""We must'"

In another example, assume your search specification must search for the following text in the Name field:

Phillie's Cheese Steaks

In this situation, you must use the following search specification:

SetSearchSpec "Name", "'Phillie''s Cheese Steaks'"

Using Quotes and Other Characters in a Search Specification in Siebel eScript or Browser Script

To mark a special character in Siebel eScript or Browser Script, you must use a backslash instead of a double quote. For example:

■ To include double quotes before the word We, you must use the following format:

```
SetSearchSpec("Comments", "\'\"We must\'")
```

■ To include the string Phillie's Cheese Steaks, you must use the following format:

```
SetSearchSpec("Name", "\'Phillie\'\'s Cheese Steaks\'")
```

For more information, see "Using Quotes and Other Characters in a Search Specification" on page 240.

Using a Search Specification to Search Text in a Nontext Field

If any of the following situations are true, then you must use double quotes to enclose the text you use in a search specification:

■ The search expression queries a field of any type other than a text field.

■ The search expression includes any character that is not included in the following list:

   ■ Any upper-case letter of the alphabet. For example:

   ABCDEFGHIJKLMNOPQRSTUVWXYZ

   ■ Any lower-case letter of the alphabet. For example:

   abcdefghijklmnopqrstuvwxyz

   ■ Any of the following special characters:

      ❏ underscore (_)

      ❏ question mark (?)

      ❏ back slash (\)

      ❏ double quote (")

      ❏ single quote (')

      ❏ opening bracket ([)

      ❏ closing bracket (])

Using a Search Specification to Return All Records

To return all records, use the ClearToQuery method and then the ExecuteQuery method. Do not use the SetSearchSpec method. For more information, see "ClearToQuery Method for a Business Component" on page 193 and "ExecuteQuery Method for a Business Component" on page 197.

Using a Search Specification to Search for a Null Field

To search for a null field, use the following form:

```
SetSearchSpec "Account", "is NULL"
```

If your search specification requests an empty string, then the search returns every record. For example:

```
SetSearchSpec "Account", ""
```

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following Siebel VB code searches for a contact by name, and then navigates to a view that displays this record:

```
(general)
(declarations)
Option Explicit

Sub Button1_Click
Dim theCurrComp As BusComp
Dim TargetView As String
Dim TargetBusObj As String
Dim TargetBusComp As String
Dim NewBusObj As BusObject
Dim NewComp As BusComp
Dim RecId1 As String
Dim RecId2 As String
Dim RecId3 As String

TargetView = "Visible Contact List View"
TargetBusObj = "Contact"
TargetBusComp = "Contact"
Set theCurrComp = Me.BusComp
RecId1 = theCurrComp.GetFieldValue("Last Name")
RecId2 = theCurrComp.GetFieldValue("First Name")
RecId3 = theCurrComp.GetFieldValue("Account Id")
Set NewBusObj = TheApplication.GetBusObject(TargetBusObj)
Set NewComp = NewBusObj.GetBusComp(TargetBusComp)
NewComp.ClearToQuery
NewComp.SetSearchSpec "Last Name", RecId1
NewComp.SetSearchSpec "First Name", RecId2
NewComp.SetSearchSpec "Account Id", RecId3
NewComp.ExecuteQuery ForwardBackward

TheApplication.GotoView TargetView , NewBusObj

Set NewComp = Nothing
Set NewBusObj = Nothing
Set theCurrComp = Nothing

End Sub
```

For other Siebel VB examples, see "FirstRecord Method for a Business Component" on page 200, "SetFormattedFieldValue Method for a Business Component" on page 230, and "BusComp_PreQuery Event" on page 271.

The following example is in Siebel eScript:

```
var oAccntBO = TheApplication().GetBusObject("Account");
var oAccntBC = oAccntBO.GetBusComp("Account");
var oAddrBC;

with (oAccntBC)
{
    SetViewMode(SalesRepView);
    ClearToQuery();
    SetSearchSpec("Name", "Hong Kong Flower Shop");
    ExecuteQuery(ForwardBackward);

    if (FirstRecord())

    oAddrBC = GetMVGBusComp("Street Address");
    with (oAddrBC)
    {
        NewRecord(NewAfter);
        SetFieldValue("City", "Denver");
        SetFieldValue("Street Address", "123 Main Street");
        WriteRecord();
    }

}

oAddrBC = null;
oAccntBC = null;
oAccntBO = null;
```

For another Siebel eScript example, see "ClearToQuery Method for a Business Component" on page 193.

SetSortSpec Method for a Business Component

The SetSortSpec method sets the sort specification for a business component. This method does not return any information.

Format

*BusComp.*SetSortSpec *sortSpec*

Table 71 describes the arguments for the SetSortSpec method.

Table 71.   Arguments for the SetSortSpec Method

Argument	Description
sortSpec	String that contains the sort specification.

The sortSpec argument uses the following format:

"*fieldName1, fieldName2, . . .* (ASCENDING)"

or

"*fieldName1, fieldName2, . . .* (DESCENDING)"

You must enclose the entire string in double quotes. To sort on various fields in different orders, you can use a comma to separate field names and order specifications.

Usage

If you use the SetSortSpec method, then you must call it after you call the ClearToQuery method and before you call the ExecuteQuery method.

The SortSpec argument works in the same way as the equal sign in a predefined query. For example, consider the following predefined query:

```
'Account'.Sort = "Name(ASCENDING)"
```

You can use the following equivalent search specification in various interface methods:

```
BC.SetSortSpec "Name(ASCENDING)"
```

Note that Name is the value in the Name property of the business component field. This example queries the Name field.

Any date you use with the SetSortSpec method must use the MM/DD/YYYY format, regardless of the Regional control panel settings of the Siebel Server or Siebel client.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The Siebel VB example in this topic sorts the Opportunity list first by Account in reverse order, and then in alphabetical order by Site. Note that the column names in the Opportunity list applet are not the same as the names in the underlying business component.

For demonstration purposes, this example sorts in ascending and descending order. In actual practice, do not sort in two directions in a single sort specification because this type of sorting can significantly degrade performance.

```
Function BusComp_PreQuery As Integer

With Me
   .ActivateField("Account")
   .ActivateField("Account Location")
   .ClearToQuery
   .SetSortSpec "Account(DESCENDING), Account Location(ASCENDING)"
   .ExecuteQuery ForwardBackward
End With
```

```
    End Function
```

The following is the equivalent example in Siebel eScript:

```
function BusComp_PreQuery {

    with (this)
    {
        ActivateField("Account");
        ActivateField("Account Location");
        ClearToQuery();
        SetSortSpec("Account(DESCENDING), Account Location(ASCENDING)");
        ExecuteQuery(ForwardBackward);
    }

}
```

Related Topics

For more information, see the following topics:

■ "GetSortSpec Method for a Business Component" on page 216

■ "SetSearchExpr Method for a Business Component" on page 236

■ "SetSearchSpec Method for a Business Component" on page 238

SetUserProperty Method for a Business Component

The SetUserProperty method sets the value of a user property in a business component. A user property is similar to an instance variable of a business component. This method does not return any information.

Format

BusComp.SetUserProperty *propertyName*, *newValue*

Table 72 describes the arguments for the SetUserProperty method.

Table 72.   Arguments for the SetUserProperty Method

Argument	Description
propertyName	String that identifies the name of the user property.
newValue	String that contains the new value.

Usage

Usage for the SetUserProperty method is similar to the usage for the GetUserProperty method. For more information, see "Usage for the GetUserProperty Method" on page 217.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel VB:

```
Sub BusComp_SetFieldValue (FieldName As String)
    Select Case FieldName
        Case "Committed"
            me.SetUserProperty "Flagged", "Y"
    End Select
End Sub
```

The following is the equivalent example in Siebel eScript:

```
function BusComp_SetFieldValue (FieldName)
{
    switch (FieldName)
    {
        case "Committed":
            this.SetUserProperty("Flagged", "Y");
    }
}
```

SetViewMode Method for a Business Component

The SetViewMode method sets the visibility type for a business component. This method does not return any information. For more information, see "GetViewMode Method for a Business Component" on page 217.

Format

*BusComp.*SetViewMode *mode*

Table 73 describes the arguments for the SetViewMode method.

Table 73.   Arguments for the SetViewMode Method

Argument	Description
mode	A Siebel ViewMode constant or the corresponding integer value for the constant. For more information, see "Constants You Can Use with the SetViewMode Method" on page 247.

Siebel ViewMode constants correspond to applet visibility types. For more information about applet visibility types, see *Siebel Security Guide.*

Constants You Can Use with the SetViewMode Method

Table 74 describes the constants you can use with the SetViewMode method. The Owner Type column indicates the value that must be set in the Owner Type property of the BusComp view mode object of the business component. For more information, see "Use Constants to Standardize Code" on page 68.

Table 74.   Constants for the SetViewMode Method

Siebel ViewMode Constant	Integer Value	Owner Type	Description

SalesRepView	0	Position	This constant does the following:
			
			■ Applies access control according to a single position or a sales team.
			
			■ Displays records according to one of the following items:
			
			  ■ The user position.
			
			  ■ The sales team that includes the user position. The Visibility field or Visibility MVField of the business component determines the visibility.

ManagerView	1	Position	Displays records that the user and the users who report to the user can access. For example, the records that Siebel CRM displays in the My Team's Accounts visibility filter. If the business component that the view references uses single position access control, then this constant displays records that Siebel CRM associates directly with the active position of the user and with subordinate positions. If the business component that the view references uses sales team access control, then this constant displays records according to one of the following positions: ■ The primary position for the user on a team. ■ A subordinate position that is the primary member on a team. If the user position does not include a subordinate position, then Siebel CRM does not display any records.

Table 74.   Constants for the SetViewMode Method

Siebel ViewMode Constant	Integer Value	Owner Type	Description
PersonalView	2	Position	Displays records that the user can access, as determined by the Visibility Field property of the BusComp view mode object. For example, the records that Siebel CRM displays in the My Accounts visibility filter.
AllView	3	Not applicable	Displays all records that includes valid owner. For example, the records that Siebel CRM displays in the All Accounts Across Organizations visibility filter.
OrganizationView	5	Position	Displays records where a valid owner is associated with the record and the user position is associated with the organization. For example, the records that Siebel CRM displays in the All Accounts List View visibility filter. Applies access control for a single organization or for multiple organizations, as determined by the Visibility field or Visibility MVField of the BusComp view mode object of the business component.
GroupView	7	Not applicable	This constant does one of the following: ■ Displays a list of the subcategories that the user can access. ■ Displays records in the current category, depending on the current applet. If the user is at the catalog level, then Siebel CRM displays the first level categories.

Table 74.   Constants for the SetViewMode Method

Siebel ViewMode Constant	Integer Value	Owner Type	Description
CatalogView	8	Catalog Category	Displays a list of records in categories across every catalog that the user can access. Siebel CRM typically uses this visibility in a product picklist and other list of products, such as a recommended product list.
SubOrganizationView	9	Organization	If the business component that the view references uses single organization access control, then this constant displays records that Siebel CRM associates directly with one of the following organizations: ■ The organization that is currently active for the user. ■ A descendent organization. This descendent organization is part of the organization hierarchy. For example, the records that Siebel CRM displays in the All Opportunities Across My Organization visibility filter. If the business component that the view references uses multiple organization access control, then this constant displays records for the primary active organization or for the primary descendent organization.

Used With

COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The following example is in Siebel VB. For another example, see “BusComp_PreDeleteRecord Event” on page 268:

```
(general)
(declarations)
Option Explicit
Dim oBO as BusObject
Dim oBC as BusComp

Set oBO = TheApplication.GetBusObject(Me.BusObject.Name)
Set oBC = oBO.GetBusComp(Me.Name)
With oBC
    .SetViewMode SalesRepView
```

```
        .ClearToQuery
        .SetSearchSpec "Name", Me.GetFieldValue("Name")
        .SetSearchSpec "Id", "<> " & Me.GetFieldValue("Id")
        .ExecuteQuery ForwardOnly
        If .FirstRecord Then
            TheApplication.Trace"Entry for name " & Me.GetFieldValue("Name") & " exists."
        End If
    End With

    Set oBC = Nothing
    Set oBO = Nothing
```

The following is the equivalent example in Siebel eScript:

```
    var oBO = TheApplication().GetBusObject(this.BusObject().Name());
    var oBC = oBO.GetBusComp(this.Name);

    TheApplication().TraceOn("c:\\trace.txt","Allocation","All");
    with (oBC)
    {
        SetViewMode(SalesRepView);
        ClearToQuery();
        SetSearchSpec("Name", this.GetFieldValue("Name"));
        SetSearchSpec("Id", "<> " + this.GetFieldValue("Id"));
        ExecuteQuery(ForwardOnly);
        if (FirstRecord())
            TheApplication().Trace("Entry for name " + this.GetFieldValue("Name") + "
    exists.");
    }

    TheApplication().TraceOff();
    oBC = null;
    oBO = null;
```

UndoRecord Method for a Business Component

The UndoRecord method reverses any unsaved modifications that Siebel CRM has made on a record. This includes reversing unsaved modifications to fields, and deleting an active record that is not saved to the Siebel database. This method does not return any information.

Format

*BusComp.*UndoRecord

No arguments are available.

Usage

You can use the UndoRecord method in the following ways:

■ **To delete a new record.** Use it after Siebel CRM calls the NewRecord method and before it saves the new record to the Siebel database.

■ **To reverse modifications made to field values.** Use it before Siebel CRM uses the WriteRecord method to save these modifications, or before the user steps off the record.

UndoRecord reverses unsaved modifications to a record. If you require a fine degree of control over the modifications that Siebel CRM reverses, then do the following:

1 Place the code in one of the following events:

   ■ PreNewRecord

   ■ PreSetFieldValue

   ■ PreWriteRecord

2 Issue a CancelOperation to cancel the modifications that the event calls.

For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58 and "NewRecord Method for a Business Component" on page 220.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

WriteRecord Method for a Business Component

The WriteRecord method saves to the Siebel database any modifications made to the current record. This method does not return any information.

Format

*oBusComp.*WriteRecord

No arguments are available.

Usage

After creating new records and setting values for fields, you can call the WriteRecord method to save the new record to the Siebel database.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

The Siebel VB example in this topic implements the following logic: if the user sets the Sales Stage field to 02, then insert an activity:

```
(general)
(declarations)
Option Explicit
```

```
Sub BusComp_SetFieldValue (FieldName As String)
    ' Run this code from the Opportunities Activities view.
    ' Opportunity is presumed to be the parent business component.

    Select Case FieldName
        Case "Sales Stage"
        if Me.GetFieldValue(FieldName) LIKE "02*" Then
            ' reference the Action business component
            Dim oBCact as BusComp
            Set oBCact = me.BusObject.GetBusComp("Action")
            With oBCact
                .NewRecord NewAfter
                .SetFieldValue "Type", "Event"
                .SetFieldValue "Description", "THRU SVB, Stage _
                    changed to 02"
                .SetFieldValue "Done", Format(Now(), _
                    "mm/dd/yyyy hh:mm:ss")
                .SetFieldValue "Status", "Done"
                .WriteRecord
            End With
            set oBCact = Nothing
        end if
    End Select
End Sub
```

For more examples, see "GetMVGBusComp Method for a Business Component" on page 211 and "NewRecord Method for a Business Component" on page 220

Business Component Invoke Methods

This topic describes methods you can use with the InvokeMethod method. It includes the following topics:

Overview of Methods That Manipulate the File System

To manipulating the file system, you can use the following methods:

- CreateFile

- GetFile

- PutFile

You can store a file in the local file system on the Siebel Server where your configuration runs the script. You can also return this file. You can use a UNC (Universal Naming Convention) format. For example, \\server\dir\file.txt. You can use a DOS folder. For example, c:\dir\file.txt.

The Siebel Server must be able to access the UNC path or mounted file system. If you use a Java client to run the Siebel Java Data Bean, then the Siebel Server must be able to access all files.

You can use these methods with business components that use the CSSBCFile class. These methods do not serialize the files from the client of a third-party application or place files from the client of a third-party application in the Siebel file system.

ClearLOVCache Method for a Business Component

The ClearLOVCache method clears the cache for the list of values (LOV) in the object manager, In works in a way that is similar to the Clear Cache button that Siebel CRM displays in the List of Values view of the Administration - Data screen. This method does not return any information.

The ClearLOVCache method clears only the object manager cache. It does not clear the session cache in a high interactivity client.

Format
BusComp.InvokeMethod("ClearLOVCache")

No arguments are available.

Used With
To use this method, you can use a *BusComp*.InvokeMethod call with the following interfaces:

- Browser Script

- COM Data Control

- COM Data Server

- Siebel Java Data Bean

- Mobile Web Client Automation Server

- Server Script

Examples
The following Siebel eScript example is for Server Script:

```
function WebApplet_PreInvokeMethod (MethodName)

{
```

```
if (MethodName == "TestMethod") {

    var lov_bo = TheApplication().GetBusObject("List Of Values");

    var lov_bc = lov_bo.GetBusComp("List Of Values");

    lov_bc.NewRecord(NewAfter);

    lov_bc.SetFieldValue("Type", "ACCOUNT_STATUS");

    lov_bc.SetFieldValue("Name", "Hello");

    lov_bc.SetFieldValue("Value", "Hello");

    lov_bc.SetFieldValue("Order By", "12");

    lov_bc.SetFieldValue("Translate", "Y");

    lov_bc.WriteRecord();

    lov_bc.InvokeMethod("ClearLOVCache");

    lov_bc = null;

    lov_bo = null;

    return (CancelOperation);

    }

    return(ContinueOperation);

}
```

CreateFile Method for a Business Component

To create a file in the Siebel file system from an external source, you can use the CreateFile method. This method returns one of the following values:

■ **Success.** The operation succeeded.

■ **Error.** The operation did not succeed.

Format

BusComp.InvokeMethod("CreateFile", *SrcFilePath*, *KeyFieldName*, *KeepLink*)

Table 75 describes the arguments for the CreateFile method.

Table 75. Arguments for the CreateFile Method

Argument	Description
SrcFilePath	The fully qualified path to the source file on the Siebel Server or the Siebel client.
KeyFieldName	The name of the field in the business component that contains the File Name. For example, AccntFileName in the Account Attachment business component.
KeepLink	Applies to URLs. You can use one of the following values: ■ **Y.** Use this value if the link to the file is stored as an attachment. ■ **N.** Use this value if you reference the actual file. The actual file is compressed in a Siebel proprietary format. Siebel CRM uploads and stores it in that format on the Siebel File System.

Usage

Before you call the CreateFile method, call the NewRecord method to make sure Siebel CRM creates a new business component record.

Used With

To use this method, you can use a *BusComp*.InvokeMethod call with the following interfaces:

■   COM Data Control

■   COM Data Server

■   Siebel Java Data Bean

■   Mobile Web Client Automation Server

■   Server Script

Examples

The following example is in Siebel VB:

```
Dim RetValue as String
Dim fileBC as BusComp

'Instantiate fileBC as the appropriate attachment business component

fileBC.NewRecord NewAfter
RetValue = fileBC.InvokeMethod ("CreateFile", "c:\Demo\Image.bmp", "AccntFileName",
"Y")
fileBC.WriteRecord
```

The following example is in Siebel eScript:

```
var fileBC;
```

```
    // Instantiate fileBC as the appropriate attachment business component

    fileBC.NewRecord(NewAfter);
    RetValue = fileBC.InvokeMethod ("CreateFile", "C:\\Demo\\Image.bmp",
    "AccntFileName", "Y");
    fileBC.WriteRecord();
```

The following example is in COM Data Control:

```
    Dim errCode as Integer
    Dim Args(2) as String
    Dim RetValue as String
    Dim fileBC as BusComp

    'Instantiate fileBC as the appropriate attachment business component

    Args(0) = "C:\Demo\Image.bmp"
    Args(1) = "AccntFileName"
    Args(2) = "Y"

    fileBC.NewRecord NewAfter, errCode
    RetValue = fileBC.InvokeMethod ("CreateFile", Args, errCode)
    fileBC.WriteRecord
```

GenerateProposal Method for a Business Component

To create a new proposal record, the GenerateProposal method uses a template and settings from the DocServer as input. The *DocServer* is third-party software that specializes in searching, storing, and serving documents. It creates the proposal.

Format

To specify a custom template, use the following format:

*BusComp.*InvokeMethod("GenerateProposal", *RecordExists*, *Replace*, *TemplateFile*)

To use the default proposal template, use the following format:

*BusComp.*InvokeMethod("GenerateProposal", *RecordExists*, *Replace*)

Table 76 describes the arguments for the GenerateProposal method.

Table 76.　Arguments for the GenerateProposal Method

Argument	Description
RecordExists	You can use one of the following values: ■ **TRUE.** Siebel CRM uses the proposal that is currently chosen. ■ **FALSE.** Siebel CRM creates a new record.
Replace	You can use one of the following values: ■ **TRUE.** Siebel CRM copies the template file from the template to the proposal as a draft file. ■ **FALSE.** You typically set the Replace argument to FALSE.
TemplateFile	Optional. You can use one of the following values: ■ **A string that specifies the name of the template to use.** If this argument receives a string, then the proposal searches for the first template record whose name contains the string passed. ■ **NULL.** Uses the default template. This is default value.

Used With

To use this method, you can use a *BusComp*.InvokeMethod call with the following interfaces:

■ Browser Script

■ COM Data Control

■ COM Data Server

■ Siebel Java Data Bean

■ Mobile Web Client Automation Server

■ Server Script

GetFile Method for a Business Component

The GetFile method gets a file from the Siebel file system and places that file in the local file system on the Siebel Server or the Siebel client. This method returns one of the following values:

■ **Operation succeeded.** Returns a string that contains Success, *OutFilePath*.

where:

OutFilePath is the fully qualified path to the file that resides in the user temp folder on the Siebel client or on the Siebel Server.

■ **Operation failed.** Returns a string that contains Error.

Format

*BusComp.*InvokeMethod("GetFile", *KeyFieldName*)

Table 77 describes the arguments for the GetFile method.

Table 77.   Arguments for the GetFile Method

Argument	Description
KeyFieldName	The name of the business component field that contains the file name. For example, AccntFileName in the Account Attachment business component.

Usage for the GetFile Method

The record pointer must point to the record you seek. If necessary, you must query for the record ID, using the NextRecord method to advance through the returned set of records until the record pointer points to the record you seek.

Used With

To use this method, you can use a *BusComp.*InvokeMethod call with the following interfaces:

■   COM Data Control

■   COM Data Server

■   Siebel Java Data Bean

■   Mobile Web Client Automation Server

■   Server Script

Examples

The following example uses Siebel VB:

```
Dim RetValue as String
Dim fileBC as BusComp

'Instantiate fileBC as the appropriate attachment business component

'Query for the required attachment record

RetValue = fileBC.InvokeMethod ("GetFile", "AccntFileName")
```

The following example uses Siebel eScript:

```
var RetValue;
var fileBC;

// Instantiate fileBC as the appropriate attachment business component

// Query for the required attachment record

var RetValue = fileBC.InvokeMethod("GetFile", "AccntFileName");
```

The following example uses COM Data Control:

```
Dim errCode as Integer
Dim Args as String
Dim RetValue as String
Dim fileBC as BusComp

'Instantiate fileBC as the appropriate attachment business component

'Query for the required attachment record

Args = "AccntFileName"
RetValue = fileBC.InvokeMethod ("GetFile", Args, errCode)
```

PutFile Method for a Business Component

The PutFile method updates a file in the Siebel file system with a newer file. This method returns one of the following values:

■ **Success.** The operation succeeded.

■ **Error.** The operation did not succeed.

Format

*BusComp.*InvokeMethod("PutFile", *SrcFilePath*, *KeyFieldName*)

Table 78 describes the arguments for the PutFile method.

Table 78.   Arguments for the PutFile Method

Argument	Description
SrcFilePath	The fully qualified path to the file on the Siebel Server or the Siebel client.
KeyFieldName	The name of the field in the business component that identifies the file name. For example, AccntFileName in the Account Attachment business component.

Usage

Usage for the PutFile method is similar to usage for the GetFile method. For more information, see "Usage for the GetFile Method" on page 258.

After Siebel CRM uses the PutFile method to save a file attachment, you must make sure it calls the WriteRecord method so that the updated attachment is visible in the Siebel client. For more information, see "WriteRecord Method for a Business Component" on page 251.

Used With

To use this method, you can use a *BusComp*.InvokeMethod call with the following interfaces:

■ COM Data Control

■ COM Data Server

■ Siebel Java Data Bean

■ Mobile Web Client Automation Server

■ Server Script

Examples

The following example uses Siebel VB:

```
Dim RetValue as String
Dim fileBC as BusComp

'Instantiate fileBC to the appropriate attachment business component

'Query for the attachment record to be updated

RetValue = fileBC.InvokeMethod ("PutFile", "c:\Demo\Image.bmp", "AccntFileName")
fileBC.WriteRecord
```

The following example uses Siebel eScript:

```
var RetValue;
var fileBC;

// Instantiate fileBC to the appropriate attachment business component

// Query for the attachment record to be updated

RetValue = fileBC.InvokeMethod("PutFile", "c:\\Demo\\Image.bmp", "AccntFileName");
fileBC.WriteRecord();
```

The following example uses COM Data Control:

```
Dim errCode as Integer
Dim Args(1) as String
Dim RetValue as String
Dim fileBC as BusComp

'Instantiate fileBC to the appropriate attachment business component

'Query for the attachment record to be updated

Args(0) = "C:\Demo\Image.bmp"
Args(1) = "AccntFileName"
RetValue = fileBC.InvokeMethod ("PutFile", Args, errCode)
fileBC.WriteRecord
```

RefreshBusComp Method for a Business Component

The RefreshBusComp method runs the current query again for a business component and makes the record that was previously active the active record. The user can see that Siebel CRM updated the view but the same record remains highlighted in the same position in the list applet. This method does not return any information.

Format
*BusComp.*InvokeMethod("RefreshBusComp")

No arguments are available.

Used With
To use this method, you can use a *BusComp*.InvokeMethod call with the following interfaces:

■ Browser Script

■ COM Data Control

■ COM Data Server

■ Siebel Java Data Bean

■ Mobile Web Client Automation Server

■ Server Script

This method only works with a business component that uses the CSSBCBase class.

RefreshRecord Method for a Business Component

The RefreshRecord method updates the currently highlighted record, including updating business component fields in the Siebel client. It positions the cursor on the highlighted record. It does not update other records that are currently available in the Siebel client. This method does not return any information.

Format
retVal = *BusComp.*InvokeMethod("RefreshRecord")

No arguments are available.

Used With
To use this method, you can use a *BusComp*.InvokeMethod call with the following interfaces:

■ Browser Script

■ COM Data Control

■ Siebel Java Data Bean

■ Mobile Web Client Automation Server

■ Server Script

This method only works with a business component that uses the CSSBCBase class.

SetAdminMode Method for a Business Component

The SetAdminMode method can enable or disable visibility rules for a business component. It sets the Admin property of a view. This method does not return any information.

Format

*BusComp.*InvokeMethod("SetAdminMode", flag)

Table 79 describes the arguments for the SetAdminMode method.

Table 79.   Arguments for the SetAdminMode Method

Argument	Description
flag	You can use one of the following values: ■ **TRUE.** Siebel CRM calls the business component in Admin mode. ■ **FALSE.** Siebel CRM does not call the business component in Admin mode.

Used With

To use this method, you can use a *BusComp.*InvokeMethod call with the following interfaces:

■ COM Data Control

■ COM Data Server

■ Siebel Java Data Bean

■ Mobile Web Client Automation Server

■ Server Script

Business Component Events

This topic describes business component events. It includes the following topics:

- ■ "BusComp_PreQuery Event" on page 271

- ■ "BusComp_PreSetFieldValue Event" on page 271

- ■ "BusComp_PreWriteRecord Event" on page 273

- ■ "BusComp_Query Event" on page 274

- ■ "BusComp_SetFieldValue Event" on page 275

- ■ "BusComp_WriteRecord Event" on page 276

You can use these events only on the Siebel Server, except for the PreSetFieldValue event, which you can use only on the browser.

You can call an event from a data operation on a business component. You define these events for each business component. You can call an event before or after Siebel CRM performs the predefined behavior.

Monitoring Modifications That the User Makes to a Multivalue Field

To monitor modifications the user makes to a multivalue field, you must use the multivalue group business component.

If the user uses the multivalue group applet to modify a value in a multivalue field, then Siebel CRM calls the PreSetFieldValue event and the SetFieldValue event for the field. It does not call any event on the parent business component.

If the user does not use the multivalue group applet to modify a value in a multivalue field, then Siebel CRM does not start the PreSetFieldValue event or the SetFieldValue event for the field. The only time Siebel CRM starts these events is if the user updates the field in the multivalue group applet.

BusComp_Associate Event

If the user adds a business component record to create an association, then Siebel CRM calls the BusComp_Associate event. This method does not return any information.

Format

BusComp_Associate

No arguments are available.

Usage

The usage for the BusComp_Associate event is the same as the usage for the BusComp_NewRecord event. For more information, see "BusComp_NewRecord Event" on page 266.

Used With

Server Script

BusComp_ChangeRecord Event

If a business component record becomes the current record, then Siebel CRM calls the BusComp_ChangeRecord event. This method does not return any information.

Format

BusComp_ChangeRecord

No arguments are available.

Usage

Siebel CRM runs code in the ChangeRecord event handler each time the active record changes. To allow smooth scrolling in a list applet, you must avoid lengthy operations in this event handler.

Used With

Server Script

Examples

The Siebel VB example in this topic uses subprograms in the declarations section of the general section to set up an audit trail for service requests. This example uses the ChangeRecord event handler to initialize the values from the service record so that Siebel CRM can compare them with current values:

```
(general)
(declarations)
Option Explicit
Dim OldClosedDate, OldCreated, OldOwner, OldOwnerGroup
Dim OldSeverity, OldSource, OldStatus
Declare Sub CreateAuditRecord
Declare Sub InitializeOldValues

Sub CreateAuditRecord (FieldName As String, NewValue As String, OldValue As String,
ChangedText As String)

    Dim ActionBC As BusComp
    Dim CurrentBO As BusObject
    Dim theSRNumber

    Set CurrentBO = TheApplication.GetBusObject("Service Request")
    Set ActionBC = CurrentBO.GetBusComp("Action")
    theSRNumber = GetFieldValue("SR Number")

    With ActionBC
        .ActivateField "Activity SR Id"
        .ActivateField "Description"
        .ActivateField "Private"
        .ActivateField "Service request id"
        .ActivateField "Type"
        .NewRecord NewAfter
```

```
                    .SetFieldValue "Activity SR Id",        theSRNumber
                    .SetFieldValue "Description",           ChangedText
                    .SetFieldValue "Private",                "Y"
                    .SetFieldValue "Type",                  "Administration"
                    .WriteRecord
            End With
        End Sub

        Sub InitializeOldValues
            OldClosedDate = GetFieldValue("Closed Date")
            OldOwner = GetFieldValue("Owner")
            OldSeverity = GetFieldValue("Severity")
            If GetFieldValue("Severity") <> OldSeverity Then
                NewValue = GetFieldValue("Severity")
                ChangedText = "Changed Priority from " + OldSeverity + _
                    " to " + NewValue
                CreateAuditRecord "Severity", NewValue, OldSeverity, _
                    ChangedText
            End If
        End Sub

        Sub BusComp_ChangeRecord
            InitializeOldValues
        End Sub
```

BusComp_CopyRecord Event

If the user copies a business component record, and if the user makes this record the active record, then Siebel CRM calls the BusComp_CopyRecord event. This method does not return any information.

Format

BusComp_CopyRecord

No arguments are available.

Usage

If a new record is created in one of the following ways, then Siebel CRM calls the BusComp_CopyRecord method instead of the BusComp_NewRecord method:

■ Siebel CRM creates a new record through one of the following:

   ■ BusComp.NewRecord NewAfterCopy

   ■ BusComp.NewRecord NewBeforeCopy

■ A user uses a copy record feature in the Siebel Client. For example, if the user chooses the Copy Record menu item from the Edit menu, or presses CTRL+B.

Used With

Server Script

BusComp_DeleteRecord Event

If the user deletes a business component record, then Siebel CRM calls the BusComp_DeleteRecord event. The fields of the deleted record are no longer available. This method does not return any information.

Format

BusComp_DeleteRecord

No arguments are available.

Usage for the BusComp_DeleteRecord Event

Siebel CRM does not start the BusComp_PreDeleteRecord event or the BusComp_DeleteRecord event for a child record that it deletes according to the Cascade Delete property on a link. For performance reasons, Siebel CRM performs these deletes directly in the data layer. Siebel CRM calls script events from the object layer, so it does not run them.

Used With

Server Script

BusComp_InvokeMethod Event

If Siebel CRM calls the InvokeMethod method on a business component, then it also calls the BusComp_InvokeMethod event. This method does not return any information.

Format

BusComp_InvokeMethod(*methodName*)

The arguments you can use with this format are the same as the arguments described in Table 25 on page 116.

Usage

If you call a specialized method on a business component, or if you call the InvokeMethod method explicitly on a business component, then Siebel CRM calls the BusComp_InvokeMethod event. For more information, see "About Specialized and Custom Methods" on page 103.

Used With

Server Script

BusComp_NewRecord Event

If the user creates a business component record, and if the user makes this record the active record, then Siebel CRM calls the BusComp_NewRecord event. You can use this event to set up default values for a field. This method does not return any information.

Format
BusComp_NewRecord

No arguments are available.

Usage
If a new record is created in one of the following ways, then Siebel CRM calls the BusComp_CopyRecord method instead of the BusComp_NewRecord method:

- Siebel CRM creates a new record using one of the following formats:

  - BusComp.NewRecord NewAfterCopy

  - BusComp.NewRecord NewBeforeCopy

- A user uses a copy record feature in the Siebel client. For example, the user chooses the Copy Record menu item from the Edit menu, or presses CTRL+B.

Used With
Server Script

Examples
For an example, see "Pick Method for a Business Component" on page 224.

BusComp_PreAssociate Event
If Siebel CRM detects that the user is about to add a business component record to create an association, then it calls the BusComp_PreAssociate event before it adds the record. This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format
BusComp_PreAssociate

No arguments are available. The format is the same as for BusComp_PreNewRecord event. For more information, see "BusComp_PreNewRecord Event" on page 270.

Used With
Server Script

BusComp_PreCopyRecord Event
If Siebel CRM detects that the user is about to copy a business component record, then it calls the BusComp_PreCopyRecord event before it copies the record. You can use this event to perform precopy validation. This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format
BusComp_PreNewRecord

No arguments are available.

Used With
Server Script

BusComp_PreDeleteRecord Event

If Siebel CRM detects that the user is about to delete a business component record, then it calls the BusComp_PreDeleteRecord event. You can use this event to prevent the deletion or to perform any actions before Siebel CRM deletes the record.This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format
BusComp_PreDeleteRecord

No arguments are available.

Usage
Usage for the BusComp_PreDeleteRecord event is the same as usage for the BusComp_DeleteRecord event. For more information, see "Usage for the BusComp_DeleteRecord Event" on page 266.

Used With
Server Script

Examples
The following Siebel VB example prevents the deletion of an account that includes associated opportunities:

```
(general)
(declarations)
Option Explicit

Function BusComp_PreDeleteRecord As Integer
    Dim oBC as BusComp
    Dim oBO as BusObject
    Dim sAcctRowId as string

    sAcctRowId = me.GetFieldValue("Id")
    set oBO = TheApplication.GetBusObject("Opportunity")
    set oBC = oBO.GetBusComp("Opportunity")
```

```
    With oBC
        .SetViewMode AllView
        .ClearToQuery
        .SetSearchSpec "Account Id", sAcctRowId
        .ExecuteQuery ForwardOnly
        If (.FirstRecord = 1) Then
            RaiseErrorText("Opportunities exist for the Account - _
                Delete is not allowed")
        End If
    End With

    BusComp_PreDeleteRecord = ContinueOperation

    Set oBC = Nothing
    Set oBO = Nothing

End Function
```

BusComp_PreGetFieldValue Event

If a user accesses a business component field, then Siebel CRM calls the BusComp_PreGetFieldValue event. This method returns the field name and field value that exists before Siebel CRM displays the field. It also returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format

BusComp_PreGetFieldValue(*FieldName*, *FieldValue*)

Table 80 describes the arguments for the BusComp_PreGetFieldValue event.

Table 80.   Arguments for the BusComp_PreGetFieldValue Event

Argument	Description
FieldName	String that contains the name of the field that the user accessed.
FieldValue	String that contains the value of the field that the user accessed.

Usage

Siebel CRM calls the BusComp_PreGetFieldValue event in the following situations:

■ At least one time for each user interface element that displays the value for a business component field

■ Every time it updates the Siebel client

■ As a result of other internal uses

Improving Performance when Calling the BusComp_PreGetFieldValue Method

Siebel CRM runs any script that is attached to this event very frequently. It even calls empty scripts. These calls might cause a Siebel application appear to be unresponsive.

To improve performance when calling the BusComp_PreGetFieldValue method

■ Remove scripts from the BusComp_PreInvokeMethod event that you do not require:

    a In Siebel Tools, open a script you do not require.

    b Delete the entire contents of the script, including the following content:

       ❏ In Siebel VB, delete the Function statement and the End Function statement.

       ❏ In Siebel eScript, delete the function () statement and the { } function statement.

    c Repeat Step a for all other scripts you do not require.

Used With

Server Script

BusComp_PreInvokeMethod Event

If Siebel CRM calls a specialized method on a business component, then it calls the BusComp_PreInvokeMethod event before it calls this specialized method. The BusComp_PreInvokeMethod event returns ContinueOperation or CancelOperation. For more information, see "About Specialized and Custom Methods" on page 103, and "Caution About Using the Cancel Operation Event Handler" on page 58.

Format

BusComp_PreInvokeMethod(*methodName*)

The arguments you can use with this format are the same as the arguments described in Table 25 on page 116.

Used With

Server Script

BusComp_PreNewRecord Event

If Siebel CRM detects that the user is about to create a new business component record, then it calls the BusComp_PreNewRecord event before it creates the record. You can use this event to perform preinsert validation. This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format

BusComp_PreNewRecord

No arguments are available.

Used With

Server Script

BusComp_PreQuery Event

Siebel CRM calls the BusComp_PreQuery event before it runs a query. This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format

BusComp_PreQuery

No arguments are available.

Usage

To modify the search criteria or to restrict Siebel CRM from running certain queries, you can use the BusComp_PreQuery event.

Used With

Server Script

Examples

The following example is in Siebel VB:

```
Function BusComp_PreQuery() As Integer
    Dim strPosition As String
    Dim strSearchSpec As String
    Dim intReturn As Integer
    intReturn = ContinueOperation
    strPosition = TheApplication.PositionName
    strSearchSpec = Me.GetSearchSpec("Owned By")
    If strPosition <> "System Administrator" Then
        if Len(strSearchSpec) = 0 or InStr(strSearchSpec,
            strPosition) = 0 Then
            Me.SetSearchSpec "Owned By", strPosition
        end if
    End if
    BusComp_PreQuery = intReturn
End Function
```

BusComp_PreSetFieldValue Event

Siebel CRM calls the BusComp_PreSetFieldValue event in the following situations:

■ After the user modifies a field value in the Siebel client and then attempts to leave the field

■ A call to the SetFieldValue method occurs, but before it performs any field-level validation

This event allows you to use custom validation before Siebel CRM applies predefined validation. This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format

BusComp_PreSetFieldValue(*FieldName*, *FieldValue*)

The arguments you can use with this format are the same as the argument described in Table 23 on page 110.

Usage

If your script returns CancelOperation for a field, then Siebel CRM does not enter data for this field. However, Siebel CRM still starts BusComp_PreSetFieldValue for the other fields that the picklist uses to enter data. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

If a user uses a picklist to enter data for multiple fields, then it starts the BusComp_PreSetFieldValue method for each field that the user uses to enter data. For example, in an applet that the user accesses to enter data for the Last Name, First Name, and Contact ID. In this example, Siebel CRM starts the BusComp_PreSetFieldValue method three times, one time for each field.

Siebel CRM does not call the BusComp_PreSetFieldValue event on a picklist or multivalue field.

Usage With Roundtrips

Siebel CRM does the following during a roundtrip to the Siebel Server:

■ In Browser Script, if the Immediate Post Changes property of the business component field is set to TRUE, then it calls the BusComp_PreSetFieldValue method after the round trip to the Siebel Server completes.

■ In Server Script, it calls the BusComp_PreSetFieldValue method as the first event in the Siebel Server round trip.

To prevent infinite recursions, if the BusComp_PreSetFieldValue event is running, then Siebel CRM does not run it again for the same business component instance, even if Siebel CRM uses it on a different field in the business component.

Used With

Browser Script, Server Script

Examples

The following Siebel VB example uses the PreSetFieldValue event to determine if a quote discount is greater than 20 percent, and to take the appropriate action if it is. For other examples of BusComp_PreSetFieldValue, see "LoginId Method for an Application" on page 155, and "ExecuteQuery Method for a Business Component" on page 197:

```
Function BusComp_PreSetFieldValue (FieldName As String,
             FieldValue As String) As Integer
'code to check if a quote discount>20%
'if it is, notify user and cancel operation
Dim value as Integer
Dim msgtext as String
   If FieldName = "Discount" then
```

```
              value = Val(FieldValue)
              If value > 20 then
                   msgtext = "Discounts greater than 20% must be approved"
                 RaiseError msgtext
                 BusComp_PreSetFieldValue = CancelOperation
              Else
                 BusComp_PreSetFieldValue = ContinueOperation
               End if
         End If
         End Function
```

The following is the equivalent example in Siebel eScript:

```
    function BusComp_PreSetFieldValue (FieldName, FieldValue)
    {
        var msgtext = "Discounts greater than 20% must be approved";
        if (FieldName == "Discount")
        {
            if (FieldValue > 20)
            {
                TheApplication().RaiseErrorText(msgtext);
            }
            else
            {
                return (ContinueOperation);
            }
        }
        else
        {
            return (ContinueOperation);
        }
    }
```

BusComp_PreWriteRecord Event

Siebel CRM calls the BusComp_PreWriteRecord event before it writes a record to the Siebel database. This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Format

BusComp_PreWriteRecord

No arguments are available.

Usage

For important caution information, see "Caution for Using an Error Method with a Write Record Event" on page 277.

You can use this event to perform any final validation before Siebel CRM performs any predefined internal record-level validation.

Siebel CRM starts the BusComp_PreWriteRecord event only if the user modifies or inserts a field value, or if the user deletes a record. If the user deletes a record, then Siebel CRM calls the BusComp_PreWriteRecord method to delete the implied join that joins any records to the initial record.

Using a Write Record Event with a Multivalue Group

If Siebel CRM associates a multivalue group record that uses a many to many relationship with the business component that calls the association, then it starts the BusComp_PreWriteRecord event and the BusComp_WriteRecord event. It starts these events even if the association does not update any fields in the multivalue group business component or in the calling business component. It runs the BusComp_PreWriteRecord event and the BusComp_WriteRecord event to acknowledge the update to the intersection table.

Used With

Server Script

Examples

The following example calls the BusComp_PreWriteRecord event:

```
Function BusComp_PreWriteRecord As Integer

    ' This code resets the probability before the write
    ' if necessary

    if Me.GetFieldValue("Sales Stage") LIKE "07*" then
        ' Resets the Probability to 75 if less than 75
        if Val(Me.GetFieldValue("Rep %")) < 75 then
            Me.SetFieldValue "Rep %", "75"
        end If
    end if

    BusComp_PreWriteRecord = ContinueOperation
End Function
```

BusComp_Query Event

Siebel CRM calls the BusComp_Query event after it completes a query but before it displays the query results. This event does not return any information.

Format

BusComp_Query

No arguments are available.

Used With

Server Script

Examples

In the following Siebel VB example, the Action business component uses a special activity type. If the user starts an account query, then this code determines if important information is available. If it is available, then Siebel CRM displays it in a message box:

```
Sub BusComp_Query

    Dim oBusObj As BusObject, oCurrFinAct As BusComp,
    Dim oActivities as BusComp, oAppl as Applet
    Dim sName as String, sDescription as String

    On error goto leave

    set oBusObj = TheApplication.ActiveBusObject
    Set oCurrFinAct = TheApplication.ActiveBusComp

    If oCurrFinAct.FirstRecord <> 0 then
        sName = oCurrFinAct.GetFieldValue("Name")
        Set oActivities = oBusObj.GetBusComp("Finance _
            Important Info Activity")
        With oActivities
            .ActivateField("Description")
            .ClearToQuery
            .SetSearchSpec "Account Name", sName
            .SetSearchSpec "Type", "Important Info"
            .ExecuteQuery ForwardOnly
            If .FirstRecord <> 0 then
                sDescription = .GetFieldValue("Description")
                TheApplication.Trace("Important Information: " + sDescription)
                do while .NextRecord <> 0
                    sDescription = .GetFieldValue("Description")
                    TheApplication.Trace("Important Information: " + sDescription)
                loop
            End If
        End With
    End If

leave:

    Set oCurrFinAct = Nothing
    set oBusObj = Nothing

End Sub
```

BusComp_SetFieldValue Event

If Siebel CRM sends a value to a business component from the Siebel client or through a call to the SetFieldValue method, then it calls the BusComp_SetFieldValue event. It does not call this event for a predefaulted field or for a calculated field. This event does not return any information.

Format

BusComp_SetFieldValue(*FieldName*)

The arguments you can use in this format are the same as the arguments that are described in Table 55 on page 187.

Used With

Server Script

Examples

In the following Siebel VB example, if Siebel CRM calls the SetFieldValue event, then it calls methods on an existing business component:

```
Sub BusComp_SetFieldValue (FieldName As String)
Dim desc As String
Dim newDesc As String
If FieldName = "Type" Then
    newDesc = [can be any valid string that contains the new description]
    desc = GetFieldValue("Description")
    SetFieldValue "Description", newDesc
End If
End Sub
```

The following is the equivalent example in Siebel eScript:

```
function BusComp_SetFieldValue (FieldName)
{
    if (FieldName == "Type" && GetFieldValue(FieldName) == "Account")
    {
        SetFieldValue("Description", "Record is of Type 'Account'." );
    }
}
```

BusComp_WriteRecord Event

Siebel CRM starts the BusComp_WriteRecord event after it saves the record to the Siebel database. This event does not return any information.

Format

BusComp_WriteRecord

No arguments are available.

Usage

Do not use the BusComp_SetFieldValue event in a BusComp_WriteRecord event. If you must use the BusComp_SetFieldValue event, then use it in the BusComp_PreWriteRecord event. For more information, see "BusComp_PreWriteRecord Event" on page 273.

For information about using the BusComp_WriteRecord event with a multivalue group, see "Using a Write Record Event with a Multivalue Group" on page 274.

Caution for Using an Error Method with a Write Record Event

CAUTION: Be careful if you use the RaiseError method or the RaiseErrorText method in the BusComp_WriteRecord event or in the BusComp_PreWriteRecord event. For example, if you use the RaiseErrorText method in the BusComp_PreWriteRecord method, then the user or the code cannot step off the current record until the condition that causes Siebel CRM to call the RaiseErrorText method is addressed.

Used With

Server Script

Business Object Methods

This topic describes business object methods. It includes the following topics:

■ "GetBusComp Method for a Business Object" on page 277

■ "GetLastErrCode Method for a Business Object" on page 278

■ "GetLastErrText Method for a Business Object" on page 279

■ "Name Method for a Business Object" on page 279

■ "Release Method for a Business Object" on page 279

In this topic, the term oBusObj indicates a variable that contains a BusObject.

GetBusComp Method for a Business Object

The GetBusComp method returns the name of a business component instance. If an instance of the business component that the BusCompName argument specifies:

■ Exists, then the GetBusComp method returns the name of that instance.

■ Does not exist, then the interpreter starts a new business component instance, and then the GetBusComp method returns the name of this instance.

Format

*oBusObj.*GetBusComp (*BusCompName*)

Table 81 describes the arguments for the GetBusComp method.

Table 81.   Arguments for the GetBusComp Method

Argument	Description
BusCompName	String that contains the name of a business component.

The BusCompName argument is case-sensitive. It must match the case of the name that Siebel Tools displays in the Name property of the business component.

Usage

If a business component instance exists but you must create a new instance, then you can do the following:

1 Use the GetBusObject method to create a new business object instance.

2 For this new business instance, use the GetBusComp method to create a new business component.

These steps create a new business component instance that is different from the business component instance that already exists.

If you use a business object instance that already exists, then your configuration includes any other business components that reference that business object instance, even if you use the GetBusComp method.

If you no longer require the business component instance, then use one of the following keywords:

■ In Siebel VB, use Nothing ().

■ In Siebel eScript or Browser Script, use null ().

In Browser Script, the GetBusComp method can only access business component instances in the current view. In Server Script, the GetBusComp method can access every business component instance that exists in the active business object.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Server Script

Examples

The following examples are in Siebel eScript:

■ To access a business component in a UI context:

```
var ActiveBO = TheApplication().ActiveBusObject();
var ConBC = ActiveBO.GetBusComp("Contact");
```

■ To access a business component in a nonUI context:

```
var BO = TheApplication().GetBusObject("Account");
var ConBC = BO.GetBusComp("Contact");
```

GetLastErrCode Method for a Business Object

The GetLastErrCode method returns the error code for the error that Siebel CRM logged most recently. This code is a short integer. 0 (zero) indicates no error.

Format

oBusObj.GetLastErrCode

No arguments are available.

Usage

For more information, see "Usage for the GetLastErrCode Method" on page 139.

Used With

COM Data Control, Mobile Web Client Automation Server

GetLastErrText Method for a Business Object

The GetLastErrText method returns a string that contains the text message for the error that Siebel CRM logged most recently.

Format

*oBusObj.*GetLastErrText

No arguments are available.

Usage

For more information, see "Usage for the GetLastErrText Method" on page 139.

Used With

COM Data Control, Mobile Web Client Automation Server

Name Method for a Business Object

The Name method returns a string that contains the name of a business object.

Format

*oBusObj.*Name

No arguments are available.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

For an example, see "Name Method for a Business Component" on page 220.

Release Method for a Business Object

The Release method for a business object releases a business object and the resources for this business object on the Siebel Server. This method does not return any information.

Format

oBusObj.release()

No arguments are available.

Used With

Siebel Java Data Bean

Examples

The following example is for Siebel Java Data Bean:

```
import com.siebel.data.*;

{
...

// create Siebel Java Data Bean
SiebelDataBean Sieb_dataBean = null;
Sieb_dataBean = new SiebelDataBean();


// log in to Siebel Java Data Bean

...

// Create Siebel Bus Object.
// get the Bus Object from SiebelDataBean
SiebelBusObject busObj = null;
busObj = Sieb_dataBean.getBusObject("Account");


...

// Use the business Object
// Release the business object resources

...

busObj.release();
}
```

Business Service Methods

This topic describes business service methods. It includes the following topics:

- ■ "InvokeMethod Method for a Business Service" on page 284

- ■ "Name Method for a Business Service" on page 285

- ■ "PropertyExists Method for a Business Service" on page 285

- ■ "Release Method for a Business Service" on page 286

- ■ "RemoveProperty Method for a Business Service" on page 287

- ■ "SetProperty Method for a Business Service" on page 288

In this topic, the oService variable identifies a business service instance.

GetFirstProperty Method for a Business Service

The GetFirstProperty method returns a string that contains the name of the first property that is defined for a business service.

Format

*oService.*GetFirstProperty()

No arguments are available.

Usage for a Method that Gets a Business Service Property

The order that Siebel CRM uses to store properties in a property set is random. For example, the Name property is the first property that Siebel Tools displays in the Business Services list for every business service. However, the GetFirstProperty method might return any business service property, not necessarily the Name property. To correct this situation it is recommended that you add the properties in a property set to an array, and then sort that array.

To get or modify a property value, you can do the following:

1 Use the GetFirstProperty method or GetNextProperty method to return the name of a property.

2 Use the name you returned in Step 1 in one of the following ways:

- ■ To return a property value, as an argument in the GetProperty method.

- ■ To set a property value, as an argument in the SetProperty method.

For more information, see the following topics:

- ■ "GetNextProperty Method for a Business Service" on page 282

- ■ "GetProperty Method for a Business Service" on page 283

- ■ "SetProperty Method for a Business Service" on page 288

Used With

Browser Script, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Example of Using Methods that Return a Business Service Property

The example in this topic returns the number of property sets that belong to a business service.

The following example is in Siebel eScript:

```
function countPropSets(busService)
{
    var propSetName = busService.GetFirstProperty();
    var count = 0;

    while(propSetName != "")
    {
        count++;
        propSetName = busService.GetNextProperty();
    }

    return count;
}
```

The following example is for Siebel Java Data Bean:

```
public int countPropSets(SiebelService busService)
{
    int count = 0;
    try
    {
        String propSetName = busService.getFirstProperty();
        while(propSetName != "")
        {
            count++;
            propSetName = busService.getNextProperty();
        }

    }

     catch(SiebelException sExcept)
    {
        return 0;
    }

    return count;
}
```

GetNextProperty Method for a Business Service

The GetNextProperty method returns a string that contains the name of the next property of a business service. If no more properties exist, then this method returns an empty string.

Format

*oService.*GetNextProperty()

No arguments are available.

Usage for the GetNextProperty Method

After you call the GetFirstProperty method to return the name of the first property of a business service, you can call the GetNextProperty to return the name of the next property. This next property is the next property that is defined for a business service after the first property.

You can use the GetNextProperty consecutively to cycle through all the properties of a business service until no more properties exist, at which point Siebel CRM returns an empty string.

Usage for the GetNextProperty is similar to usage for the GetFirstProperty method. For more information, see "Usage for a Method that Gets a Business Service Property" on page 281.

Used With

Browser Script, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

For examples, see "Example of Using Methods that Return a Business Service Property" on page 282.

GetProperty Method for a Business Service

The GetProperty method returns a string that contains the value of a property. If the property does not exist, then this method returns NULL.

Format

*oService.*GetProperty(*propName*)

Table 82 describes the arguments for the GetProperty method.

Table 82.   Arguments for the GetProperty Method

Argument	Description
propName	A string that contains the name of the property that Siebel CRM returns.

Usage

To return the value for this property you must know the name of the property. To return a property name, use the GetFirstProperty method or the GetNextProperty method. For more information, see "Usage for a Method that Gets a Business Service Property" on page 281.

Used With

Browser Script, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

InvokeMethod Method for a Business Service

The InvokeMethod method calls a method on a business service. This method can be a specialized method or a custom method. For more information, see "About Specialized and Custom Methods" on page 103. This method does not return any information.

Siebel eScript Format

*oService.*InvokeMethod(*methodName, InputArguments, OutputArguments*)

Table 83 describes the arguments for the Siebel eScript format of the InvokeMethod method.

Table 83.   Arguments for the Siebel eScript Format of the InvokeMethod Method

Argument	Description
methodName	A string that contains the name of the method that Siebel CRM must run.
InputArguments	A property set that identifies the arguments that the method uses as input.
InputArguments	A property set that identifies the arguments that the method returns as output.

Siebel VB Format

*oService.*InvokeMethod *methodName, InputArguments, OutputArguments*

The arguments you use in this format are the same as the arguments that are described in Table 83 on page 284.

Browser Script Format

*outputPropSet=Service.*InvokeMethod(*MethodName, inputPropSet*)

The arguments you use with this format are the same as the arguments described in Table 24 on page 113.

In Browser Script, you cannot use an output property set for this format.

Usage

A predefined business service works in a way that is similar to how a call to a business component method works. You can call a specialized method on a business service that is not available directly through the object interface.

You must use this method only with Siebel VB or Siebel eScript scripts. You must use Siebel Tools to write these scripts. You can call these scripts from an external interface.

A run-time business service can include a custom method.

Although the InvokeMethod function does not return a value, the properties in the OutputArguments property set might be modified.

For more information, see "Caution About Using the InvokeMethod Method" on page 108.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Related Topics

For more information, see the following topics:

■

■

Name Method for a Business Service

The Name method returns a string that contains the name of a business service.

Format

oService.Name

No arguments are available.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

The following example is in Browser Script:

```
var svc = theApplication().GetService("Data Quality Manager"):
theApplication().SWEAlert("The active service is " + svc.Name());
```

PropertyExists Method for a Business Service

The PropertyExists method returns one of the following values to indicate if a property exists:

■ In Siebel VB, this method returns one of the following integers:

    ■ **1.** Indicates the property exists.

    ■ **0 (zero)**. Indicates the property does not exist.

■ In other interfaces, this method returns a Boolean value.

Format

oService.PropertyExists(*propName*)

Table 84 describes the arguments for the PropertyExists method.

Table 84.   Arguments for the PropertyExists Method

Argument	Description
propName	A string that contains the name of the property.

Usage
Use the PropertyExists method in an If statement to determine if a specific property is set.

Used With
Browser Script, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Release Method for a Business Service
The Release method for a business service releases a business service and the resources that this business service uses on the Siebel Server.

Format
oBusSvc.release()

No arguments are available.

Used With
Siebel Java Data Bean

Examples
The following example logs in to a Siebel Server. It then creates a business object instance, a business component instance, and a business service instance. Next, it releases them in reverse order.

```
import com.siebel.data.*;
import com.siebel.data.SiebelException;

public class JDBReleaseDemo
{
    private SiebelDataBean m_dataBean = null;
    private SiebelBusObject m_busObject = null;
    private SiebelBusComp      m_busComp = null;
    private SiebelService  m_busServ = null;

    public static void main(String[] args)
    {
        JDBReleaseDemo demo = new JDBReleaseDemo();
    }
```

```
public JDBReleaseDemo()
{
    try
    {

        // instantiate the Siebel Java Data Bean
        m_dataBean = new SiebelDataBean();

        // login to the Siebel Servers
        m_dataBean.login("siebel.tcpip.none.none://gateway:port/enterprise/
        object manager","userid","password");
        System.out.println("Logged in to the Siebel Server ");

        // get the business object
        m_busObject = m_dataBean.getBusObject("Account");

        // get the business component
        m_busComp = m_busObject.getBusComp("Account");

        // get the business service
        m_busServ = m_dataBean.getService("Workflow Process Manager");

        //release the business service
        m_busServ.release();
        System.out.println("BS released ");

        //release the business component
        m_busComp.release();

        System.out.println("BC released ");

        //release the business object
        m_busObject.release();
        System.out.println("BO released ");

        // logoff
        m_dataBean.logoff();
        System.out.println("Logged off the Siebel Server ");
    }

    catch (SiebelException e)
    {
        System.out.println(e.getErrorMessage());
    }

}

}
```

RemoveProperty Method for a Business Service

The RemoveProperty method removes a property from a business service. This method does not
return any information.

Format

oService.RemoveProperty(*propName*)

Table 85 describes the arguments for the RemoveProperty method.

Table 85.   Arguments for the RemoveProperty Method

Argument	Description
propName	A string that contains the name of the property that Siebel CRM must remove.

Usage

This method removes the property that the propName argument identifies from the business service that the oService parameter specifies. As a result, a subsequent call to the PropertyExists method for that property returns FALSE. For more information, see "PropertyExists Method for a Business Service" on page 285.

Used With

Browser Script, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

SetProperty Method for a Business Service

The SetProperty method sets a value in the property of a business service. This method does not return any information.

Format

oService.SetProperty(*propName, propValue*)

Table 86 describes the arguments for the SetProperty method.

Table 86.   Arguments for the SetProperty Method

Argument	Description
propName	A string that contains the name of the property that Siebel CRM must modify.
propValue	A string that contains the value that Siebel CRM sets in the property that the propName argument identifies.

Usage

You can use the SetProperty method to set the value of a property of a business service from one of the methods of this business service or from an external object. For more information, see "GetProperty Method for a Business Service" on page 283.

Used With

Browser Script, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script

Examples

For an example, see .

Business Service Events

This topic describes business service events. It includes the following topics:

Service_InvokeMethod Event

Siebel CRM calls the Service_InvokeMethod event after it calls the InvokeMethod method on a business service. This event does not return any information. For more information, see "Service_PreInvokeMethod Event" on page 291.

Server Script Format

Service_InvokeMethod(*MethodName, InputArguments, OutputArguments*)

The arguments you can use in this format are the same as the arguments that are described in Table 83 on page 284.

Browser Script Format

*OutputArguments=oService.*InvokeMethod(*methodName, InputArguments*)

Table 87 describes the arguments for the Browser Script format of the Service_InvokeMethod Event

Table 87.   Arguments for the Browser Script Format of the Service_InvokeMethod Event

Argument	Description
methodName	A string that contains the name of the method that Siebel CRM must run.
InputArguments	A property set that identifies the arguments that the method uses as input.

In Browser Script, you cannot use an output property set for this format.

Usage

You can use this event in the following ways:

■ **In Server Script.** It can add properties to or modify values of the properties in the property set that the OutputArguments argument identifies.

■ **In Browser Script.** It cannot modify, store, or update the values of the properties in the output property set.

If you call a business service method through Browser Script, then the business service that this method calls can use a browser or the Siebel Server. For high interactivity mode, Siebel CRM determines if the business service resides in the browser. If the business service does not reside in the browser, then it sends the request to the Siebel Server.

Browser Script can call a business service on the browser or the Siebel Server. Server Script can call only a business service on the Siebel Server.

Used With
Browser Script, Server Script

Examples
To handle transactions that are not approved, the following example in Siebel eScript adds custom logic to the predefined Credit Card Transaction Service business service:

```
function Service_InvokeMethod (MethodName, Inputs, Outputs)

if (Outputs.GetProperty("SiebelResponseMessage") != "Approved")

{

    // special handling for failed transactions here

}
```

Service_PreCanInvokeMethod Event
Siebel CRM calls the Service_PreCanInvokeMethod event before it calls the PreInvokeMethod event. This configuration allows you to determine if the user possesses the authority to call a business service method. This method returns CancelOperation or ContinueOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Server Script Format
Service_PreCanInvokeMethod(*MethodName*, &CanInvoke)

Table 88 describes the arguments for the Server Script format of the Service_PreCanInvokeMethod event.

Table 88.   Arguments for the Server Script Format of the Service_PreCanInvokeMethod Event

Argument	Description
MethodName	A string that contains the name of the method that Siebel CRM must run.
&CanInvoke	A string that indicates if Siebel CRM can call the business service method. You can use one of the following values: ■ **TRUE.** Siebel CRM can call the business service method. ■ **FALSE.** Siebel CRM cannot call the business service method.

Browser Script Format

Service_PreCanInvokeMethod(*MethodName*)

The arguments you can use with this format are the same as the arguments described in Table 25 on page 116.

Used With

Browser Script, Server Script

Service_PreInvokeMethod Event

Siebel CRM calls the Service_PreInvokeMethod event before it calls a specialized method on a business service. For more information, see "About Specialized and Custom Methods" on page 103 and "Service_InvokeMethod Event" on page 289.

This method returns ContinueOperation or CancelOperation. For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Server Script Format

Service_PreInvokeMethod(*MethodName, InputArguments, OutputArguments*)

The arguments you can use in this format are the same as the arguments that are described in Table 83 on page 284.

Browser Script Format

Service_PreInvokeMethod(*name, inputPropSet*)

The arguments you can use in this format are the same as the arguments that are described in Table 24 on page 113.

Usage with Server Script

Siebel CRM uses the Server Script version of the Service_PreInvokeMethod event to perform the following work:

■ Performing business logic

■ Setting an output in the output property set

■ If you use a custom business service, then returning CancelOperation

Usage with Browser Script

Siebel CRM uses the Browser Script version of the Service_PreInvokeMethod event to perform the following work:

■ Performing a user interaction, such as asking for input data.

■ Setting an input property.

■ Canceling a user operation. For example, prompting the user to confirm a record deletion.

The Browser Script version is not intended to perform business logic. It does not return an output property set.

How Siebel CRM Handles a Predefined Business Service Method

Figure 7 illustrates how Siebel CRM handles a predefined business service method.
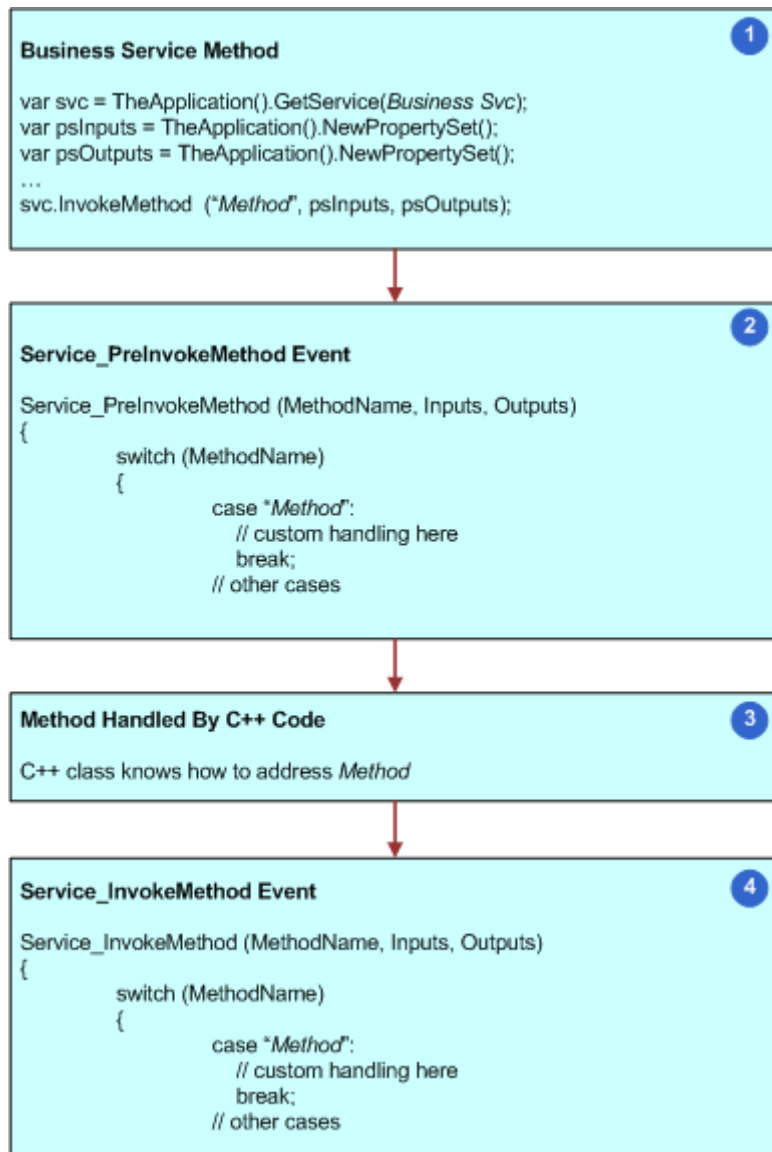
Figure 7.   Handling for a Predefined Business Service Method

With a predefined business service method, the script can do the following:

1 Call the Business Service Method.

2 In the Service_PreInvokeMethod event, process the Method and perform any necessary custom work before it runs the C++ code.

3 When the C++ code runs, it sets values in the outputs that the service code defines.

4 If the C++ code runs successfully, then the Service_InvokeMethod event can inspect and modify the output, or perform other tasks depending on the successful completion of the C++ code. At this point, the calling function takes control of the script flow.

How Siebel CRM Handles a Custom Business Service Method
Figure 8 illustrates how Siebel CRM handles a custom business service method.
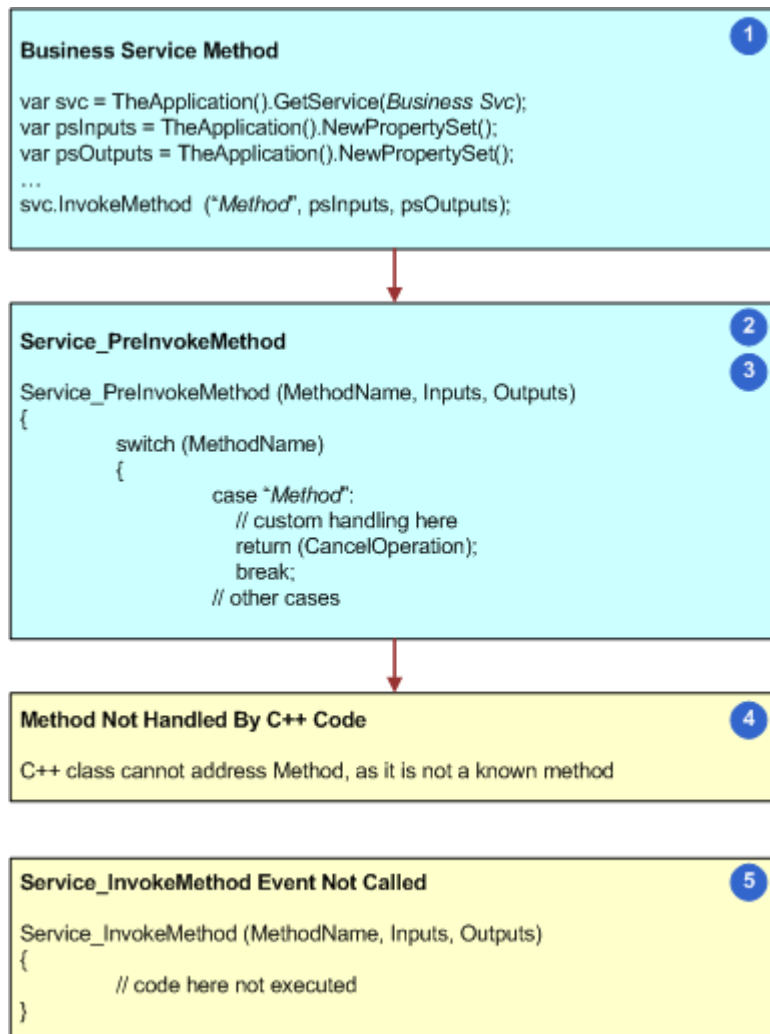
Figure 8.   Handling for a Custom Business Service Method

With a custom business service method, the script can do the following:

1 Call the Business Service Method.

2 In the Service_PreInvokeMethod event, process the method and take any necessary custom actions.

3 The script must return CancelOperation. This operation configures Siebel CRM to cancel the remaining operations that it associates with the event. If Siebel CRM does not cancel the remaining operations, then the flow continues to the C++ code.

4 this C++ code cannot handle the custom method, so it issues an error that is similar to the following error message:

       Unknown method name

5 Siebel CRM cancels the call to the method, so it does not run the Service_InvokeMethod event.

For more information, see "Caution About Using the Cancel Operation Event Handler" on page 58.

Used With

Browser Script, Server Script

Examples

The following Siebel VB example sets properties in the custom Shipping Engine business service:

```
Function Service_PreInvokeMethod (MethodName As String, Inputs As PropertySet,
Outputs As PropertySet) As Integer

    If MethodName = "CalculateShipping" Then

        Dim sShipper As String, sShipMethod As String
        Dim dWeight As Double, dSize As Double, dCost As Double
        Dim sZone As String, DelDate As Variant
        Dim sCost As String, iReturn As Integer

        iReturn = ContinueOperation
        sShipper = Inputs.GetProperty("Shipping Company")
        sShipMethod = Inputs.GetProperty("Ship Method")
        dWeight = Val(Inputs.GetProperty("Weight"))
        dSize = Val(Inputs.GetProperty("Volume"))
        iZone = Val(Inputs.GetProperty("Zone"))
        DelDate = DateValue(Now)

        Select Case sShipper
            Case "GlobalEx"
                Select Case sShipMethod
                    Case "Next-Day Air"
                        dCost = 14 + dWeight
                        DelDate = DelDate + 1
                    Case "Second-Day Air"
                        dCost = 11 + (dWeight * .54)
                        DelDate = DelDate + 2
                End Select

            Case "Airline"
                Select Case sShipMethod
                    Case "Next-Day Air"
                        dCost = 5 + (dWeight * .3) + (dSize * .33) + _
                            (Val(sZone) * .5)
```

```
            DelDate = DelDate + 1
        Case "Second-Day Air"
            dCost = 4 + (dWeight * .3) + (dSize * .2) + _
              (Val(sZone) * .3)
            DelDate = DelDate + 2

        Case "Ground"
            dCost = 3 + (dWeight * .18) + (dSize * .1) + _
              (Val(sZone) * .1)
            DelDate = DelDate + 2 + Int(Val(sZone) * .8)
      End Select
   End Select

   sCost = Format(dCost, "Currency")
   Outputs.SetProperty "Cost", sCost
   Outputs.SetProperty "Delivery Date", DelDate
   iReturn = CancelOperation

 End If

 Service_PreInvokeMethod = iReturn

End Function
```

Control Methods

This topic describes control methods. It includes the following topics:

In this topic, the controlVar variable indicates the name of the control that causes Siebel CRM to call the method. For example, Button1_Click.

A control method does not work with an ActiveX control.

Applet Method for a Control

The Applet method returns a string that contains the name of the applet that contains the control.

Format
*controlVar.*Applet

No arguments are available.

Usage
Getting the name of the applet that contains the control allows you to configure Siebel CRM to do operations on the applet, not only on the control.

Used With
Browser Script

BusComp Method for a Control
The BusComp method returns a string that contains the name of the business component that an applet references. The control resides in this applet.

Format
*controlVar.*BusComp

No arguments are available.

Used With
Browser Script

Examples
For an example, see "Name Method for a Business Component" on page 220.

GetProperty Method for a Control
The GetProperty method returns a string that contains the value of a property. If the property does not exist, then this method returns NULL.

Format
*controlVar.*GetProperty(*propName*)

No arguments are available.

Usage
You can use the GetProperty method with the following controls:

■ CheckBox

■ ComboBox

- TextBox

- TextArea

- Label

You can use the GetProperty method to get values for the following properties:

- Background Color

- Enabled

- FontType

- FontColor

- FontSize

- FontStyle

- Height

- Width

- Read Only

- Visible

For more information about these properties, see Table 90 on page 301.

To return more than one property, you must use a separate statement for each property.

Used With
Browser Script

Examples
The following example uses the GetProperty method to return values for the FontSize, Background Color, Width, and Height properties:

```
theApplication().SWEAlert("checkbox.FontSize : " +
objCheckBox.GetProperty("FontSize"));
theApplication().SWEAlert("checkbox.BgColor : " +
objCheckBox.GetProperty("BgColor"));
theApplication().SWEAlert("checkbox.Width : " + objCheckBox.GetProperty("Width"));
theApplication().SWEAlert("checkbox.Height : " +
objCheckBox.GetProperty("Height"));
```

GetValue Method for a Control

The GetValue method returns the value that a control displays for the data type of the field that the control references. The type of value depends on the specific control. This method returns the value in a string.

The GetValue method cannot return a literal value that a user provides as input to a control. This method returns the value that Siebel CRM stores for the user entry, according to the data type of the field that the control references.

Format
*controlVar.*GetValue

No arguments are available.

Usage
For more information, see "Usage for the GetValue Method and the SetValue Method" on page 305.

Used With
Browser Script

Examples
For an example, see "Examples for the GetValue Method and the SetValue Method" on page 305.

Name Method for a Control
The Name method for a control returns a string that contains the name of a control.

Format
*controlVar.*Name

No arguments are available.

Used With
Browser Script

Examples
For an example, see "Name Method for a Business Component" on page 220.

SetLabelProperty Method for a Control
The SetLabelProperty method sets the properties of a label. This method does not return any information.

Format
*controlVar.*SetLabelProperty(*propName*, *propValue*)

Table 89 describes the arguments for the SetLabelProperty method.

Table 89. Arguments for the SetLabelProperty Method

Argument	Description
propName	The name of the property that Siebel CRM must set. For a description of the values you can enter, see "Properties You Can Set For a Label" on page 301.
propValue	The value to set for the property. For a description of the values you can enter, see "Properties You Can Set For a Label" on page 301.

Usage

If you must set more than one property, then you must use a separate statement for each property.

Enabling the SetLabelProperty Method

Siebel CRM does not enable the SetLabelProperty method by default. You must enable it in Siebel Tools before you use it in a script.

To enable the SetLabelProperty method

1 Open Siebel Tools.

2 Display the Control User Prop object type:

   a Choose the View menu, and then the Options menu item.

   b Click the Object Explorer tab.

   c Scroll down through the Object Explorer Hierarchy window until you locate the Applet tree.

   d Expand the Applet tree, expand the Control tree, and then make sure the Control User Prop object type includes a check mark.

   e Click OK.

3 In the Object Explorer, click Applet.

4 In the Applets list, locate the applet that includes the control you must modify.

5 In the Object Explorer, expand the Applet tree, and then click Control.

6 In the Controls list, locate the control you must modify.

7 In the Object Explorer, expand the Control tree, and then click Control User Prop.

8 In the Control User Props list, add a new control user property using values from the following table.

Property	Value
Name	useLabelID
Value	TRUE

Properties You Can Set For a Label

Table 90 lists the properties you can set for a label.

Table 90.   Properties You Can Set For a Label

Property	Value	Description
BgColor	string	Determines the background color for a label. For example: ■ Red is #ff0000. ■ Green is #00ff00. ■ Blue is #0000ff.
FontColor	string	Determines the font color for a label. For example, green is #00ff00.
FontType	string	Determines the font type for a label. For example, Times Roman.
FontSize	string	Determines the font size for a label. For example, 12 pt.
FontStyle	string	Determines the font style for a label. For example, italic.
FontWeight	string	Determines the font weight for a label. You can use the following values: ■ bold ■ bolder ■ lighter ■ normal ■ 100, 200, 300, or 400. These values are equivalent to light. ■ 500, 600, or 700. These values are equivalent to normal. ■ 800 or 900. These values are equivalent to bold. The default value is normal.
Height	string	Determines height for a label, in pixels. For example, 5.
Visible	visible or hidden	Determines if the label is visible. The default value is the value in the Siebel repository file (SRF).
Width	string	Determines the width for a label, in pixels. For example, 80.

Used With

Browser Script

Examples

The following code uses the SetLabelProperty method:

```
function Applet_PreInvokeMethod (name, inputPropSet){
```

```
switch (name) {

    // Example of changing the font size of the Location label
    case ("fontsize"):
    {

        var ctl = this.FindControl("Location");
        var fSize = prompt("Specify the required label font size (numeric value
        only).");
        ctl.SetLabelProperty("FontSize", fSize);
        return ("CancelOperation");

    }

    // Example of changing the background color of the Location label
    case ("bgcolor"):
    {

        var ctl = this.FindControl("Location");
        var bgColor = prompt("Specify th ebackground color of the label. Enter a valid
        six hexadecimal digit RGB value preceded by #");
        ctl.SetLabelProperty("BgColor", bgColor);
        return ("CancelOperation");

    }

    // Example of changing the font type of the Location label
    case ("fonttype"):
    {

        var ctl = this.FindControl("Location");
        var fontType =prompt("Specify the font type for the label.");
        ctl.SetLabelProperty("FontType", fontType);
        return ("CancelOperation");

    }

    // Example of changing the font color of the Location label
    case ("fontcolor"):
    {

        var ctl = this.FindControl("Location");
        var fontColor = prompt("Specify the font color of the label. Enter a valid six
        hexadecimal digit RGB value preceded by #");
        ctl.SetLabelProperty("FontColor", fontColor);
        return ("CancelOperation");

    }

    break;

    }

}
```

SetProperty Method for a Control

The SetProperty method sets the properties of a control. This method does not return any information.

Format

*controlVar.*SetProperty(*propName*, *propValue*)

Table 91 describes the arguments for the SetProperty method.

Table 91.   Arguments for the SetProperty Method

Argument	Description
propName	The name of the property that Siebel CRM must set. For a description of the values you can enter, see "Properties You Can Set for a Control" on page 303.
propValue	The value that Siebel CRM must set for the property. For a description of the values you can enter, see "Properties You Can Set for a Control" on page 303.

Usage

You can use the SetProperty method with the following controls:

■ CheckBox

■ ComboBox

■ TextBox

■ TextArea

If you must set more than one property, then you must use a separate statement to set each property.

Properties You Can Set for a Control

Table 92 describes the properties you can set for a control.

Table 92.   Properties You Can Set for a Control

Property	Value	Description
Enabled	TRUE or FALSE	Determines if the control is active. The default value is the value in the Siebel repository file (SRF).
Shown	TRUE or FALSE	Determines if Siebel CRM displays the control. The default value is the value in the Siebel repository file (SRF).
ReadOnly	TRUE or FALSE	Determines if the control is read-only. The default value is the value in the Siebel repository file (SRF).

Table 92.   Properties You Can Set for a Control

Property	Value	Description
BgColor	To modify these control properties, you can use these same properties you use to modify a label. For a description of the values you can enter, see "Properties You Can Set For a Label" on page 301.	
FontColor		
FontType		
FontSize		
FontStyle		
FontWeight		
Height		
Visible		
Width		

Used With
Browser Script

Using the SetProperty Method to Control Font Weight
To use the SetProperty method to control font weight, you must use the FontWeight property. For example:

```
control.SetProperty("FontWeight", "600")
```

You cannot use the FontStyle argument to control font weight. For example, the following code fails:

```
control.SetProperty("FontStyle",  "Bold")
```

Examples
The following code uses the SetProperty method:

```
objCheckBox.SetProperty("FontColor",  "#00ff00");
objCheckBox.SetProperty("FontStyle",  "italic");
objCheckBox.SetProperty("FontType",  "Verdana");
objCheckBox.SetProperty("FontSize",  "14 pt");
objCheckBox.SetProperty("BgColor",  "#00f000");
objCheckBox.SetProperty("Width",  "100");
objCheckBox.SetProperty("Height",  "100");
```

SetValue Method for a Control
The SetValue method sets the contents a control. This method does not return any information.

Format
controlVar.SetValue (*controlValue*)

Table 93 describes the arguments for the SetValue method.

Table 93.   Arguments for the SetValue Method

Argument	Description
controlValue	String that contains the value that Siebel CRM must set for the control.

Usage for the GetValue Method and the SetValue Method
Note the following usage for the SetValue method:

■   This method does not validate the format of the data. Data validation occurs when the user steps off the field or the record, or explicitly saves the record.

■   This method can set the value for a read-only control, but Siebel CRM does not save this information when the user saves the record.

■   The user can modify the contents of a control before Siebel CRM saves control information to the business component field.

Note the following usage for the GetValue method and the SetValue method:

■   These methods only work on form applets.

■   These methods work only for a control that references a business component field.

■   You cannot use these methods with a label.

Used With
Browser Script

Examples for the GetValue Method and the SetValue Method
The following code uses the GetValue method and the SetValue method:

```
function Applet_PreInvokeMethod (name, inputPropSet)
{

switch (name) {

    // Example of changing the value of the Abstract control to uppercase
    case ("SR Abstract"):
    {

        var ctlName = "Abstract";
        var ctl = this.FindControl(ctlName);
        var ctlVal = ctl.GetValue();
        ctl.SetValue(ctlVal.toUpperCase());
        ctl = null;
        return("CancelOperation");

    }
```

```
// Example of changing the value of a checkbox control
case ("SR Billable"):
{

    var ctlName = "Billable Flag";
    var ctl  = this.FindControl(ctlName);
    var ctlVal  = ctl.GetValue();
    if (ctlVal  == "Y")
       ctl.SetValue("N"); // clear the box
    else
       ctl.SetValue("Y"); // check the box
    ctl = null;
    return("CancelOperation");

}

// Example of changing the value of a date/time control
case ("SR Commit time"):
{

    var ctlName = "Agent Committed";
    var ctl  = this.FindControl(ctlName);
    ctl.SetValue("12/1/2001 1:09:31 AM");
    // format is not validated until user saves the record
    ctl = null;
    return("CancelOperation");

}

break;

}

}
```

Property Set Methods

This topic describes property set methods. It includes the following topics:

In this topic, the oPropSet variable indicates the variable that contains a property set.

AddChild Method for a Property Set

The AddChild method adds a child property set to a property set. This method returns an integer that indicates the index of the child property set.

Format

oPropSet.AddChild(*childPropSet*)

Table 94 describes the arguments for the AddChild method.

Table 94.   Arguments for the AddChild Method

Argument	Description
childObject	A property set that Siebel CRM must make as a child to the property set that the oPropSet variable identifies.

Usage

You can use a property set to create a tree data structure. You can add any number of arbitrarily structured child properties to a property set. You can use a child property set to structure a property set in a manner that is similar to the structure that the data model uses. For example, a parent account property set can include child property sets for opportunities, contacts, activities, and so forth. In this example, you could create an independent property set named Opportunity, where accounts, contacts, and activities can be children.

If Siebel CRM creates an instance of a property set through script, and then adds it to a parent property set, and if the parent property set is subsequently released, then Siebel CRM does not release this child instance. The reference to the child instance exists independently.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

The following fragment of Siebel eScript code adds child property sets to a parent property set:

```
var Account = TheApplication().NewPropertySet();
var Opportunity = TheApplication().NewPropertySet();
var Contact = TheApplication().NewPropertySet();
var Activity = TheApplication().NewPropertySet();

Account.AddChild(Opportunity);
Account.AddChild(Contact);
Account.AddChild(Activity);
```

Related Topics

For more information, see the following topics:

■ "GetChild Method for a Property Set" on page 310

■ "InsertChildAt Method for a Property Set" on page 317

■ "RemoveChild Method for a Property Set" on page 318

Copy Method for a Property Set

The Copy method returns a copy of a property set.

Format

oPropSet.Copy()

No arguments are available.

Usage

The Copy method creates a copy of a property set, including any properties and child property sets. Siebel CRM typically passes a property set through a reference, so making a copy allows you to manipulate a property set without affecting the original property set.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

The following Siebel VB example uses a copy of a property set to store the original values of the properties, and displays the original and Pig-Latin forms of the properties:

```
(general)
(declarations)
Option Explicit

Function PigLatin (Name1 As String) As String
    Dim Name2 As String, FirstLetter As String
    Name2 = Right$(Name1, len(Name1) - 1)
    FirstLetter = Left$(Name1, 1)
    Name2 = UCase(Mid$(Name1, 2, 1)) & _
        Right$(Name2, Len(Name2) - 1)
    Name2 = Name2 & LCase(FirstLetter) & "ay"
    PigLatin = Name2
End Function

Sub ClickMe_Click()

    Dim Inputs As PropertySet, Outputs As PropertySet
    Dim message As String, propName, propVal, newPropVal
    set Inputs = TheApplication.NewPropertySet

    Inputs.SetProperty "Name", "Harold"
    Inputs.SetProperty "Assistant", "Kathryn"
    Inputs.SetProperty "Driver", "Merton"

    set Outputs = Inputs.Copy()

    propName = Outputs.GetFirstProperty()
    do while propName <> ""
        propVal  = Outputs.GetProperty(propName)
        newPropVal = PigLatin(propVal)
        Outputs.SetProperty propName, newPropVal
        message = message & propVal & " has become " & _
            newPropVal & Chr$(13)
        propName = Outputs.GetNextProperty()
    loop
    TheApplication.RaiseErrorText message

Set message = Nothing
Set Outputs = Nothing
Set Inputs = Nothing

End Sub
```

GetByteValue Method for a Property Set

The GetByteValue method returns the following information:

■ If a byte value is set, then this method returns a byte array.

■ If a string value is set, then this method returns a null value.

For more information, see "SetByteValue Method for a Property Set" on page 319.

Format

oPropSet.getByteValue()

No arguments are available.

Used With

Siebel Java Data Bean

Examples

The following example uses a binary value as input and provides a binary output. The angle brackets (< >) indicate a variable:

```
SiebelPropertySet input = new SiebelPropertySet();
SiebelPropertySet output = new SiebelPropertySet();

input.setProperty("ProcessName", "LMS3 Jason");

// XML to send
String str="<?xml version=\"1.0\" encoding=\"UTF8\"
?><GetCommunicationDataInput><MemberID>20048963</MemberID></
GetCommunicationDataInput>";

// convert string to byte array
byte [] bvalue = new String(str).getBytes();

input.setByteValue(bvalue);
businessService.invokeMethod("RunProcess",input,output);

// Use getByteValue to return the value..and pop it in a String..for example
String out2 = new String (output.getByteValue());
System.out.println(out2);
```

GetChild Method for a Property Set

The GetChild method returns the index number of a child property set.

Format

oPropSet.GetChild(*index*)

Table 95 describes the arguments for the GetChild method.

Table 95.   Arguments for the Arguments for the GetChild Method

Argument	Description
index	An integer that identifies the index number of the child property set that Siebel CRM must return.

How Siebel CRM Handles Indexing for Child Property Sets

Note how Siebel CRM handles indexing for a child property set you add, insert, or remove:

■ If Siebel CRM creates a child property set, then it creates an index number for this child property set, starting at 0 (zero). It increments this index for each child property set it adds to a given parent property set.

■ If you use the AddChild Property method, then Siebel CRM uses the next available index number for the child property set it adds.

■ If you use the InsertChildAt method, then Siebel CRM inserts the new child property set at a specified index. It also increases the index by 1 for the property set that the new child displaces, and for all child property sets that occur after the displaced property set.

■ If you use the RemoveChild method, then Siebel CRM removes the child property set you specify, and then decreases the index by 1 for all property sets that follow the removed child.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

If you use the Web Client Automation Server, then the child object that Siebel CRM returns is a copy of the object. Siebel CRM updates any modifications that occur to the object that it returns, but it does not update the originating object.

Examples

The following Siebel eScript example sets the Name property of child property sets to the same value:

```
function Test1_Click ()
{
    var Account = TheApplication().NewPropertySet();
    var Opportunity = TheApplication().NewPropertySet();
    var Contact = TheApplication().NewPropertySet();
    var Activity = TheApplication().NewPropertySet();
    var j;

    Account.AddChild(Opportunity);
    Account.AddChild(Contact);
    Account.AddChild(Activity);

    for (var i = 0; i < Account.GetChildCount(); i++)
    {
        j = Account.GetChild(i);
        j.SetProperty('Name', 'Allied Handbooks');
    }
}
```

Related Topics

For more information, see the following topics:

■ "AddChild Method for a Property Set" on page 307

■ "InsertChildAt Method for a Property Set" on page 317

GetChildCount Method for a Property Set

The GetChildCount method returns the number of child property sets that exist for a parent property set.

Format

oPropSet.GetChildCount()

No arguments are available.

Usage

The GetChildCount method returns the number of child property sets for the property set that the oPropSet variable identifies. The index number for child property sets start at 0, so a child count of 3 indicates that there are child property sets at indexes 0, 1, and 2.

The GetChildCount method returns only the number of direct descendants. If a child property set includes children, then Siebel CRM does not include these grandchildren in the count that it provides in the return value.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

For an example, see "GetChild Method for a Property Set" on page 310.

GetFirstProperty Method for a Property Set

The GetFirstProperty method for a property set returns a string that contains the name of the first property in a property set.

Format

oPropSet.GetFirstProperty()

No arguments are available.

Usage

The usage for the GetFirstProperty method for a property set is similar to the usage for the GetFirstProperty method for a business service. For more information, see "Usage for a Method that Gets a Business Service Property" on page 281.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

The following example uses the GetFirstProperty method to get the first property, and then uses the GetNextProperty method to return all subsequent properties. If the GetNextProperty method returns a null value, then Siebel CRM terminates the loop:

```
function Service_PreInvokeMethod (MethodName, Inputs, Outputs)
{
    var propName = "";
    var propVal  = "";

    propName = Inputs.GetFirstProperty();

    // stay in loop if the property name is not an empty string
    while (propName != "") {
        propVal  = Inputs.GetProperty(propName);

        // if a property with the same name does not exist
        // add the name value pair to the output
        if (!Outputs.PropertyExists(propName)) {
            Outputs.SetProperty(propName, propVal);
        }

        propName = Inputs.GetNextProperty();

    }
    return (CancelOperation);
}
```

Related Topics

For more information, see the following topics:

GetLastErrCode Method for a Property Set

The GetLastErrCode method returns the error code for the error that Siebel CRM logged most recently. This code is a short integer. 0 (zero) indicates no error.

Format

*oPropSet.*GetLastErrCode

No arguments are available.

Usage

For more information, see "Usage for the GetLastErrCode Method" on page 139.

Used With

Mobile Web Client Automation Server, Web Client Automation Server

GetLastErrText Method for a Property Set

The GetLastErrText method returns a string that contains the text message for the error that Siebel CRM logged most recently.

Format

*oPropSet.*GetLastErrText

No arguments are available.

Usage

For more information, see "Usage for the GetLastErrText Method" on page 139.

Used With

Mobile Web Client Automation Server, Web Client Automation Server

GetNextProperty Method for a Property Set

The GetNextProperty method returns a string that contains the name of the next property of a property set. If no more properties exist, then this method returns an empty string.

Format

*oPropSet.*GetNextProperty()

No arguments are available.

Usage

Usage for the GetNextProperty method for a property set is similar to the usage for the GetNextProperty method for a business service. For more information, see "Usage for the GetNextProperty Method" on page 283.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

For an example, see "GetFirstProperty Method for a Property Set" on page 312.

Related Topics

For more information, see the following topics:

■ "GetFirstProperty Method for a Property Set" on page 312

■ "GetProperty Method for a Property Set" on page 315

GetProperty Method for a Property Set

The GetProperty method returns a string that contains the value of a property. If the property does not exist, then this method returns NULL.

Format

oPropSet.GetProperty(*propName*)

The arguments you can use with this format are the same as the arguments described in Table 82 on page 283.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

The following fragment of Siebel eScript code receives a set of input properties used with the Shipping Engine business service described in "Service_PreInvokeMethod Event" on page 291:

```
var sShipper = Inputs.GetProperty("Shipping Company");
var dWeight = Val(Inputs.GetProperty("Weight"));
var dSize = Val(Inputs.GetProperty("Total Dimensions"));
var iZone = Val(Inputs.GetProperty("Zone"));
```

Related Topics

For more information, see the following topics:

■ "GetFirstProperty Method for a Property Set" on page 312

■ "GetNextProperty Method for a Property Set" on page 314

■ "SetProperty Method for a Property Set" on page 320

GetPropertyCount Method for a Property Set

The GetPropertyCount method returns the number of properties that exist in the current level in the hierarchy. It does not return all properties in the entire property set hierarchy.

Format
oPropSet.GetPropertyCount

No arguments are available.

Used With
Browser Script, COM Data Control, COM Data Server, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

GetType Method for a Property Set
The GetType method returns a string that contains the value of the type attribute of a property set.

Format
oPropSet.GetType

No arguments are available.

Used With
Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Related Topics
For more information, see the following topics:

■ "GetValue Method for a Property Set" on page 316

■ "SetType Method for a Property Set" on page 321

GetValue Method for a Property Set
The GetValue method returns a string that contains the value of the value attribute of a property set.

Format
oPropSet.GetValue

No arguments are available.

Used With
Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Related Topics
For more information, see the following topics:

InsertChildAt Method for a Property Set

The InsertChildAt method inserts a child property set in a parent property set at a specific location. This method does not return any information. For more information, see "AddChild Method for a Property Set" on page 307.

Format

oPropSet.InsertChildAt *childObject*, *index*

Table 96 describes the arguments for the InsertChildAt method.

Table 96.   Arguments for the InsertChildAt Method

Argument	Description
childObject	The property set that Siebel CRM must make a child. It makes this property set a child of the property set that the oPropSet variable identifies.
index	An integer that identifies the position where Siebel CRM must insert the property set. The childObject argument identifies this property set.

Usage

For more information, see "How Siebel CRM Handles Indexing for Child Property Sets" on page 311.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

PropertyExists Method for a Property Set

The description of the PropertyExists method for a property set is the same as the description of the PropertyExists method for a business service. For more information, see "PropertyExists Method for a Business Service" on page 285.

Format

oPropSet.PropertyExists(*propName*)

The arguments you can use with this format are the same as the arguments described in Table 84 on page 286.

Usage

The GetProperty method returns an empty string for every nonexistent property, so you can use the PropertyExists method in an If statement to determine if a specific property is set.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

For an example, see "GetFirstProperty Method for a Property Set" on page 312.

RemoveChild Method for a Property Set

The RemoveChild method removes a child property set from a parent property set. This method does not return any information.

Format

oPropSet.RemoveChild *index*

Table 97 describes the arguments for the RemoveChild method.

Table 97.   Arguments for the RemoveChild Method

Argument	Description
index	An integer that identifies the index number of the child property set that Siebel CRM must remove.

Usage

For more information, see "How Siebel CRM Handles Indexing for Child Property Sets" on page 311.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Examples

The following Siebel VB code fragment removes every child property set of a property set:

```
Dim i As Integer
for i = 0 to outputs.GetChildCount()
   outputs.RemoveChild(0)
Next i
```

Related Topics

For more information, see the following topics:

■ "AddChild Method for a Property Set" on page 307

■ "InsertChildAt Method for a Property Set" on page 317

RemoveProperty Method for a Property Set

The RemoveProperty method removes a property from a property set. This method does not return any information.

Format

oPropSet.RemoveProperty *propName*

The arguments you can use with this format are the same as the arguments described in Table 85 on page 288.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Reset Method for a Property Set

The Reset method removes all properties and children from a property set. This method does not return any information.

Format

oPropSet.Reset()

No arguments are available.

Usage

The Reset method allows you to reuse a property set.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

SetByteValue Method for a Property Set

The SetByteValue method sets the value of a property set. This method does not return any information.

Format

oPropSet.setByteValue(*value*)

Table 98 describes the arguments for the SetByteValue method.

Table 98.   Arguments for the SetByteValue Method

Argument	Description
value	The byte array that contains the value that Siebel CRM must set.

Used With

Siebel Java Data Bean

Examples

The following example uses a binary value as input and then provides a binary output. For more information, see "GetByteValue Method for a Property Set" on page 309:

```
SiebelPropertySet input = new SiebelPropertySet();
SiebelPropertySet output = new SiebelPropertySet();

input.setProperty("ProcessName", "LMS3 Jason");

// XML to send
String str="<?xml version=\"1.0\" encoding=\"UTF8\"
?><GetCommunicationDataInput><MemberID>20048963</MemberID></
GetCommunicationDataInput>";

// convert string to byte array
byte [] bvalue = new String(str).getBytes();

input.setByteValue(bvalue);
businessService.invokeMethod("RunProcess",input,output);

// use getByteValue to return the value and put it in a String
String out2 = new String (output.getByteValue());
System.out.println(out2);
```

SetProperty Method for a Property Set

The SetProperty method sets a value in the property of a property set. This method does not return any information. For more information, see "GetProperty Method for a Property Set" on page 315.

Format

oPropSet.SetProperty *propName, propValue*

The arguments you can use with this format are the same as the arguments described in Table 86 on page 288.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client
Automation Server, Server Script, Web Client Automation Server

Examples

This Siebel VB fragment uses the Shipping Engine business service:

```
Dim Svc As Service
Dim Inputs As PropertySet, Outputs As PropertySet
Set Svc = TheApplication.GetService("Shipping Engine")
Set Inputs = TheApplication.NewPropertySet()

With Inputs
    .SetProperty "Shipping Company", "Airline"
    .SetProperty "Weight", "12"
    .SetProperty "Total Dimensions", "48"
    .SetProperty "Shipping Method", "Second-Day Air"
End With
```

For more information, see "Service_PreInvokeMethod Event" on page 291.

SetType Method for a Property Set

The SetType method sets the value for the type attribute of a property set. This method does not
return any information.

Format

oPropSet.SetType *type*

Table 99 describes the arguments for the SetType method.

Table 99.   Arguments for the SetType Method

Argument	Description
type	A string that contains data that Siebel CRM must store in the type attribute.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client
Automation Server, Server Script, Web Client Automation Server

Related Topics

For more information, see the following topics:

■ "GetType Method for a Property Set" on page 316

■ "SetValue Method for a Property Set" on page 322

SetValue Method for a Property Set

The SetValue method sets the value for the value attribute of a property set. This method does not return any information.

Format

oPropSet.SetValue *value*

Table 100 describes the arguments for the SetValue method.

Table 100. Arguments for the Arguments for the SetValue Method

Argument	Description
value	A string that contains data that Siebel CRM must store in the value attribute.

Used With

Browser Script, COM Data Control, COM Data Server, Siebel Java Data Bean, Mobile Web Client Automation Server, Server Script, Web Client Automation Server

Related Topics

For more information, see the following topics:

■ "GetValue Method for a Property Set" on page 316

■ "SetProperty Method for a Property Set" on page 320

Miscellaneous Methods

This topic describes other methods. It includes the following topics:

■ "GetErrorCode Method" on page 322

■ "GetErrorMessage Method" on page 323

■ "TheApplication Method" on page 324

GetErrorCode Method

The GetErrorCode method returns a string that contains a numeric error code. For more information, see "GetErrorMessage Method" on page 323.

Format

public int getErrorCode()

No arguments are available.

Used With

Siebel Java Data Bean

Examples

The following example for the Siebel Java Data Bean returns the first record in the Account business component. If an error occurs, then the script displays the error code and error message:

```
try
{
    //Instantiate the Siebel Java Data Bean
    Sieb_dataBean = new SiebelDataBean();
    String Cstr = "GatewayServer, EntServer, FINSObjMgr";
    Sieb_dataBean.login(Cstr, "SADMIN", "SADMIN");
    SiebelBusObject m_busObject = Sieb_dataBean.getBusObject("Account");
    SiebelBusComp m_busComp = m_busObject.getBusComp("Account");
    m_busComp.activateField("Name");
    m_busComp.executeQuery(true);
    m_busComp.firstRecord();
    Name = m_busComp.getFieldValue("Name");
    System.out.println("Account Name : " + Name);

    m_busComp.release();
    m_busComp = null;

    m_busObject.release();
    m_busObject = null;

    Sieb_dataBean.logoff();
    Sieb_dataBean = null;
}

catch (SiebelException e)
{
    ErrorText = "Code: " +  e.getErrorCode() + "\n" + "Description: " +
e.getErrorMessage();
    System.out.println("Error Occurred\n " + ErrorText);
}

...
```

GetErrorMessage Method

The GetErrorMessage method returns a string that contains an error message. For more information, see "GetErrorCode Method" on page 322.

Format

public string getErrorMessage()

No arguments are available.

Used With

Siebel Java Data Bean

TheApplication Method

The theApplication method is a global method that returns a string that contains the name of an application object. This object is the root of objects in the Siebel Application Object hierarchy.

Browser Script Format

theApplication()

Siebel VB Format

TheApplication

Siebel eScript Format

TheApplication()

No arguments are available.

Usage

You can use the theApplication method to determine the object reference of the Siebel application. You can then use this information to find other objects or to call a method on the application object. For example, if you use Siebel eScript to determine if you are logged in to a server database or local database, then you can use the following code:

```
TheApplication().InvokeMethod("GetDataSource")
```

Used With

Browser Script, Server Script

Examples

The following Siebel VB example returns the login name from the application object and creates the Employee business object:

```
Dim oEmpBusObj as BusObject
Dim sLoginName as String

sLoginName = TheApplication.LoginName
Set oEmpBusObj = TheApplication.GetBusObject("Employee")

...

Set oEmpBusObj = Nothing
```

6 Browser Script Quick Reference

This chapter describes summary information for Browser Script. It includes the following topics:

For more information, see "Browser Script" on page 18.

Applet Methods for Browser Script

Table 101 describes a summary of the applet methods you can use in Browser Script.

Table 101. Summary of Applet Methods for Browser Script

Method	Description	Format
ActiveMode Method for an Applet	Returns a string that contains the name of the current Web template mode.	`var oApplet;` `var mode = oApplet.ActiveMode();`
BusComp Method for an Applet	Returns the name of the business component that an applet references.	`var oApplet;` `var busComp = oApplet.BusComp();`
BusObject Method for an Applet	Returns the name of the business object for the business component that an applet references.	`var oApplet;` `var oBusObject = oApplet.BusObject();`

Table 101. Summary of Applet Methods for Browser Script

Method	Description	Format

FindActiveXControl Method for an Applet	Returns the name of a control that is a Document Object Model element.	`var oApplet;`
		`var oControl;`
		`oControl = oApplet.FindActiveXControl`
		`(controlName as String);`

FindControl Method for an Applet	Returns the name of a control.	`var oApplet;`
		`var oControl;`
		`oControl =`
		`oApplet.FindControl (controlName as`
		`String);`

InvokeMethod Method for an Applet	Calls a method.	`var oApplet;`
		`var outPs;`
		`outPs =`
		`oApplet.InvokeMethod(MethodName as`
		`String, inputPropSet as PropertySet);`

Name Method for an Applet	Returns the name of an applet.	`var oApplet;` `var name = oApplet.Name();`

Applet Events For Browser Script

Table 102 describes a summary of the applet events you can use in Browser Script.

Table 102. Summary of Applet Events for Browser Script

Event	Description	Format
Applet_ChangeFieldValue Event	Starts if the user uses an applet to modify data in a field.	`Applet_ChangeFieldValue` `(field, value)`
Applet_ChangeRecord Event	Starts if the user moves to a different record or view.	`Applet_ChangeRecord()`
Applet_InvokeMethod Event	Starts after a specialized method or after a custom method is called.	`Applet_InvokeMethod (name,` `inputPropSet)`
Applet_Load Event	Starts after Siebel CRM loads an applet and after it displays data.	`Applet_Load()`
Applet_PreInvokeMethod Event	Siebel CRM calls this event immediately before it calls a specialized method on an applet.	`Applet_PreInvokeMethod` `(name, inputPropSet)`

Application Methods for Browser Script

Table 103 describes a summary of the application methods you can use in Browser Script. It does not include object interface methods that Siebel CRM does not call directly from an application object instance. For information about methods it calls with the InvokeMethod method on the application object, see "LoadObjects Method for an Application" on page 150.

Table 103. Summary of Application Methods for Browser Script

Method	Description	Format
ActiveApplet Method for an Application	Returns the name of the active applet.	var applet; applet = theApplication().ActiveApplet();
ActiveBusComp Method for an Application	Returns the name of the business component that the active applet references.	var busComp; busComp = theApplication().ActiveBusComp();
ActiveBusObject Method for an Application	Returns the name of the business object for the business component that the active applet references.	var busObject; busObject = theApplication().ActiveBusObject();
ActiveViewName Method for an Application	Returns the name of the active view.	var viewName; viewName = theApplication().ActiveViewName();
FindApplet Method for an Application	Returns the name of an applet.	var applet; applet = theApplication().FindApplet (*appletName*);
GetProfileAttr Method for an Application	Returns the name of an attribute in a user profile.	var sAttr; sAttr = theApplication().GetProfileAttr(*name*);
GetService Method for an Application	Locates a business service. If this business service is not already running, then Siebel CRM starts it.	var svc; svc = theApplication().GetService (*serviceName*);
InvokeMethod Method for an Application	Calls a method.	var outPs; outPs = theApplication().InvokeMethod (*methodName, methArg1, methArg2, methArgN*);
Name Method for an Application	Returns the name of the Siebel application.	var appName; appName = theApplication().Name();
NewPropertySet Method for an Application	Creates a new property set.	var PropSet; PropSet = theApplication().NewPropertySet();

Table 103. Summary of Application Methods for Browser Script

Method	Description	Format
SetProfileAttr Method for an Application	Personalization uses this method to set a value for an attribute in a user profile.	theApplication().SetProfileAttr(*name*, *value*);
ShowModalDialog Method for an Application	Allows you to display a dialog box with the cursor in the default state.	theApplication().ShowModalDialog (*url[, argin][, options]*)
SWEAlert Method for an Application	Displays a modal dialog box that includes a message.	theApplication().SWEAlert(message);

Application Events for Browser Script

Table 104 describes a summary of the application events you can use in Browser Script.

Table 104. Summary of Application Events for Browser Script

Event	Description	Format
Application_InvokeMethod Event	Called after Siebel CRM calls a specialized method.	Application_InvokeMethod (*name*, *inputPropSet*)
Application_PreInvokeMethod Event	Called after Siebel CRM calls a specialized method.	Application_PreInvokeMethod (*name*, *inputPropSet*)

Business Component Methods for Browser Script

Table 105 describes a summary of the business component methods you can use in Browser Script. It does not include object interface methods that Siebel CRM does not call directly from a Business Component object instance. For information about methods that it calls with InvokeMethod method on the Business Component object, see "Business Component Invoke Methods" on page 252.

Table 105. Summary of Business Component Methods for Browser Script

Method	Description	Format
BusObject Method for a Business Component	Returns the name of the business object that the business component references.	var busComp; var busObject; busObject = busComp.BusObject();
GetFieldValue Method for a Business Component	Returns the value of a field from the current record of a business component.	var busComp; var value; value = busComp.GetFieldValue(*fieldName*);

Table 105. Summary of Business Component Methods for Browser Script

Method	Description	Format

GetFormattedFieldValue Method for a Business Component	Returns a field value that is in the same format that the Siebel client uses.	`var busComp;`
		`var sValue;`
		`sValue =`
		`busComp.GetFormattedFieldValue`
		`(fieldName);`

GetSearchExpr Method for a Business Component	Returns the current search expression that is defined for the business component.	`var busComp;`
		`var sExpr;`
		`sExpr = busComp.GetSearchExpr();`

GetSearchSpec Method for a Business Component	Returns the search specification for a field.	`var busComp;`
		`var sSpec;`
		`sSpec =`
		`busComp.GetSearchSpec(fieldName);`

InvokeMethod Method for a Business Component	Calls a method.	`var busComp;`
		`var sReturn;`
		`sReturn =`
		`busComp.InvokeMethod(methodName,`
		`methodArg1, methodArg2, ...,`
		`methodArgn);`

Name Method for a Business Component	Returns the name of a business component.	`var busComp;`
		`var sName;`
		`sName = busComp.Name();`

SetFieldValue Method for a Business Component	Sets a new value for a field in the current record of a business component.	`var busComp;`
		`busComp.SetFieldValue(fieldName,`
		`fieldValue);`

SetFormattedFieldValue Method for a Business Component	Sets the new value to a field for the current record of a business component.	`var busComp;`
		`busComp.SetFormattedFieldValue`
		`(fieldName, fieldValue);`

UndoRecord Method for a Business Component	Reverses any unsaved modifications that Siebel CRM has made on a record.	`var busComp;` `busComp.UndoRecord();`
WriteRecord Method for a Business Component	Saves to the Siebel database any modifications made to the current record.	`var busComp;` `busComp.WriteRecord();`

Business Component Events for Browser Script

Table 106 describes a summary of the business component events you can use in Browser Script.

Table 106. Summary of Business Component Events for Browser Script

Event	Description	Format
BusComp_PreSetFieldValue Event	Called if the user modifies a value in the Siebel client.	BusComp_PreSetFieldValue (*fieldName, value*)

Business Object Methods for Browser Script

Table 107 describes a summary of the business object methods you can use in Browser Script.

Table 107. Summary of Business Object Methods for Browser Script

Method	Description	Format
GetBusComp Method for a Business Object	Returns the name of a business component.	var busObject; var busComp; busComp = busObject.GetBusComp(*busCompName*);
Name Method for a Business Object	Returns the name of a business object.	Var sName; var busObject; sName = busObject.Name();

Business Service Methods for Browser Script

Table 108 describes a summary of the business service methods you can use in Browser Script.

Table 108. Summary of Business Service Methods for Browser Script

Method	Description	Format
GetNextProperty Method for a Business Service	Returns the name of the next property of a business service.	var svc; var sName = svc.GetNextProperty();
GetProperty Method for a Business Service	Returns the value of a property.	var svc; var value; value = svc.GetProperty(*name*);

Table 108. Summary of Business Service Methods for Browser Script

Method	Description	Format
InvokeMethod Method for a Business Service	Calls a method on a business service.	```var svc = TheApplication().GetService("Business Service"); var inputPropSet = TheApplication().NewPropertySet(); svc.InvokeMethod(methodName, inputPropSet);```
Name Method for a Business Service	Returns the name of a business service.	```var svc; var name; name = svc.Name();```
PropertyExists Method for a Business Service	Returns a Boolean value that indicates if the property that the argument identifies exists.	```var svc; var bool; bool = svc.PropertyExists(name);```
RemoveProperty Method for a Business Service	Removes a property from a business service.	```var svc; svc.RemoveProperty(name);```
SetProperty Method for a Business Service	Sets a value for a property of a business service.	```var svc; svc.SetProperty(name, value);```

Business Service Events for Browser Script

Table 109 describes a summary of the business service events you can use in Browser Script.

Table 109. Summary Business Service Events for Browser Script

Method	Description	Format
Service_InvokeMethod Event	Called after Siebel CRM calls the InvokeMethod method on a business service.	Service_InvokeMethod (methodName, input)
Service_PreCanInvokeMethod Event	Called before Siebel CRM calls the PreInvokeMethod event. It allows you to determine if the user possesses the authority to call the business service method.	Service_PreCanInvokeMethod (methodName)
Service_PreInvokeMethod Event	Called before Siebel CRM calls a method on a business service.	Service_PreInvokeMethod (methodName, inputPropSet)

Property Set Methods for Browser Script

Table 110 describes a summary of the property set methods you can use in Browser Script.

Table 110. Summary of Property Set Methods for Browser Script

Method	Description	Format

AddChild Method for a Property Set	Adds a child property set to a property set.	`var oPropSet;`
		`var iIndex;`
		`iIndex = oPropSet.AddChild(childObject);`

Copy Method for a Property Set	Returns a copy of a property set.	`var oPropSet1;`
		`var oPropSet2;`
		`oPropSet2 = oPropSet1.Copy();`

GetChild Method for a Property Set	Returns the index number of a child property set.	`var oPropSet;`
		`var oChildPropSet;`
		`oChildPropSet = oPropSet.GetChild(index);`

GetChildCount Method for a Property Set	Returns the number of child property sets that exist for a parent property set	`var oPropSet;`
		`var iCount;`
		`iCount = oPropSet.GetChildCount();`

GetFirstProperty Method for a Property Set	Returns the name of the first property in a property set.	`var oPropSet;`
		`var sPropName;`
		`sPropName = oPropSet.GetFirstProperty();`

GetNextProperty Method for a Property Set	Returns the name of the next property in a property set.	`var oPropSet;`
		`var sPropName;`
		`sPropName = oPropSet.GetNextProperty();`

GetProperty Method for a Property Set	Returns the value of a property.	`var oPropSet;`
		`var sValue;`
		`sValue = oPropSet.GetProperty(propName);`

GetPropertyCount Method for a Property Set	Returns the number of properties that exist in the current level in the hierarchy.	`var oPropSet;`
		`var iCount;`
		`iCount = oPropSet.GetPropertyCount();`

GetType Method for a Property Set	Returns the value of the type attribute of a property set.	`var oPropSet;`
		`var type;`
		`type = oPropSet.GetType();`

GetValue Method for a Property Set	Returns the value of the value attribute of a property set.	`var oPropSet;`
		`var sValue;`
		`sValue = oPropSet.GetValue();`

Table 110. Summary of Property Set Methods for Browser Script

Method	Description	Format
InsertChildAt Method for a Property Set	Inserts a child property set in a parent property set at a specific location.	`var oPropSet;` `oPropSet.InsertChildAt(childObject, index);`

PropertyExists Method for a Property Set	Returns a Boolean value that indicates if the property that the argument identifies exists.	`var oPropSet;`
		`var bool;`
		`bool = oPropSet.PropertyExists(propName);`

RemoveChild Method for a Property Set	Removes a child property set from a parent property set.	`var oPropSet;` `oPropSet.RemoveChild(index);`
RemoveProperty Method for a Property Set	Removes a property from a property set.	`var oPropSet;` `oPropSet.RemoveProperty(propName);`
Reset Method for a Property Set	Removes every property and child property set from a property set.	`var oPropSet;` `oPropSet.Reset();`
SetProperty Method for a Property Set	Sets a value in the property of a property set.	`var oPropSet;` `oPropSet.SetProperty(propName, propValue);`
SetType Method for a Property Set	Sets the value for the type attribute of a property set.	`var oPropSet;` `oPropSet.SetType(value);`
SetValue Method for a Property Set	Sets the value for the value attribute of a property set.	`var oPropSet;` `oPropSet.SetValue(value);`

Control Methods for Browser Script

Table 111 describes a summary of the control methods you can use in Browser Script.

Table 111. Summary of Control Methods for Browser Script

Method	Description	Format

Applet Method for a Control	Returns the name of the applet where a control resides.	`var oControl;`
		`var oApplet;`
		`oApplet = oControl.Applet();`

BusComp Method for a Control	Returns the name of the business component that an applet references. The control resides in this applet.	`var oControl;`
		`var busComp;`
		`busComp = oControl.BusComp();`

Table 111. Summary of Control Methods for Browser Script

Method	Description	Format

GetProperty Method for a Control	Returns the value of the property of a control.	`var oControl;`
		`var propVal;`
		`propVal = oControl.GetProperty(propName);`

GetValue Method for a Control	Returns the value of a control.	`var oControl;`
		`var sValue;`
		`sValue = oControl.GetValue();`

Name Method for a Control	Returns the name of a control.	`var oControl;`
		`var sName;`
		`sName = oControl.Name();`

SetProperty Method for a Control	Sets the visual properties of a control.	`var oControl;` `oControl.SetProperty(propName, propValue);`
SetValue Method for a Control	Sets the contents of a control.	`var oControl;` `oControl.SetValue(value);`

Document Object Model Events You Can Use

This topic describes Document Object Model events you can use.

Document Object Model Events for High Interactivity Mode

Table 112 lists the Document Object Model events you can use in high interactivity mode. For each control, you can use the following events:

■ OnFocus

■ OnBlur

Note that scriptable events are not available for List Column and Tree controls.

Table 112. Document Object Model Events You Can Use in High Interactivity

Control	Siebel Control Type	Description
Button	Native	None
CheckBox	Native	Rendered as Input Type is CHECKBOX.
Link	Native	Rendered through paired anchor tags or as INPUT TYPE is TEXT in edit mode.
List Column	Native	None
Mailto	Native	Rendered as anchor tags with HREF is mailto or as INPUT TYPE is TEXT in Edit mode.

Table 112. Document Object Model Events You Can Use in High Interactivity

Control	Siebel Control Type	Description
MiniButton	Native	None
Password	Native	Rendered as Input Type is password.
Text	Native	Rendered as INPUT TYPE is TEXT or as SELECT if attached to a picklist. If there is a pop-up window, then Siebel CRM renders it as an edit box plus a button.
TextArea	Native	Rendered as TEXTAREA.
Tree	Native	None
URL	Native	Rendered through paired anchor tags with an HREF equal to the underlying field value or as INPUT TYPE is TEXT in edit mode.

You cannot access a Siebel object from a Document Object Model event. Business components and applets are examples of Siebel objects.

You can typically call code in the General section from anywhere in an object. However, you cannot call code written in the General section from a Document Object Model event.

To associate a script with the control_OnClick event in high interactivity mode only, use the Applet_PreInvokeMethod event that is associated with the applet. For more information, see "Using a MiniButton Control to Call a Custom Method" on page 78.

Document Object Model Events for Standard Interactivity Mode

Table 113 lists the Document Object Model events and template modes you can use in standard interactivity mode. The Type property for each control uses a type that is native to the browser.

Table 113. Document Object Model Events You Can Use in Standard Interactivity

Control	Event	Description
Button	You can use the following events: ■ OnBlur (Base/Edit) ■ OnFocus (Base/Edit) ■ OnMouseOut (Base/Edit) ■ OnMouseOver (Base/Edit)	Not applicable
CheckBox	You can use the following events: ■ OnBlur (Base/Edit) ■ OnChange (Edit) ■ OnFocus (Edit) ■ OnMouseOut (Edit) ■ OnMouseOver(Edit)	Siebel CRM renders a CheckBox control in the following ways: ■ In Base mode, as a Y or N text value ■ In Edit mode, as Input Type is CHECKBOX
Link	You can use the following events: ■ OnBlur (Base/Edit) ■ OnClick (Base/Edit) ■ OnFocus (Base/Edit) ■ OnMouseOut (Base/Edit) ■ OnMouseOver (Base/Edit)	Siebel CRM renders a Link control in the following ways: ■ In Base mode, through paired anchor tags ■ In Edit mode, as INPUT TYPE is TEXT
Mailto	You can use the following events: ■ OnBlur (Base/Edit) ■ OnChange (Edit) ■ OnFocus (Base/Edit) ■ OnMouseOut (Base/Edit) ■ OnMouseOver (Base/Edit)	Siebel CRM renders a Mailto control in the following ways: ■ In Base mode, as anchor tags with HREF is mailto ■ In Edit mode, as INPUT TYPE is TEXT

Table 113. Document Object Model Events You Can Use in Standard Interactivity

Control	Event	Description
MiniButton	You can use the following events: ■ OnBlur (Base/Edit) ■ OnClick (Base/Edit) ■ OnFocus (Base/Edit) ■ OnMouseOut (Base/Edit) ■ OnMouseOver (Base/Edit)	Not applicable
Password	You can use the following events: ■ OnBlur (Edit) ■ OnChange (Edit) ■ OnFocus (Edit) ■ OnMouseOut (Edit) ■ OnMouseOver (Edit)	Siebel CRM renders a Password control in Edit mode as Input type is password.
Text	You can use the following events: ■ OnBlur (Edit) ■ OnChange (Edit) ■ OnFocus (Edit) ■ OnMouseOut (Edit) ■ OnMouseOver (Edit)	Siebel CRM renders a Text control in the following ways: ■ In base mode, as plain text, unless a pop-up window is associated with the control. ■ In Edit mode, as INPUT TYPE is TEXT, unless the control is attached to a picklist. If the control is attached to a picklist, then Siebel CRM renders it as INPUT TYPE is SELECT.
TextArea	You can use the following events: ■ OnBlur (Edit) ■ OnChange (Edit) ■ OnFocus (Edit) ■ OnMouseOut (Base/Edit) ■ OnMouseOver (Edit)	Siebel CRM renders a TEXTAREA control in the following ways: ■ In base mode, as plain text, unless a pop-up window is associated with the control ■ In Edit mode, as INPUT TYPE is TEXTAREA
URL	You can use the following events: ■ OnBlur (Base/Edit) ■ OnChange (Edit) ■ OnFocus (Base/Edit) ■ OnMouseOut (Base/Edit) ■ OnMouseOver (Base/Edit)	Siebel CRM renders a TEXTAREA control in the following ways: ■ In Base mode, through paired anchor tags with an HREF that is the underlying field value ■ In Edit mode, as INPUT TYPE is TEXT

Table 113. Document Object Model Events You Can Use in Standard Interactivity

Control	Event	Description
Tree	Scriptable events are not available.	Not applicable
List Column		

7   Siebel VB Quick Reference

This chapter describes summary information for Siebel VB. It includes the following topics:

- Applet Methods for Siebel VB on page 339
- Web Applet Events for Siebel VB on page 340
- Application Methods for Siebel VB on page 341
- Application Events for Siebel VB on page 343
- Business Component Methods for Siebel VB on page 344
- Business Component Events for Siebel VB on page 348
- Business Object Methods for Siebel VB
- Business Service Methods for Siebel VB on page 351
- Business Service Events for Siebel VB on page 352
- Property Set Methods for Siebel VB on page 353
- Miscellaneous Methods for Siebel VB on page 354

Applet Methods for Siebel VB

Table 114 describes a summary of the applet methods you can use with Siebel VB.

Table 114. Summary of Applet Methods for Siebel VB

Method	Description	Format

BusComp Method for an Applet	Returns the name of the business component that an applet references.	`Dim oApplet as Applet`
		`Dim oBusComp as BusComp`
		`Set oBusComp = oApplet.BusComp`

BusObject Method for an Applet	Returns the name of the business object for the business component that the applet references.	`Dim oApplet as Applet`
		`Dim oBusObject as BusObject`
		`Set oBusObject = oApplet.BusObject`

InvokeMethod Method for an Applet	Calls a specialized method.	`Dim oApplet as Applet`
		`oApplet.InvokeMethod methodName as`
		`String, methArg1, methArg2,`
		`methArgN as String or StringArray`

Name Method for an Applet	Returns the name of an applet.	`Dim oApplet as Applet`
		`Dim sApplet as String`
		`sApplet = oApplet.Name`

Web Applet Events for Siebel VB

Table 115 describes a summary of web applet events you can use with Siebel VB.

Table 115. Summary of Web Applet Events for Siebel VB

Event	Description	Format
WebApplet_InvokeMethod Event	Called after Siebel CRM runs a specialized method on the Web applet.	WebApplet_InvokeMethod(*MethodName* as String)
WebApplet_PreCanInvokeMethod Event	Called before Siebel CRM calls the PreInvokeMethod event, allowing you to determine if the user possesses the authority to call the applet method.	WebApplet_PreCanInvokeMethod (*MethodName* as String, CanInvoke as String)
WebApplet_PreInvokeMethod Event	Called before Siebel CRM calls a specialized method for the Web applet or before it calls a custom method through oWebApplet.Invoke Method.	WebApplet_PreInvokeMethod (*MethodName* as String) As Integer
WebApplet_Load Event	Called immediately after Siebel CRM loads an applet.	WebApplet_Load
WebApplet_ShowControl Event	Allows a script to modify the HTML that the Siebel Web Engine creates when it renders a control on a Web page in a Siebel application that runs in standard interactivity mode.	WebApplet_ShowControl (*controlName* as String, *property* as String, *mode* as String, HTML as String)
WebApplet_ShowListColumn Event	Allows a script to modify the HTML that the Siebel Web Engine creates when it renders a list column on a Web page in a Siebel application that runs in standard interactivity mode.	WebApplet_ShowListColumn (*columnName* as String, *property* as String, *mode* as String, HTML as String)

Application Methods for Siebel VB

Table 116 describes a summary of the application methods you can use with Siebel VB. It does not include object interface methods that are not called directly from an application object instance. For information about methods that are called with the InvokeMethod method on the application object, see "LoadObjects Method for an Application" on page 150.

Table 116. Summary of Application Methods for Siebel VB

Method	Description	Format

ActiveBusObject Method for an Application	Returns the name of the business object of the active view.	`Dim oApplication as Application`
		`Dim oBusObject as BusObject`
		`Set oBusObject =`
		`oApplication.ActiveBusObject`

ActiveViewName Method for an Application	Returns the name of the active view.	`Dim oApplication as Application`
		`Dim sView as String`
		`sView = oApplication.ActiveViewName`

CurrencyCode Method for an Application	Returns the currency code that is associated with the division of the user position	`Dim oApplication as Application`
		`Dim sCur as String`
		`sCur = oApplication.CurrencyCode`

GetBusObject Method for an Application	Creates a new instance of a business object.	`Dim oApplication as Application`
		`Dim oBusObject as BusObject`
		`set oBusObject =`
		`oApplication.GetBusObject (`*busobject* `as`
		`String)`

GetProfileAttr Method for an Application	Returns the name of an attribute in a user profile.	`Dim oApplication as Application`
		`Dim sAttr as String`
		`SAttr = oApplication.GetProfileAttr(`*name*
		`as String)`

GetService Method for an Application	Locates a business service. If this business service is not already running, then Siebel CRM starts it.	`Dim oApplication as Application`
		`Dim oService as Service`
		`set oService =`
		`oApplication.GetService(`*serviceName* `as`
		`String)`

GetSharedGlobal Method for an Application	Returns the shared global variables.	`Dim oApplication as Application`
		`Dim sName as String`
		`sName =`
		`Application.GetSharedGlobal (`*varName* `as`
		`String)`

Table 116. Summary of Application Methods for Siebel VB

Method	Description	Format
GotoView Method for an Application	Does the following: ■ Deactivates any business object, business component, applet, or control that is active. ■ Activates a view. ■ Activates the primary applet of the view and the business component that this applet references. ■ Activates the first tab sequence control of the primary applet	`Dim oApplication as Application` `oApplication.GotoView` *viewName* `as String[,` *BusinessObjectName* `as BusObject]`
InvokeMethod Method for an Application	Calls a method.	`Dim oApplication as Application` `oApplication.InvokeMethod(`*methodName* `as String,` *methArg1, methArg2, methArgN* `as String or StringArray)`

LoginId Method for an Application	Returns the login ID of the user who started the Siebel application.	`Dim oApplication as Application`
		`Dim sID as String`
		`iID = oApplication.LoginId`

LoginName Method for an Application	Returns the login name of the user who started the Siebel application.	`Dim oApplication as Application`
		`Dim sUser as String`
		`sUser = oApplication.LoginName`

NewPropertySet Method for an Application	Creates a new property set.	`Dim oApplication as Application`
		`Dim oPropSet as PropertySet`
		`oPropSet = oApplication.NewPropertySet`

PositionId Method for an Application	Returns the name of the current user position.	`Dim oApplication as Application`
		`Dim sRow as String`
		`sRow = oApplication.PositionId`

PositionName Method for an Application	Returns the name of the current user position.	`Dim oApplication as Application`
		`Dim sPosition as String`
		`sPosition = oApplication.PositionName`

RaiseError Method for an Application	Sends a scripting error message to the browser. To determine the error text, Siebel CRM uses a key to look up the current language.	`Dim oApplication as Application` `oApplication.RaiseError keyValue as String, param1 as String, ...`
RaiseErrorText Method for an Application	Sends a scripting error message to the browser.	`Dim oApplication as Application` `oApplication.RaiseErrorText message as String`

Table 116. Summary of Application Methods for Siebel VB

Method	Description	Format
SetPositionId Method for an Application	Sets the active position to a Position Id.	`Dim oApplication as Application` `oApplication.SetPositionId posId as string`
SetPositionName Method for an Application	Sets the active position to a position name.	`Dim oApplication as Application` `oApplication.SetPositionName posName as string`
SetProfileAttr Method for an Application	Sets a value for an attribute in a user profile.	`Dim oApplication as Application` `oApplication.SetProfileAttr name as String, value as String`
SetSharedGlobal Method for an Application	Sets a shared global variable.	`Dim oApplication as Application` `oApplication.SetSharedGlobal varName as String, value as String`
Trace Method for an Application	Appends a message to the trace file.	`Dim oApplication as Application` `oApplication.Trace message as String`
TraceOff Method for an Application	Turns off tracing.	`Dim oApplication as Application` `oApplication.TraceOff`
TraceOn Method for an Application	Turns on tracing.	`Dim oApplication as Application` `oApplication.TraceOn filename as String, type as String, selection as String`

Application Events for Siebel VB

Table 117 describes a summary of the application events you can use with Siebel VB.

Table 117. Summary of Application Events for Siebel VB

Event	Description	Format
Application_Close Event	Allows scripts to perform cleanup, before the Siebel application closes.	`Application_Close`
Application_Navigate Event	Called after the user navigates to a view.	`Application_Navigate`
Application_InvokeMethod Event	Called after a specialized method is called.	`Application_InvokeMethod (MethodName as String)`
Application_PreInvokeMethod Event	Called before an applet menu or the InvokeMethod method calls a specialized method.	`Application_PreInvokeMethod (MethodName as String) As Integer`

Table 117. Summary of Application Events for Siebel VB

Event	Description	Format
Application_PreNavigate Event	Called before the Siebel application displays the view where the user navigates.	Application_PreNavigate (*DestViewName* As String, *DestBusObjName* As String)
Application_Start Event	Called when the Siebel client starts and again when it displays the client interface for the first time.	Application_Start(*commandLine* as String)

Business Component Methods for Siebel VB

Table 118 describes a summary of the business component methods you can use with Siebel VB. It does not include object interface methods that are not called directly from a business component. For information about methods that you can call with the InvokeMethod method on the business component, see "Business Component Invoke Methods" on page 252.

Table 118. Summary of Business Component Methods for Siebel VB

Method	Description	Format
ActivateField Method for a Business Component	Activates a field.	Dim oBusComp as BusComp oBusComp.ActivateField *fieldName* as String
ActivateMultipleFields Method for a Business Component	Activates multiple fields.	Dim oBusComp as BusComp oBusComp.ActivateMultipleFields *oPropSet* as PropertySet
Associate Method for a Business Component	Creates a new many-to-many relationship for the parent object through an association business component.	Dim oBusComp as BusComp oBusComp.Associate *whereIndicator* as Integer
BusObject Method for a Business Component	Returns the name of the business object that the business component references.	Dim oBusComp as BusComp Dim oBusObject as BusObject Set oBusObject = oBusComp.BusObject
ClearToQuery Method for a Business Component	Clears the current query but does not clear sort specifications on the business component.	Dim oBusComp as BusComp oBusComp.ClearToQuery
DeactivateFields Method for a Business Component	Deactivates the fields that are currently active from the SQL query statement of a business component.	Dim oBusComp as BusComp oBusComp.DeactivateFields

Table 118. Summary of Business Component Methods for Siebel VB

Method	Description	Format
DeleteRecord Method for a Business Component	Removes the current record from a business component.	`Dim oBusComp as BusComp` `oBusComp.DeleteRecord`

ExecuteQuery Method for a Business Component	Returns a set of business component records.	`Dim oBusComp as BusComp`
		`oBusComp.ExecuteQuery cursorMode`
		`as Integer`

ExecuteQuery2 Method for a Business Component	Returns a set of business component records. Allows you to control the number of records Siebel CRM returns.	`Dim oBusComp as BusComp`
		`oBusComp.ExecuteQuery2 cursorMode`
		`as Integer, ignoreMaxCursorSize as`
		`Integer`

FirstRecord Method for a Business Component	Moves the record pointer to the first record in a business component, making that record the current record.	`Dim oBusComp as BusComp`
		`Dim iIsRecord as Integer`
		`iIsRecord = oBusComp.FirstRecord`

FirstSelected Method for a Business Component	Makes the first record of the multiple selection in a business component active.	`Dim oBusComp as BusComp`
		`Dim iIsMultipleSection as Integer`
		`iIsMultipleSelection =`
		`oBusComp.FirstSelected`

GetAssocBusComp Method for a Business Component	Returns the name of the association business component.	`Dim oBusComp as BusComp`
		`Dim AssocBusComp as BusComp`
		`Set AssocBusComp =`
		`oBusComp.GetAssocBusComp`

GetFieldValue Method for a Business Component	Returns the value of a field from the current record of a business component.	`Dim oBusComp as BusComp`
		`Dim sValue as String`
		`sValue =`
		`oBusComp.GetFieldValue(FieldName`
		`as String)`

GetFormattedFieldValue Method for a Business Component	A field value that is in the same format that the Siebel client uses.	`Dim oBusComp as BusComp`
		`Dim sValue as String`
		`sValue =`
		`oBusComp.GetFormattedFieldValue(F`
		`ieldName as String)`

GetMultipleFieldValues Method for a Business Component	Returns values for the fields specified in a property set.	`Dim oBusComp as BusComp`
		`oBusComp.GetMultipleFieldValues`
		`oFields as PropertySet, oValues as`
		`PropertySet`

GetMVGBusComp Method for a Business Component	Returns the multivalue group business component associated a business component field.	`Dim oBusComp as BusComp`
		`Dim MvgBusComp as BusComp`
		`set MvgBusComp =`
		`oBusComp.GetMVGBusComp(FieldName`
		`as String)`

Table 118. Summary of Business Component Methods for Siebel VB

Method	Description	Format

GetNamedSearch Method for a Business Component	Returns the name of a search specification.	`Dim oBusComp as BusComp`
		`Dim sValue as String`
		`sValue =`
		`oBusComp.GetNamedSearch(`*SearchName* `as String)`

GetPicklistBusComp Method for a Business Component	Returns the name of the pick business component that is associated with a field in the current business component.	`Dim oBusComp as BusComp`
		`Dim pickBusComp as BusComp`
		`Set pickBusComp =`
		`oBusComp.GetPicklistBusComp(`*Field Name* `as String)`

GetSearchExpr Method for a Business Component	Returns the current search expression that is defined for a business component.	`Dim oBusComp as BusComp`
		`Dim sExpr as String`
		`sExpr = oBusComp.GetSearchExpr`

GetSearchSpec Method for a Business Component	Returns the search specification for a field.	`Dim oBusComp as BusComp`
		`Dim sSpec as String`
		`sSpec =`
		`oBusComp.GetSearchSpec(`*FieldName* `as String)`

GetSortSpec Method for a Business Component	Returns the sort specification for a business component.	`Dim sSortSpec as String` `sSortSpec = GetSortSpec`

GetUserProperty Method for a Business Component	Returns the value of a user property.	`Dim oBusComp as BusComp`
		`Dim sValue as String`
		`sValue =`
		`oBusComp.GetUserProperty(`*property Name* `as String)`

GetViewMode Method for a Business Component	Returns the current visibility mode for a business component.	`Dim oBusComp as BusComp`
		`Dim iMode as Integer`
		`iMode = oBusComp.GetViewMode`

InvokeMethod Method for a Business Component	Calls a method.	`Dim oBusComp as BusComp` `oBusComp.InvokeMethod(`*methodName* `as String,` *methArg1, methArg2, methArgN* `as String or StringArray)`

LastRecord Method for a Business Component	Moves the record pointer to the last record in a business component.	`Dim oBusComp as BusComp`
		`Dim iReturn as Integer`
		`iReturn = oBusComp.LastRecord`

Name Method for a Business Component	Returns the name of a business component.	`Dim oBusComp as BusComp`
		`Dim sName as String`
		`sName = oBusComp.Name`

NewRecord Method for a Business Component	Adds a new record to a business component.	`Dim oBusComp as BusComp` `oBusComp.NewRecord` *whereIndicator* `as Integer`

Table 118. Summary of Business Component Methods for Siebel VB

Method	Description	Format

NextRecord Method for a Business Component	Moves the record pointer to the next record in a business component, making that record the current record.	`Dim oBusComp as BusComp`
		`Dim iReturn as Integer`
		`iReturn = oBusComp.NextRecord`

NextSelected Method for a Business Component	Makes the next record of the current multiple selection the active record.	`Dim oBusComp as BusComp`
		`Dim iReturn as Integer`
		`iReturn = oBusComp.NextSelected`

ParentBusComp Method for a Business Component	Returns the name of the parent business component.	`Dim oBusComp as BusComp`
		`Dim parentBusComp as BusComp`
		`Set parentBusComp =`
		`oBusComp.ParentBusComp`

Pick Method for a Business Component	Places the currently chosen record in a pick business component into the appropriate fields of the parent business component.	`Dim oBusComp as BusComp` `oBusComp.Pick`

PreviousRecord Method for a Business Component	Moves the record pointer to the previous record in a business component.	`Dim oBusComp as BusComp`
		`Dim iReturn as Integer`
		`iReturn = oBusComp.PreviousRecord`

RefineQuery Method for a Business Component	Refines a query.	`Dim oBusComp as BusComp` `oBusComp.RefineQuery`
SetFieldValue Method for a Business Component	Sets a new value for a field in the current record of a business component.	`Dim oBusComp as BusComp` `oBusComp.SetFieldValue` *FieldName* `as String,` *FieldValue* `as String`
SetFormattedFieldValue Method for a Business Component	Sets the new value to a field for the current record of a business component.	`Dim oBusComp as BusComp` `oBusComp.SetFormattedFieldValue` *FieldName* `as String,` *FieldValue* `as String`
SetMultipleFieldValues Method for a Business Component	Sets new values in the fields of the current record of a business component.	`Dim oBusComp as BusComp` `oBusComp.SetMultipleFieldValues` *oPropSet* `as PropertySet`
SetNamedSearch Method for a Business Component	Sets the named search specification on a business component.	`Dim oBusComp as BusComp` `oBusComp.SetNamedSearch` *searchName* `as String,` *searchSpec* `as String`
SetSearchExpr Method for a Business Component	Sets a search expression for a business component rather than for each field.	`Dim oBusComp as BusComp` `oBusComp.SetSearchExpr` *searchSpec* `as String`
SetSearchSpec Method for a Business Component	Sets the search specification for a field.	`Dim oBusComp as BusComp` `oBusComp.SetSearchSpec` *fieldName* `as String,` *searchSpec* `as String`

Table 118. Summary of Business Component Methods for Siebel VB

Method	Description	Format
SetSortSpec Method for a Business Component	Sets the sort specification for a business component.	`Dim oBusComp as BusComp` `oBusComp.SetSortSpec` *sortSpec* `as String`
SetUserProperty Method for a Business Component	Sets the value of a user property in a business component.	`Dim oBusComp as BusComp` `oBusComp.SetUserProperty` *propertyName* `as String,` *newValue* `as String`
SetViewMode Method for a Business Component	Sets the visibility type for a business component.	`Dim oBusComp as BusComp` `oBusComp.SetViewMode` *viewMode* `as Integer`
UndoRecord Method for a Business Component	Reverses any unsaved modifications that Siebel CRM has made on a record.	`Dim oBusComp as BusComp` `oBusComp.UndoRecord`
WriteRecord Method for a Business Component	Saves to the Siebel database any modifications made to the current record.	`Dim oBusComp as BusComp` `oBusComp.WriteRecord`

Business Component Events for Siebel VB

Table 119 describes a summary of the business component events you can use with Siebel VB.

Table 119. Summary of Business Component Events for Siebel VB

Event	Description	Format
BusComp_Associate Event	Called if the user adds a business component record to create an association.	`BusComp_Associate`
BusComp_ChangeRecord Event	Called if a business component record becomes the current record.	`BusComp_ChangeRecord`
BusComp_CopyRecord Event	Called if the user copies a business component record, and if the user makes this record the active record.	`BusComp_CopyRecord`
BusComp_DeleteRecord Event	Called if the user deletes a business component record.	`BusComp_DeleteRecord`
BusComp_InvokeMethod Event	Called if Siebel CRM calls the InvokeMethod method on a business component.	`BusComp_InvokeMethod (`*methodName* `as String)`

Table 119. Summary of Business Component Events for Siebel VB

Event	Description	Format
BusComp_NewRecord Event	Called if the user creates a business component record, and if the user makes this record the active record. You can use this event to set up default values for a field.	BusComp_NewRecord
BusComp_PreAssociate Event	Called if Siebel CRM detects that the user is about to add a business component record to create an association.	BusComp_PreAssociate
BusComp_PreCopyRecord Event	Called if Siebel CRM detects that the user is about to copy a business component record. You can use this event to perform precopy validation.	BusComp_PreCopyRecord
BusComp_PreDeleteRecord Event	Called if Siebel CRM detects that the user is about to delete a business component record. You can use this event to prevent the deletion or to perform any actions before Siebel CRM deletes the record.	BusComp_PreDeleteRecord
BusComp_PreGetFieldValue Event	Called if a user accesses a business component field.	BusComp_PreGetFieldValue (*FieldName* as String, *FieldValue* as String)

Table 119. Summary of Business Component Events for Siebel VB

Event	Description	Format
BusComp_PreInvokeMethod Event	Called if Siebel CRM calls a specialized method on a business component. Siebel CRM calls it before it calls this specialized method.	BusComp_PreInvokeMethod (*methodName* as String)
BusComp_PreNewRecord Event	Called if Siebel CRM detects that the user is about to create a new business component record. You can use this event to perform preinsert validation.	BusComp_PreNewRecord
BusComp_PreQuery Event	Siebel CRM calls the BusComp_PreQuery event before it runs a query. You can use this event to modify the search criteria or to restrict Siebel CRM from running certain queries.	BusComp_PreQuery
BusComp_PreSetFieldValue Event	Siebel CRM calls this event after the user modifies a field value or after a call to the SetFieldValue method occurs. This event allows you to use custom validation before Siebel CRM applies predefined validation.	BusComp_PreSetFieldValue (*FieldName* as String, *FieldValue* as String)
BusComp_PreWriteRecord Event	Called before Siebel CRM writes a record to the Siebel database.	BusComp_PreWriteRecord
BusComp_Query Event	Called after Siebel CRM completes a query but before it displays the query results.	BusComp_Query
BusComp_SetFieldValue Event	Called if Siebel CRM sends a value to a business component from the Siebel client or through a call to the SetFieldValue method.	BusComp_SetFieldValue (*fieldName* as String)
BusComp_WriteRecord Event	Called after Siebel CRM saves the record to the Siebel database.	BusComp_WriteRecord

Business Object Methods for Siebel VB

Table 120 describes a summary of business object methods you can use with Siebel VB.

Table 120. Summary of Business Object Methods for Siebel VB

Method	Description	Format

GetBusComp Method for a Business Object	Returns the name of a business component instance.	`Dim oBusObject as BusObject`
		`Dim oBusComp as BusComp`
		`set oBusComp =`
		`BusObject.GetBusComp(`*BusCompName*
		`as String)`

Name Method for a Business Object	Returns the name of a business object.	`Dim oBusObject as BusObject`
		`Dim sName as String`
		`sName = oBusObject.Name`

Business Service Methods for Siebel VB

Table 121 describes a summary of the business service methods you can use with Siebel VB.

Table 121. Summary of Business Service Methods for Siebel VB

Method	Description	Format

GetFirstProperty Method for a Business Service	Returns the name of the first property that is defined for a business service.	`Dim oService as Service`
		`Dim sName as String`
		`sName = oService.GetFirstProperty`

GetNextProperty Method for a Business Service	Returns the name of the next property of a business service.	`Dim oService as Service`
		`Dim sName as String`
		`sName = oService.GetNextProperty`

GetProperty Method for a Business Service	Returns the value of a property.	`Dim oService as Service`
		`Dim sValue as String`
		`sValue =`
		`oService.GetProperty(`*propName* `as`
		`String)`

InvokeMethod Method for a Business Service	Calls a method on a business service.	`Dim oService as Service`
		`oService.InvokeMethod(methodName`
		`as String, InputArguments as`
		`PropertySet, OutputArguments as`
		`PropertySet)`

Name Method for a Business Service	Returns the name of a business service.	`Dim oService as Service`
		`Dim sName as String`
		`sName = oService.Name`

Table 121. Summary of Business Service Methods for Siebel VB

Method	Description	Format

PropertyExists Method for a Business Service	Returns a Boolean value that indicates if the property that the argument identifies exists.	`Dim oService as Service`
		`Dim iReturn as Boolean`
		`iReturn =`
		`oService.PropertyExists(`*propName*
		`as String)`

RemoveProperty Method for a Business Service	Removes a property from a business service.	`Dim oService as Service`
		`oService.RemoveProperty` *propName*
		`as String`

SetProperty Method for a Business Service	Sets a value for a property of a business service.	`Dim oService as Service`
		`oService.SetProperty` *propName* `as`
		`String,` *propValue* `as String`

Business Service Events for Siebel VB

Table 122 describes a summary of business service events you can use with Siebel VB.

Table 122. Summary of Business Service Events for Siebel VB

Method	Description	Format
Service_InvokeMethod Event	Siebel CRM calls this event after it calls the InvokeMethod method.	`Service_InvokeMethod` `(`*methodName* `as String)`

Service_PreCanInvokeMethod Event	Siebel CRM calls this event before it calls the PreInvokeMethod event. This configuration allows you to determine if the user possesses the authority to call the business service method.	`Service_PreCanInvokeMethod`
		`(`*methodName* `as String,`
		`CanInvoke As String)`

Service_PreInvokeMethod Event	Siebel CRM calls this event before it calls a specialized method on a business service.	`Service_PreInvokeMethod`
		`(`*methodName* `as String,`
		`Inputs as PropertySet,`
		`Outputs as PropertySet)`

Property Set Methods for Siebel VB

Table 123 describes a summary of the property set methods you can use with Siebel VB.

Table 123. Summary of Property Set Methods for Siebel VB

Method	Description	Format
AddChild Method for a Property Set	Adds a child property set to a property set.	`Dim oPropSet as PropertySet` `oPropSet.AddChild childObject as PropertySet`

Copy Method for a Property Set	Returns a copy of a property set.	`Dim oPropSet1 as PropertySet`
		`Dim oPropSet2 as PropertySet`
		`set oPropSet2 = oPropSet1.Copy`

GetChild Method for a Property Set	Returns a child property set of a property set.	`Dim oPropSet as PropertySet`
		`Dim childPropSet as SiebelPropertySet`
		`set childPropSet = oPropSet.GetChild(index as Long)`

GetChildCount Method for a Property Set	Returns the number of child property sets that exist for a parent property set.	`Dim oPropSet as PropertySet`
		`Dim iCount as Integer`
		`iCount = oPropSet.GetChildCount`

GetFirstProperty Method for a Property Set	Returns the name of the first property in a property set.	`Dim oPropSet as PropertySet`
		`Dim sPropName as String`
		`sPropName = oPropSet.GetFirstProperty`

GetNextProperty Method for a Property Set	Returns the name of the next property in a property set.	`Dim oPropSet as PropertySet`
		`Dim sPropName as String`
		`sPropName = oPropSet.GetNextProperty`

GetProperty Method for a Property Set	Returns the value of a property.	`Dim oPropSet as PropertySet`
		`Dim sPropVal as String`
		`sPropVal = oPropSet.GetProperty(propName as String)`

GetPropertyCount Method for a Property Set	Returns the number of properties that exist in the current level in the hierarchy.	`Dim oPropSet as PropertySet`
		`Dim count as Long`
		`count = oPropSet.GetPropertyCount`

GetType Method for a Property Set	Returns the value of the type attribute of a property set.	`Dim oPropSet as PropertySet`
		`Dim sTypeVal as String`
		`sTypeVal = oPropSet.GetType`

GetValue Method for a Property Set	Returns the value stored in the value attribute of a property set.	`Dim oPropSet as PropertySet`
		`Dim sValVal as String`
		`sValVal = oPropSet.GetValue`

InsertChildAt Method for a Property Set	Inserts a child property set in a parent property set at a specific location.	`Dim oPropSet as PropertySet` `oPropSet.InsertChildAt childObject as SiebelPropertySet, index as Integer`

Table 123. Summary of Property Set Methods for Siebel VB

Method	Description	Format
PropertyExists Method for a Property Set	Returns a Boolean value that indicates if the property that the argument identifies exists.	`Dim oPropSet as PropertySet` `oPropSet.PropertyExists(propName as String)`
RemoveChild Method for a Property Set	Removes a child property set from a parent property set.	`Dim oPropSet as PropertySet` `oPropSet.RemoveChild index as Integer`
RemoveProperty Method for a Property Set	Removes a property from a property set.	`Dim oPropSet as PropertySet` `oPropSet.RemoveProperty propName as String`
Reset Method for a Property Set	Removes every property and child property set from a property set.	`Dim oPropSet as PropertySet` `oPropSet.Reset`
SetProperty Method for a Property Set	Sets a value in the property of a property set.	`Dim oPropSet as PropertySet` `oPropSet.SetProperty propName as String, propValue as String`
SetType Method for a Property Set	Sets a data value for the type attribute of a property set.	`Dim oPropSet as PropertySet` `oPropSet.SetType value as String`
SetValue Method for a Property Set	Sets a data value for the value attribute of a property set.	`Dim oPropSet as PropertySet` `oPropSet.SetValue value as String`

Miscellaneous Methods for Siebel VB

Table 124 describes a summary of miscellaneous methods you can use with Siebel VB.

Table 124. Summary of Miscellaneous Methods for Siebel VB

Method	Description	Format
TheApplication Method	Returns the name of an application object.	`TheApplication`

8 Siebel eScript Quick Reference

This chapter describes summary information for Siebel eScript. It includes the following topics:

- Applet Methods for Siebel eScript on page 355
- Web Applet Events for Siebel eScript on page 356
- Application Methods for Siebel eScript on page 357
- Application Events for Siebel eScript on page 359
- Business Component Methods for Siebel eScript
- Business Component Events for Siebel eScript on page 364
- Business Object Methods for Siebel eScript on page 366
- Business Service Methods for Siebel eScript on page 367
- Business Service Events for Siebel eScript on page 368
- Property Set Methods for Siebel eScript
- Miscellaneous Methods for Siebel eScript on page 370

The ST eScript engine is the default Siebel eScript scripting engine in Siebel CRM version 8.0 and later. For information about format differences between it and the traditional (T) engine, see *Siebel eScript Language Reference.*

Applet Methods for Siebel eScript

Table 125 describes a summary of the applet methods you can use with Siebel eScript.

Table 125. Summary of Applet Methods for Siebel eScript

Method	Description	Format

BusComp Method for an Applet	Returns the name of the business component that an applet references.	`var applet;`
		`var myBusComp;`
		`myBusComp = applet.BusComp();`

BusObject Method for an Applet	Returns the name of the business object for the business component that an applet references.	`var applet;`
		`var busObject;`
		`busObject = applet.BusObject();`

Table 125. Summary of Applet Methods for Siebel eScript

Method	Description	Format

InvokeMethod Method for an Applet	Calls a specialized method.	`var applet;`
		`applet.InvokeMethod(`*methodName*`, `*methodArg1*`, `*methodArg2*`, …,`
		methodArgn`);`

Name Method for an Applet	Returns the name of an applet.	`var applet;`
		`var sApplet;`
		`sApplet = applet.Name();`

Web Applet Events for Siebel eScript

Table 126 describes a summary of web applet events you can use with Siebel eScript.

Table 126. Summary of Web Applet Events for Siebel eScript

Event	Description	Format
WebApplet_InvokeMethod Event	Siebel CRM calls this event after a specialized method on the Web applet runs.	`WebApplet_InvokeMethod` `(`*MethodName*`)`
WebApplet_Load Event	Siebel CRM calls this event immediately after it loads an applet.	`WebApplet_Load`
WebApplet_PreCanInvokeMethod Event	Called before Siebel CRM calls the PreInvokeMethod event, allowing you to determine if the user possesses the authority to call the applet method.	`WebApplet_PreCanInvokeMethod` `(`*MethodName*`, &`*CanInvoke*`)`
WebApplet_PreInvokeMethod Event	Siebel CRM calls this event before it calls a specialized method for the Web applet or a custom method that it calls through the oWebApplet object of the InvokeMethod method.	`WebApplet_PreInvokeMethod` `(`*MethodName*`)`

WebApplet_ShowControl Event	Allows a script to modify the HTML that the Siebel Web Engine creates when it renders a control on a Web page in a Siebel application that runs in standard interactivity mode.	`WebApplet_ShowControl`
		`(`*controlName*`, `*property*`, `*mode*`,`
		`&`*HTML*`)`

WebApplet_ShowListColumn Event	Allows a script to modify the HTML that the Siebel Web Engine creates when it renders a list column on a Web page in a Siebel application that runs in standard interactivity mode.	`WebApplet_ShowListColumn`
		`(`*columnName*`, `*property*`, `*mode*`,`
		`&`*HTML*`)`

Application Methods for Siebel eScript

Table 127 describes a summary of application methods you can use with Siebel eScript. It does not include object interface methods that Siebel CRM does not call directly from an application instance. For information about methods that Siebel CRM calls with the InvokeMethod method on the application, see "LoadObjects Method for an Application" on page 150.

Table 127. Summary of Application Methods for Siebel eScript

Method	Description	Format

ActiveBusObject Method for an Application	Returns the name of the business object that the active view references.	`var busObject;`
		`busObject = `
		`TheApplication().ActiveBusObject();`

ActiveViewName Method for an Application	Returns the name of the active view.	`var sView;`
		`sView = `
		`TheApplication().ActiveViewName();`

CurrencyCode Method for an Application	Returns the currency code that is associated with the division of the user position.	`var sCur;`
		`sCur = `
		`TheApplication().CurrencyCode();`

GetBusObject Method for an Application	Creates a new instance of a business object.	`var myBusObject;`
		`myBusObject = `
		`TheApplication().GetBusObject(`
		`BusObjectName);`

Name Method for an Application	Returns the name of the Siebel application.	`var name;` `name = TheApplication().Name();`

GetService Method for an Application	Locates a business service. If this business service is not already running, then Siebel CRM starts it.	`var Service;`
		`Service = `
		`TheApplication().GetService`
		`(serviceName);`

GetSharedGlobal Method for an Application	Returns the shared global variables.	`var sName;`
		`sName = `
		`TheApplication().GetSharedGlobal`
		`(varName);`

GotoView Method for an Application	Activates a view.	`TheApplication().GotoView(` `viewName[, BusinessObject]);`

InvokeMethod Method for an Application	Calls a method.	`TheApplication().InvokeMethod(`
		`methodName, methodArg1,`
		`methodArg2,..., methodArgn);`

LoginId Method for an Application	Returns the login ID of the user who started the Siebel application.	`var sID;` `sID = TheApplication().LoginId();`

LoginName Method for an Application	Returns the login name of the user who started Oracle's Siebel application.	`var sUser;`
		`sUser = `
		`TheApplication().LoginName();`

Table 127. Summary of Application Methods for Siebel eScript

Method	Description	Format

NewPropertySet Method for an Application	Creates a new property set.	`var oPropSet;`
		`oPropSet =`
		`TheApplication().NewPropertySet();`

PositionId Method for an Application	Returns the position ID of the user position.	`var sRow;`
		`sRow =`
		`TheApplication().PositionId();`

PositionName Method for an Application	Returns the name of the current user position.	`var sPosition;`
		`sPosition =`
		`TheApplication().PositionName();`

RaiseError Method for an Application	Sends a scripting error message to the browser. To determine the error text, Siebel CRM uses a key to look up the current language.	`var keyVal;`
		`var arg1 ...;`
		`TheApplication().RaiseError(keyVal,`
		`arg1, …);`

RaiseErrorText Method for an Application	Sends a scripting error message to the browser.	`var message;`
		`TheApplication().RaiseErrorText`
		`(message);`

SetPositionId Method for an Application	Sets the active position to a position ID.	`var success;`
		`success =`
		`TheApplication().SetPositionId`
		`(posId);`

SetPositionName Method for an Application	Sets the active position to a position name.	`var success;`
		`success =`
		`TheApplication().SetPositionName`
		`(posName);`

SetProfileAttr Method for an Application	Personalization uses this method to set a value for an attribute in a user profile.	`TheApplication().SetProfileAttr` `(name, value);`
SetSharedGlobal Method for an Application	Sets a shared global variable.	`TheApplication().SetSharedGlobal` `(varName, value);`
Trace Method for an Application	Appends a message to the trace file.	`TheApplication().Trace(message);`
TraceOff Method for an Application	Turns off tracing.	`TheApplication().TraceOff();`
TraceOn Method for an Application	Turns on tracing.	`TheApplication().TraceOn(filename,` `type, selection);`

Application Events for Siebel eScript

Table 128 describes a summary of application events you can use with Siebel eScript.

Table 128. Summary of Application Events for Siebel eScript

Event	Description	Format
Application_Close Event	Called before the Siebel application exits.	Application_Close()
Application_InvokeMethod Event	Called after a specialized method is called.	Application_InvokeMethod (*methodName*)
Application_Navigate Event	Called after the user navigates to a view.	Application_Navigate()
Application_PreInvokeMethod Event	Called before Siebel CRM calls a specialized method.	Application_PreInvokeMethod (*methodName*)
Application_PreNavigate Event	Called before the Siebel application displays the view where the user navigates.	Application_PreNavigate (*DestViewName*, *DestBusObjName*)
Application_Start Event	Called when the Siebel client starts.	Application_Start(*commandLine*)

Business Component Methods for Siebel eScript

Table 129 describes a summary of business component methods you can use with Siebel eScript. It does not include object interface methods that Siebel CRM does not call directly from a business component. For information about methods that Siebel CRM calls with the InvokeMethod method on a business component, see "Business Component Invoke Methods" on page 252.

Table 129. Summary of Business Component Methods for Siebel eScript

Method	Description	Format
ActivateField Method for a Business Component	Activates a field.	var myBusComp; myBusComp.ActivateField(fieldName);
ActivateMultipleFields Method for a Business Component	Activates multiple fields.	var myBusComp; myBusComp.ActivateMultipleFields(oPropSet);

Table 129. Summary of Business Component Methods for Siebel eScript

Method	Description	Format
Associate Method for a Business Component	Creates a new many-to-many relationship for the parent object through an association business component.	`var myBusComp;` `myBusComp.Associate(whereIndicator);`

BusObject Method for a Business Component	Returns the name of the business object that the business component references.	`var myBusComp;`
		`var busObject;`
		`busObject = myBusComp.BusObject();`

ClearToQuery Method for a Business Component	Clears the current query but does not clear sort specifications on a business component.	`var myBusComp;` `myBusComp.ClearToQuery();`
DeactivateFields Method for a Business Component	Deactivates the fields that are currently active from the SQL query statement of a business component.	`var myBusComp;` `myBusComp.DeactivateFields();`
DeleteRecord Method for a Business Component	Removes the current record from a business component.	`var myBusComp;` `myBusComp.DeleteRecord();`
ExecuteQuery Method for a Business Component	Returns a set of business component records.	`var myBusComp;` `myBusComp.ExecuteQuery(cursorMode);`

ExecuteQuery2 Method for a Business Component	Returns a set of business component records. Allows you to control the number of records Siebel CRM returns.	`var myBusComp;`
		`myBusComp.ExecuteQuery2(cursorMode,`
		`ignoreMaxCursorSize);`

FirstRecord Method for a Business Component	Moves the record pointer to the first record in a business component, making that record the current record.	`var myBusComp;`
		`var bIsRecord;`
		`bIsRecord = myBusComp.FirstRecord();`

FirstSelected Method for a Business Component	Makes the first record of the multiple selection in a business component active.	`var myBusComp;`
		`var bIsMultipleSelection;`
		`bIsMultipleSelection =`
		`myBusComp.FirstSelected();`

GetAssocBusComp Method for a Business Component	Returns the name of the association business component.	`var myBusComp;`
		`var AssocBusComp;`
		`AssocBusComp =`
		`myBusComp.GetAssocBusComp();`

Table 129. Summary of Business Component Methods for Siebel eScript

Method	Description	Format

GetFieldValue Method for a Business Component	Returns the value of a field from the current record of a business component.	`var myBusComp;`
		`var sValue;`
		`sValue = myBusComp.GetFieldValue(FieldName);`

GetFormattedFieldValue Method for a Business Component	Returns a field value that is in the same format that the Siebel client uses.	`var myBusComp;`
		`var sValue;`
		`sValue = myBusComp.GetFormattedFieldValue(FieldName);`

GetMultipleFieldValues Method for a Business Component	Returns a value for each field specified in a property set.	`var myBusComp;` `myBusComp.GetMultipleFieldValues (oFields, oValues );`

GetMVGBusComp Method for a Business Component	Returns the multivalue group business component that is associated a business component field.	`var myBusComp;`
		`var MvgBusComp;`
		`MvgBusComp= myBusComp.GetMVGBusComp(FieldName);`

GetNamedSearch Method for a Business Component	Returns the name of a search specification.	`var myBusComp;`
		`var sValue;`
		`sValue = myBusComp.GetNamedSearch(SearchName);`

GetPicklistBusComp Method for a Business Component	Returns the name of the pick business component that is associated with a field in the current business component.	`var myBusComp;`
		`var pickBusComp;`
		`pickBusComp = myBusComp.GetPicklistBusComp (FieldName);`

GetSearchExpr Method for a Business Component	Returns the current search expression that is defined for a business component.	`var myBusComp;`
		`var sExpr;`
		`sExpr = myBusComp.GetSearchExpr();`

GetSearchSpec Method for a Business Component	Returns the search specification that is defined for a business component.	`var myBusComp;`
		`var sSpec;`
		`sSpec = myBusComp.GetSearchSpec(FieldName);`

GetSortSpec Method for a Business Component	Returns the sort specification for a business component.	`var sSortSpec = this.GetSortSpec();`

GetUserProperty Method for a Business Component	Returns the value of a user property.	`var myBusComp;`
		`var sValue;`
		`sValue = myBusComp.GetUserProperty (propertyName);`

Table 129. Summary of Business Component Methods for Siebel eScript

Method	Description	Format

GetViewMode Method for a Business Component	Returns the visibility mode for a business component.	`var myBusComp;`
		`var iMode;`
		`iMode = myBusComp.GetViewMode();`

InvokeMethod Method for a Business Component	Calls a method.	`var myBusComp;`
		`var sReturn;`
		`sReturn = myBusComp.InvokeMethod(`*methodName*`,` *methodArg1*, *methodArg2*,`...,` *methodArgn*`);`

LastRecord Method for a Business Component	Moves the record pointer to the last record in a business component.	`var myBusComp;`
		`var iReturn;`
		`iReturn = myBusComp.LastRecord();`

Name Method for a Business Component	Returns the name of a business component.	`var myBusComp;`
		`var sName;`
		`sName = myBusComp.Name();`

NewRecord Method for a Business Component	Adds a new record to a business component.	`var myBusComp;` `myBusComp.NewRecord(`*whereIndicator*`);`

NextRecord Method for a Business Component	Moves the record pointer to the next record in a business component, making that record the current record.	`var myBusComp;`
		`var bFound;`
		`bFound = myBusComp.NextRecord();`

NextSelected Method for a Business Component	Makes the next record of the current multiple selection the active record.	`var myBusComp;`
		`var iReturn;`
		`iReturn = myBusComp.NextSelected();`

ParentBusComp Method for a Business Component	Returns the name of a parent business component.	`var myBusComp;`
		`var parentBusComp;`
		`parentBusComp = myBusComp.ParentBusComp();`

Pick Method for a Business Component	Places the currently chosen record in a pick business component into the appropriate fields of the parent business component.	`var myBusComp;` `myBusComp.Pick();`

PreviousRecord Method for a Business Component	Moves the record pointer to the previous record in a business component, making that record the current record.	`var myBusComp;`
		`var iReturn;`
		`iReturn = myBusComp.PreviousRecord();`

Table 129. Summary of Business Component Methods for Siebel eScript

Method	Description	Format
RefineQuery Method for a Business Component	Refines a query.	var myBusComp; myBusComp.RefineQuery();
SetFieldValue Method for a Business Component	Sets a new value in a field for the current record of a business component.	var myBusComp; myBusComp.SetFieldValue(*FieldName*, *FieldValue*);
SetFormattedFieldValue Method for a Business Component	Sets a new value in a field in the current record of a business component. It accepts the field value in the current local format.	var myBusComp; myBusComp.SetFormattedFieldValue (*FieldName*, *FieldValue*);
SetMultipleFieldValues Method for a Business Component	Sets new values in the fields of the current record of a business component.	var myBusComp; myBusComp.SetMultipleFieldValues (oPropSet);
SetNamedSearch Method for a Business Component	Sets a named search specification on a business component.	var myBusComp; myBusComp.SetNamedSearch(*searchName*, *searchSpec*);
SetSearchExpr Method for a Business Component	Sets a search expression for a business component.	var myBusComp; myBusComp.SetSearchExpr(*searchSpec*);
SetSearchSpec Method for a Business Component	Sets the search specification for a business component.	var myBusComp; myBusComp.SetSearchSpec(*FieldName*, *searchSpec*);
SetSortSpec Method for a Business Component	Sets the sort specification for a business component.	var myBusComp; myBusComp.SetSortSpec(*sortSpec*);
SetUserProperty Method for a Business Component	Sets the value of a user property in a business component.	var myBusComp; myBusComp.SetUserProperty (*propertyName*, *newValue*);
SetViewMode Method for a Business Component	Sets the visibility type for a business component.	var myBusComp; myBusComp.SetViewMode(viewMode);
UndoRecord Method for a Business Component	Reverses any unsaved modifications made to the record.	var myBusComp; myBusComp.UndoRecord();
WriteRecord Method for a Business Component	Saves to the Siebel database any modifications made to the current record.	var myBusComp; myBusComp.WriteRecord();

Business Component Events for Siebel eScript

Table 130 describes a summary of business component events you can use with Siebel eScript.

Table 130. Summary of Business Component Events for Siebel eScript

Event	Description	Format
BusComp_Associate Event	Called if the user adds a business component record to create an association.	BusComp_Associate()
BusComp_ChangeRecord Event	Called if a business component record becomes the current record.	BusComp_ChangeRecord()
BusComp_CopyRecord Event	Called if the user copies a business component record, and if the user makes this record the active record.	BusComp_CopyRecord()
BusComp_DeleteRecord Event	Called if the user deletes a business component record.	BusComp_DeleteRecord()
BusComp_InvokeMethod Event	Called if Siebel CRM calls the InvokeMethod method on a business component.	BusComp_InvokeMethod(*methodName*)
BusComp_NewRecord Event	Called if the user creates a business component record, and if the user makes this record the active record. You can use this event to set up default values for a field.	BusComp_NewRecord()
BusComp_PreAssociate Event	Called if Siebel CRM detects that the user is about to add a business component record to create an association.	BusComp_PreAssociate()
BusComp_PreCopyRecord Event	Called if Siebel CRM detects that the user is about to copy a business component record. You can use this event to perform precopy validation.	BusComp_PreCopyRecord()

Table 130. Summary of Business Component Events for Siebel eScript

Event	Description	Format
BusComp_PreDeleteRecord Event	Called if Siebel CRM detects that the user is about to delete a business component record. You can use this event to prevent the deletion or to perform any actions before Siebel CRM deletes the record.	BusComp_PreDeleteRecord()
BusComp_PreGetFieldValue Event	Called if a user accesses a business component field.	BusComp_PreGetFieldValue (*FieldName*, *&FieldValue*)
BusComp_PreInvokeMethod Event	Called if Siebel CRM calls a specialized method on a business component. Siebel CRM calls it before it calls this specialized method.	BusComp_PreInvokeMethod (*methodName*)
BusComp_PreNewRecord Event	Called if Siebel CRM detects that the user is about to create a new business component record. You can use this event to perform preinsert validation.	BusComp_PreNewRecord()
BusComp_PreQuery Event	Siebel CRM calls the BusComp_PreQuery event before it runs a query. You can use this event to modify the search criteria or to restrict Siebel CRM from running certain queries.	BusComp_PreQuery()
BusComp_PreSetFieldValue Event	Siebel CRM calls this event after the user modifies a field value or after a call to the SetFieldValue method occurs. This event allows you to use custom validation before Siebel CRM applies predefined validation.	BusComp_PreSetFieldValue (*FieldName*, *FieldValue)*

Table 130. Summary of Business Component Events for Siebel eScript

Event	Description	Format
BusComp_PreWriteRecord Event	Called before Siebel CRM writes a record to the Siebel database.	BusComp_PreWriteRecord()
BusComp_Query Event	Called after Siebel CRM completes a query but before it displays the query results.	BusComp_Query()
BusComp_SetFieldValue Event	Called if Siebel CRM sends a value to a business component from the Siebel client or through a call to the SetFieldValue method.	BusComp_SetFieldValue(*FieldName*)
BusComp_WriteRecord Event	Called after Siebel CRM saves the record to the Siebel database.	BusComp_WriteRecord()

Business Object Methods for Siebel eScript

Table 131 describes a summary of business object methods you can use with Siebel eScript.

Table 131. Summary of Business Object Methods for Siebel eScript

Method	Description	Format
GetBusComp Method for a Business Object	Returns the name of a business component instance.	var myBusObject; var myBusComp; myBusComp = myBusObject.GetBusComp(*BusCompName*);
Name Method for a Business Object	Returns the name of a business object.	var myBusObject as BusObject; var sName; sName = myBusObject.Name();

Business Service Methods for Siebel eScript

Table 132 describes a summary of business service methods you can use with Siebel eScript.

Table 132. Summary of Business Service Methods for Siebel eScript

Method	Description	Format
GetFirstProperty Method for a Business Service	Returns the name of the first property of a business service.	var oService; var sName; sName = oService.GetFirstProperty();
GetNextProperty Method for a Business Service	Returns the name of the next property of a business service.	var oService; var sName; sName = oService.GetNextProperty();
GetProperty Method for a Business Service	Returns the value of a property.	var oService; var sValue; sValue = oService.GetProperty(*propName*);
Name Method for a Business Service	Returns the name of a business service.	var oService; var sName; sName = oService.Name();
InvokeMethod Method for a Business Service	Calls a method.	var oService; oService.InvokeMethod(methodName, InputArguments, OutputArguments);
PropertyExists Method for a Business Service	Returns a Boolean value that indicates if the property that the argument identifies exists.	var oService; var propExists; propExists = oService.PropertyExists(*propName*);
RemoveProperty Method for a Business Service	Removes a property from a business service.	var oService; oService.RemoveProperty(*propName*);
SetProperty Method for a Business Service	Sets a value for a property of a business service.	var oService; oService.SetProperty(*propName*, *propValue*);

Business Service Events for Siebel eScript

Table 133 describes a summary of business service events you can use with Siebel eScript.

Table 133. Summary of Business Service Events for Siebel eScript

Method	Description	Format
Service_InvokeMethod Event	Siebel CRM calls this event after it calls the InvokeMethod method.	Service_InvokeMethod (*methodName*)
Service_PreCanInvokeMethod Event	Siebel CRM calls this event before it calls the PreInvokeMethod event. This configuration allows you to determine if the user possesses the authority to call the business service method.	Service_PreCanInvokeMethod (*MethodName*, &CanInvoke)
Service_PreInvokeMethod Event	Siebel CRM calls this event before it calls a specialized method on a business service.	Service_PreInvokeMethod (*methodName*, Inputs, Outputs)

Property Set Methods for Siebel eScript

Table 134 describes a summary of property set methods you can use with Siebel eScript.

Table 134. Summary of Property Set Methods for Siebel eScript

Method	Description	Format
AddChild Method for a Property Set	Adds a child property set to a property set.	var oPropSet; var iIndex; iIndex = oPropSet.AddChild(childObject);
Copy Method for a Property Set	Returns a copy of a property set.	var oPropSet1; var oPropSet2; oPropSet2 = oPropSet1.Copy();
GetChild Method for a Property Set	Returns the index number of a child property set.	var oPropSet; var sPropVal; sPropVal = oPropSet.GetChild(*index*);
GetChildCount Method for a Property Set	Returns the number of child property sets that exist for a parent property set.	var oPropSet; var iCount; iCount = oPropSet.GetChildCount();

Table 134. Summary of Property Set Methods for Siebel eScript

Method	Description	Format

GetFirstProperty Method for a Property Set	Returns the name of the first property in a property set.	`var oPropSet;`
		`var sPropName;`
		`sPropName =`
		`oPropSet.GetFirstProperty();`

GetNextProperty Method for a Property Set	Returns the name of the next property in a property set.	`var oPropSet;`
		`var sPropName`
		`sPropName =`
		`oPropSet.GetNextProperty();`

GetProperty Method for a Property Set	Returns the value of a property.	`var oPropSet;`
		`var sPropVal`
		`sPropVal =`
		`oPropSet.GetProperty(propName);`

GetPropertyCount Method for a Property Set	Returns the number of properties that exist in the current level in the hierarchy.	`var count;`
		`count =`
		`oPropSet.GetPropertyCount();`

GetType Method for a Property Set	Returns the value of the type attribute of a property set.	`var oPropSet;`
		`var sTypeVal`
		`sTypeVal = oPropSet.GetType(value);`

GetValue Method for a Property Set	Returns the value of the value attribute of a property set.	`var oPropSet;`
		`var sValVal;`
		`sValVal = oPropSet.GetValue(value);`

InsertChildAt Method for a Property Set	Inserts a child property set in a parent property set at a specific location.	`var oPropSet;`
		`oPropSet.InsertChildAt(childObject,`
		`index);`

PropertyExists Method for a Property Set	Returns a Boolean value that indicates if the property that the argument identifies exists.	`Dim oService as SiebelService`
		`Dim propExists as Boolean`
		`propExists =`
		`oService.PropertyExists(propName)`

RemoveChild Method for a Property Set	Removes a child property set from a parent property set.	`var oPropSet;` `oPropSet.RemoveChild(index);`
RemoveProperty Method for a Property Set	Removes a property from a property set.	`var oPropSet;` `oPropSet.RemoveProperty(propName);`
Reset Method for a Property Set	Removes every property and child property set from a property set.	`var oPropSet;` `oPropSet.Reset();`

SetProperty Method for a Property Set	Sets a value in the property of a property set.	`var oPropSet;`
		`oPropSet.SetProperty (propName,`
		`propValue);`

Table 134. Summary of Property Set Methods for Siebel eScript

Method	Description	Format
SetType Method for a Property Set	Sets the value for the type attribute of a property set.	var oPropSet; oPropSet.SetType(*value*);
SetValue Method for a Property Set	Sets the value for the value attribute of a property set.	var oPropSet; oPropSet.SetValue(*value*);

Miscellaneous Methods for Siebel eScript

Table 135 describes a summary of miscellaneous methods you can use with Siebel eScript.

Table 135. Summary of Miscellaneous Methods for Siebel eScript

Method	Description	Format
TheApplication Method	Returns the name of the application object.	TheApplication().*Application_method;*

9 COM Data Server Quick Reference

This chapter describes summary information for COM Data Server. It includes the following topics:

- Application Methods for COM Data Server on page 371
- Business Component Methods for COM Data Server on page 374
- Business Object Methods for COM Data Server on page 378
- Business Service Methods for COM Data Server on page 379
- Property Set Methods for COM Data Server on page 380

Application Methods for COM Data Server

Table 136 describes a summary of application methods you can use with COM Data Server. It does not include object interface methods that Siebel CRM does not call directly from an application object. For information about methods that Siebel CRM calls with the InvokeMethod method on an application object, see "LoadObjects Method for an Application" on page 150.

Table 136. Summary of Application Methods for COM Data Server

Method	Description	Format

CurrencyCode Method for an Application	Returns the currency code that is associated with the division of the user position.	`Dim application as SiebelApplication`
		`Dim sCur as String`
		`sCur = Application.CurrencyCode(ErrCode as Integer)`

GetBusObject Method for an Application	Creates a new instance of a business object.	`Dim application as SiebelApplication`
		`Dim busObject as SiebelBusObject`
		`set busObject =`
		`application.GetBusObject(busobjName as String, ErrCode as Integer)`

GetLastErrText Method for an Application	Returns the text message for the error that Siebel CRM logged most recently.	`Dim application as SiebelApplication`
		`Dim sText as String`
		`sText =`
		`application.GetLastErrText(ErrCode as Integer)`

GetProfileAttr Method for an Application	Returns the name of an attribute in a user profile.	`Dim application as SiebelApplication`
		`Dim sText as String`
		`sText = application.GetProfileAttr(Name as String)`

Table 136. Summary of Application Methods for COM Data Server

Method	Description	Format

GetService Method for an Application	Locates a business service. If this business service is not already running, then Siebel CRM starts it.	`Dim Application as SiebelApplication`
		`Dim Service as SiebelService`
		`set Service =`
		`Application.GetService(serviceName as String, ErrCode as Integer)`

GetSharedGlobal Method for an Application	Returns the shared global variables.	`Dim application as SiebelApplication`
		`Dim sName as String`
		`sName =`
		`application.GetSharedGlobal(varName as String, ErrCode as Integer)`

InvokeMethod Method for an Application	Calls a method.	`Dim application as SiebelApplication` `application.InvokeMethod(methodName as String, methArg1, methArg2, methArgN as String or StringArray)`
LoadObjects Method for an Application	Starts the COM Data Server.	`Dim application as SiebelApplication` `application.LoadObjects(pathName\cfgFileName as String, ErrCode as Integer)`
Login Method for an Application	Allows an external application to log in to the COM Data Server, COM Data Control, or Siebel Java Data Bean, and to access Siebel objects.	`Dim application as SiebelApplication` `application.Login(userName as String, password as String, ErrCode as Integer)`

LoginId Method for an Application	Returns the login ID of the user who started the Siebel application.	`Dim application as SiebelApplication`
		`Dim sID as String`
		`sID = application.LoginId(ErrCode as Integer)`

LoginName Method for an Application	Returns the login name of the user who started the Siebel application.	`Dim application as SiebelApplication`
		`Dim sUser as String`
		`sUser = application.LoginName(ErrCode as Integer)`

NewPropertySet Method for an Application	Creates a new property set.	`Dim oApplication as SiebelApplication`
		`Dim oPropSet as SiebelPropertySet`
		`oPropSet = oApplication.NewPropertySet()`

PositionId Method for an Application	Returns the position ID of the user position.	`Dim application as SiebelApplication`
		`Dim sRow as String`
		`sRow = application.PositionId(ErrCode as Integer)`

PositionName Method for an Application	Returns the name of the current user position.	`Dim application as SiebelApplication`
		`Dim sPosition as String`
		`sPosition =`
		`application.PositionName(ErrCode as Integer)`

Table 136. Summary of Application Methods for COM Data Server

Method	Description	Format

SetPositionId Method for an Application	Sets the active position to a position ID.	`Dim application as SiebelApplication`
		`Dim posId as String`
		`Dim status as Boolean`
		`status = application.SetPositionId(posId as String, ErrCode as Integer)`

SetPositionName Method for an Application	Sets the active position to a position name.	`Dim application as SiebelApplication`
		`Dim posName as String`
		`Dim status as Boolean`
		`status = application.SetPositionName(posName as String, ErrCode as Integer)`

SetProfileAttr Method for an Application	Personalization uses this method to set a value for an attribute in a user profile.	`Dim application as SiebelApplication` `application.SetProfileAttr(name as String, value as String, ErrCode as Integer)`
SetSharedGlobal Method for an Application	Sets a shared global variable.	`Dim application as SiebelApplication` `application.SetSharedGlobal(varName as String, value as String, ErrCode as Integer)`
Trace Method for an Application	Appends a message to the trace file.	`Dim application as SiebelApplication` `application.Trace(message as String, ErrCode as Integer)`
TraceOff Method for an Application	Turns off tracing.	`Dim application as SiebelApplication` `application.TraceOff(ErrCode as Integer)`
TraceOn Method for an Application	Turns on tracing.	`Dim application as SiebelApplication` `application.TraceOn(filename as String, type as Integer, Selection as String, ErrCode as Integer)`

Business Component Methods for COM Data Server

Table 137 describes a summary of the business component methods you can use with the COM Data Server. It does not include object interface methods that Siebel CRM calls with the InvokeMethod method. For information about methods that Siebel CRM calls with the InvokeMethod method on a business component, see "Business Component Invoke Methods" on page 252.

Table 137. Summary of Business Component Methods for COM Data Server

Method	Description	Format
ActivateField Method for a Business Component	Activates a field.	`Dim busComp as SiebelBusComp busComp.ActivateField(fieldName as String, ErrCode as Integer)`
ActivateMultipleFields Method for a Business Component	Activates multiple fields.	`Dim buscomp as SiebelBusComp buscomp.ActivateMultipleFields(oPropSet as SiebelPropertySet, ErrCode as Integer)`
Associate Method for a Business Component	Creates a new many-to-many relationship for the parent object through an association business component.	`Dim busComp as SiebelBusComp busComp.Associate(whereIndicator as Integer, ErrCode as Integer)`
BusObject Method for a Business Component	Returns the name of the business object that the business component references.	`Dim busComp as SiebelBusComp Dim busObject as BusObject Set busObject = busComp.BusObject(ErrCode as Integer)`
ClearToQuery Method for a Business Component	Clears the current query but does not clear sort specifications on a business component.	`Dim busComp as SiebelBusComp busComp.ClearToQuery(ErrCode as Integer)`
DeactivateFields Method for a Business Component	Deactivates the fields that are currently active from the SQL query statement of a business component.	`Dim busComp as SiebelBusComp busComp.DeactivateFields(ErrCode as Integer)`
DeleteRecord Method for a Business Component	Removes the current record from a business component.	`Dim busComp as SiebelBusComp busComp.DeleteRecord(ErrCode as Integer)`
ExecuteQuery Method for a Business Component	Returns a set of business component records.	`Dim busComp as SiebelBusComp busComp.ExecuteQuery(cursorMode as Boolean, ErrCode as Integer)`
ExecuteQuery2 Method for a Business Component	Returns a set of business component records. Allows you to control the number of records Siebel CRM returns.	`Dim busComp as SiebelBusComp busComp.ExecuteQuery2(cursorMode as Boolean, ignoreMaxCursorSize as Boolean, ErrCode as Integer)`

Table 137. Summary of Business Component Methods for COM Data Server

Method	Description	Format

FirstRecord Method for a Business Component	Moves the record pointer to the first record in a business component, making that record the current record.	`Dim busComp as SiebelBusComp`
		`Dim bIsRecord as Boolean`
		`bIsRecord =`
		`busComp.FirstRecord(ErrCode as`
		`Integer)`

FirstSelected Method for a Business Component	Makes the first record of the multiple selection in a business component active.	`Dim busComp as SiebelBusComp`
		`Dim iRecord as Integer`
		`iRecord = busComp.FirstSelected`

GetAssocBusComp Method for a Business Component	Returns the name of the association business component.	`Dim busComp as SiebelBusComp`
		`Dim AssocBusComp as BusComp`
		`Set AssocBusComp =`
		`busComp.GetAssocBusComp(ErrCode as`
		`Integer)`

GetFieldValue Method for a Business Component	Returns the value of a field from the current record of a business component.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetFieldValue(FieldName as`
		`String, ErrCode as Integer)`

GetFormattedFieldValue Method for a Business Component	Returns a field value that is in the same format that the Siebel client uses.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetFormattedFieldValue(FieldName as String, ErrCode as`
		`Integer)`

GetMultipleFieldValues Method for a Business Component	Returns a value for each field specified in a property set.	`Dim buscomp as SiebelBusComp`
		`Dim retValue as Boolean`
		`retValue =`
		`buscomp.GetMultipleFieldValues(oPropSetName as SiebelPropertySet,`
		`oPropSetValue as`
		`SiebelPropertySet, ErrCode as`
		`Integer)`

GetMVGBusComp Method for a Business Component	Returns the multivalue group business component that is associated a business component field.	`Dim busComp as SiebelBusComp`
		`Dim mVGBusComp as SiebelBusComp`
		`set mVGBusComp =`
		`busComp.GetMVGBusComp(FieldName as`
		`String, ErrCode as Integer)`

GetNamedSearch Method for a Business Component	Returns the name of a search specification.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetNamedSearch(SearchName`
		`as String, ErrCode as Integer)`

Table 137. Summary of Business Component Methods for COM Data Server

Method	Description	Format

GetPicklistBusComp Method for a Business Component	Returns the name of the pick business component that is associated with a field in the current business component.	`Dim busComp as SiebelBusComp`
		`Dim pickBusComp as SiebelBusComp`
		`Set pickBusComp =`
		`busComp.GetPicklistBusComp(FieldName as String, ErrCode as Integer)`

GetSearchExpr Method for a Business Component	Returns the current search expression that is defined for a business component.	`Dim busComp as SiebelBusComp`
		`Dim sExpr as String`
		`sExpr =`
		`busComp.GetSearchExpr(ErrCode as Integer)`

GetSearchSpec Method for a Business Component	Returns the search specification that is defined for a business component.	`Dim busComp as BusComp`
		`Dim sSpec as String`
		`sSpec =`
		`busComp.GetSearchSpec(FieldName as String, ErrCode as Integer)`

GetUserProperty Method for a Business Component	Returns the value of a user property.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetUserProperty(propertyName as String, ErrCode as Integer)`

GetViewMode Method for a Business Component	Returns the visibility mode for a business component.	`Dim busComp as SiebelBusComp`
		`Dim iMode as Integer`
		`iMode =`
		`busComp.GetViewMode(ErrCode as Integer)`

LastRecord Method for a Business Component	Moves the record pointer to the last record in a business component.	`Dim busComp as SiebelBusComp`
		`Dim bReturn as Boolean`
		`bReturn =`
		`busComp.LastRecord(ErrCode as Integer)`

Name Method for a Business Component	Returns the name of a business component.	`Dim busComp as SiebelBusComp`
		`Dim sName as String`
		`sName = busComp.Name(ErrCode as Integer)`

NewRecord Method for a Business Component	Adds a new record to a business component.	`Dim busComp as SiebelBusComp` `busComp.NewRecord(whereIndicator as Integer, ErrCode as Integer)`

NextRecord Method for a Business Component	Moves the record pointer to the next record in a business component, making that record the current record.	`Dim busComp as SiebelBusComp`
		`Dim bReturn as Boolean`
		`bReturn =`
		`busComp.NextRecord(ErrCode as Integer)`

Table 137. Summary of Business Component Methods for COM Data Server

Method	Description	Format
ParentBusComp Method for a Business Component	Returns the name of a parent business component.	```Dim busComp as SiebelBusComp Dim parentBusComp as SiebelBusComp Set parentBusComp = busComp.ParentBusComp(ErrCode as Integer)```
Pick Method for a Business Component	Places the currently chosen record in a pick business component into the appropriate fields of the parent business component.	```Dim busComp as SiebelBusComp busComp.Pick(ErrCode as Integer)```
PreviousRecord Method for a Business Component	Moves the record pointer to the previous record in a business component, making that record the current record.	```Dim busComp as SiebelBusComp Dim bReturn as Boolean bReturn = busComp.PreviousRecord(ErrCode as Integer)```
RefineQuery Method for a Business Component	Refines a query.	```Dim busComp as SiebelBusComp busComp.RefineQuery(ErrCode as Integer)```
SetFieldValue Method for a Business Component	Sets a new value in a field for the current record of a business component.	```Dim busComp as SiebelBusComp SetFieldValue(fieldname As String, fieldValue As string, errCode as Integer)```
SetFormattedFieldValue Method for a Business Component	Sets a new value in a field in the current record of a business component. It accepts the field value in the current local format.	```Dim busComp as SiebelBusComp busComp.SetFormattedFieldValue(FieldName as String, FieldValue as String, ErrCode as Integer)```
SetMultipleFieldValues Method for a Business Component	Sets new values in the fields of the current record of a business component.	```Dim buscomp as SiebelBusComp buscomp.SetMultipleFieldValues(oPropSet as SiebelPropertySet, ErrCode as Integer)```
SetNamedSearch Method for a Business Component	Sets a named search specification on a business component.	```Dim busComp as SiebelBusComp busComp.SetNamedSearch(searchName as String, searchSpec as String, ErrCode as Integer)```
SetSearchExpr Method for a Business Component	Sets a search expression for a business component.	```Dim busComp as SiebelBusComp busComp.SetSearchExpr(searchSpec as String, ErrCode as Integer)```
SetSearchSpec Method for a Business Component	Sets the search specification for a business component.	```Dim busComp as SiebelBusComp busComp.SetSearchSpec(FieldName as String, searchSpec as String, ErrCode as Integer)```

Table 137. Summary of Business Component Methods for COM Data Server

Method	Description	Format
SetSortSpec Method for a Business Component	Sets the sort specification for a business component.	`Dim busComp as SiebelBusComp busComp.SetSortSpec(`*`sortSpec`*` as String, ErrCode as Integer)`
SetUserProperty Method for a Business Component	Sets the value of a user property in a business component.	`Dim busComp as SiebelBusComp busComp.SetUserProperty(`*`propertyName`*` as String, `*`newValue`*` as String, ErrCode as Integer)`
SetViewMode Method for a Business Component	Sets the visibility type for a business component.	`Dim buscomp as SiebelBusComp buscomp.SetViewMode(mode As Integer, errCode As Integer)`
UndoRecord Method for a Business Component	Reverses any unsaved modifications made to the record.	`Dim busComp as SiebelBusComp busComp.UndoRecord(ErrCode as Integer)`
WriteRecord Method for a Business Component	Saves to the Siebel database any modifications made to the current record.	`Dim busComp as SiebelBusComp busComp.WriteRecord(ErrCode as Integer)`

Business Object Methods for COM Data Server

Table 138 describes a summary of business object methods you can use with the COM Data Server.

Table 138. Summary of Business Object Methods for COM Data Server

Method	Description	Format
GetBusComp Method for a Business Object	Returns the name of a business component.	`Dim busObject as SiebelBusObject Dim busComp as SiebelBusComp set busComp = busObject.GetBusComp(`*`BusCompName`*` as String, ErrCode as Integer)`
Name Method for a Business Object	Returns the name of a control.	`Dim busObject as SiebelBusObject Dim sName as String sName = busObject.Name(ErrCode as Integer)`

Business Service Methods for COM Data Server

Table 139 describes a summary of business service methods you can use with the COM Data Server.

Table 139. Summary of Business Service Methods for COM Data Server

Method	Description	Format

GetFirstProperty Method for a Business Service	Returns the name of the first property of a business service.	`Dim oService as SiebelService`
		`Dim sName as String`
		`sName =`
		`oService.GetFirstProperty(ErrCode as Integer)`

GetNextProperty Method for a Business Service	Returns the name of the next property of a business service.	`Dim oService as SiebelService`
		`Dim sName as String`
		`sName = oService.GetNextProperty(ErrCode as Integer)`

GetProperty Method for a Business Service	Returns the value of a property.	`Dim oService as SiebelService`
		`Dim sValue as String`
		`sValue = oService.GetProperty(propName as String, ErrCode as Integer)`

Name Method for a Business Service	Returns the name of a business service.	`Dim oService as SiebelService`
		`Dim sName as String`
		`sName = oService.Name`

InvokeMethod Method for a Business Service	Calls a method.	`Dim oService as SiebelService` `oService.InvokeMethod(methodName as String, InputArguments as SiebelPropertySet, OutputArguments as SiebelPropertySet, ErrCode as Integer)`

PropertyExists Method for a Business Service	Returns a Boolean value that indicates if the property that the argument identifies exists.	`Dim oService as SiebelService`
		`Dim propExists as Boolean`
		`propExists = oService.PropertyExists(propName as String)`

RemoveProperty Method for a Business Service	Removes a property from a business service.	`Dim oService as SiebelService` `oService.RemoveProperty(propName as String, ErrCode as Integer)`
SetProperty Method for a Business Service	Sets a value for a property of a business service.	`Dim oService as SiebelService` `oService.SetProperty(propName as String, propValue as String, ErrCode as Integer)`

Property Set Methods for COM Data Server

Table 140 describes a summary of property set methods you can use with the COM Data Server.

Table 140. Summary of Property Set Methods for COM Data Server

Method	Description	Format

AddChild Method for a Property Set	Adds child property sets to a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim iIndex as Integer`
		`iIndex = oPropSet.AddChild( childObject as PropertySet, errCode as Integer)`

Copy Method for a Property Set	Returns a copy of a property set.	`Dim oPropSet1 as SiebelPropertySet`
		`Dim oPropSet2 as SiebelPropertySet`
		`oPropSet2 = oPropSet1.Copy(ErrCode as Integer)`

GetChild Method for a Property Set	Returns a child property set of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim oChildPropSet as SiebelPropertySet`
		`oChildPropSet = oPropSet.GetChild( index as Integer, ErrCode as Integer)`

GetChildCount Method for a Property Set	Returns the number of child property sets that exist for a parent property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim iCount as Integer`
		`iCount = oPropSet.GetChildCount(ErrCode as Integer)`

GetFirstProperty Method for a Property Set	Returns the name of the first property in a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropName as String`
		`sPropName = oPropSet.GetFirstProperty(ErrCode as Integer)`

GetNextProperty Method for a Property Set	Returns the name of the next property in a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropName as String`
		`sPropName = oPropSet.GetNextProperty(ErrCode as Integer)`

GetProperty Method for a Property Set	Returns the value of a property.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropVal as String`
		`sPropVal = oPropSet.GetProperty(propName as String, ErrCode as Integer)`

GetPropertyCount Method for a Property Set	Returns the number of properties that exist in the current level in the hierarchy.	`Dim oPropSet as SiebelPropertySet`
		`Dim propCount as Integer`
		`propCount = oPropSet.GetPropertyCount (ErrCode as Integer)`

GetType Method for a Property Set	Returns the value of the type attribute of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sTypeVal as String`
		`sTypeVal = oPropSet.GetType(value as String)`

Table 140. Summary of Property Set Methods for COM Data Server

Method	Description	Format

GetValue Method for a Property Set	Returns the value of the value attribute of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sValVal as String`
		`sValVal = oPropSet.GetValue(ErrCode as Integer)`

InsertChildAt Method for a Property Set	Inserts a child property set in a parent property set at a specific location.	`Dim oPropSet as SiebelPropertySet` `oPropSet.InsertChildAt(childObject as String, index as Integer, ErrCode as Integer)`

PropertyExists Method for a Property Set	Returns a Boolean value that indicates if the property that the argument identifies exists.	`Dim oPropSet as SiebelPropertySet`
		`Dim propExists as Boolean`
		`propExists = oPropSet.PropertyExists(propName as String, ErrCode as Integer)`

RemoveChild Method for a Property Set	Removes a child property set from a parent property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.RemoveChild(index as Integer, errCode as Integer)`
RemoveProperty Method for a Property Set	Removes a property from a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.RemoveProperty(propName as String, ErrCode as Integer)`
Reset Method for a Property Set	Removes every property and child property set from a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.Reset(ErrCode as Integer)`
SetProperty Method for a Property Set	Sets a value in the property of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetProperty(propName as String, propValue as String, ErrCode as Integer)`
SetType Method for a Property Set	Sets the value for the type attribute of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetType(value as String, ErrCode as Integer)`
SetValue Method for a Property Set	Sets the value for the value attribute of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetValue(value as String, errCode as Integer)`

10 COM Data Control Quick Reference

This chapter describes summary information for COM Data Control. It includes the following topics:

- Application Methods for COM Data Control
- Business Component Methods for COM Data Control on page 386
- Business Object Methods for COM Data Control on page 390
- Business Service Methods for COM Data Control on page 391
- Property Set Methods for COM Data Control on page 392

Application Methods for COM Data Control

Table 141 describes a summary of application methods you can use with COM Data Control. It does not include object interface methods that Siebel CRM does not call directly from an application instance. For information about methods that Siebel CRM calls with the InvokeMethod method on the application object, see "LoadObjects Method for an Application" on page 150.

Table 141. Summary of Application Methods for COM Data Control

Method	Description	Format

Attach Method for an Application	Allows an external application to reconnect to an existing Siebel session.	`Dim application as SiebelDataControl`
		`Dim status as Boolean`
		`status = application.Attach(sessionID As String)`

CurrencyCode Method for an Application	Returns the currency code that is associated with the division of the user position.	`Dim application as SiebelDataControl`
		`Dim sCur as String`
		`sCur = Application.CurrencyCode`

Detach Method for an Application	Returns a string that contains the Siebel session ID.	`Dim application as SiebelDataControl`
		`Dim sessionId as String`
		`sessionId = application.Detach()`

EnableExceptions Method for an Application	Enables or disables native Component Object Model (COM) error handling.	`Dim application as SiebelDataControl`
		`Dim bEnable as Boolean`
		`bEnable = true`
		`application.EnableExceptions(bEnable)`

GetBusObject Method for an Application	Creates a new instance of a business object.	`Dim application as SiebelDataControl`
		`Dim busObject as SiebelBusObject`
		`set busObject = application.GetBusObject(busobjName as String)`

Table 141. Summary of Application Methods for COM Data Control

Method	Description	Format

GetLastErrCode Method for an Application	Returns the error code for the error that Siebel CRM logged most recently.	`Dim application as SiebelDataControl`
		`Dim iErr as Integer`
		`iErr = application.GetLastErrCode`

GetLastErrText Method for an Application	Returns the text message for the error that Siebel CRM logged most recently.	`Dim application as SiebelDataControl`
		`Dim sText as String`
		`sText = application.GetLastErrText`

GetProfileAttr Method for an Application	Returns the name of an attribute in a user profile.	`Dim application as SiebelDataControl`
		`Dim sText as String`
		`sText =`
		`application.GetProfileAttr(profileAttr`
		`ibuteName as string)`

GetService Method for an Application	Locates a business service. If this business service is not already running, then Siebel CRM starts it.	`Dim application as SiebelDataControl`
		`Dim service as SiebelService`
		`set service = application.GetService(`
		`serviceName as String)`

GetSharedGlobal Method for an Application	Returns the shared global variables.	`Dim application as SiebelDataControl`
		`Dim sText as string`
		`sText =`
		`application.GetSharedGlobal (globalVari`
		`ableName as string)`

InvokeMethod Method for an Application	Calls a method.	`Dim application as SiebelDataControl`
		`Dim sReturn as String`
		`sReturn =`
		`application.InvokeMethod(`*`methodName`* `as`
		`String,` *`methArg1, methArg2, methArgN`* `as`
		`String or StringArray)`

Login Method for an Application	Allows an external application to log in to the COM Data Server, COM Data Control, or Siebel Java Data Bean, and to access Siebel objects.	`Dim application as SiebelDataControl`
		`Dim sErr as String`
		`sErr = application.Login(`*`connectString`*
		`as String,` *`userName`* `as String,` *`password`*
		`as String)`

LoginId Method for an Application	Returns the login ID of the user who started the Siebel application.	`Dim application as SiebelDataControl`
		`Dim sID as String`
		`sID = application.LoginId`

LoginName Method for an Application	Returns the login name of the user who started the Siebel application.	`Dim application as SiebelDataControl`
		`Dim sUser as String`
		`sUser = application.LoginName`

Logoff Method for an Application	Disconnects the Siebel client from the Siebel Server.	`Dim SiebApp as SiebelDataControl`
		`Dim boolVal as Boolean`
		`boolVal = siebApp.LogOff`

Table 141. Summary of Application Methods for COM Data Control

Method	Description	Format

NewPropertySet Method for an Application	Creates a new property set.	`Dim application as SiebelDataControl`
		`Dim PropSet as SiebelPropertySet`
		`PropSet = oApplication.NewPropertySet`

PositionId Method for an Application	Returns the position ID of the user position.	`Dim application as SiebelDataControl`
		`Dim sRow as String`
		`sRow = application.PositionId`

PositionName Method for an Application	Returns the name of the current user position.	`Dim application as SiebelDataControl`
		`Dim sPosition as String`
		`sPosition = application.PositionName`

SetPositionId Method for an Application	Sets the active position to a Position ID.	`Dim application as SiebelDataControl`
		`Dim status as Boolean`
		`status =`
		`application.SetPositionId(sPosId)`

SetPositionName Method for an Application	Sets the active position to a position name.	`Dim application as SiebelDataControl`
		`Dim status as Boolean`
		`status =`
		`application.SetPositionName(sPosName)`

SetProfileAttr Method for an Application	Personalization uses this method to set a value for an attribute in a user profile.	`Dim application as SiebelDataControl` `application.SetProfileAttr(name as String, value as String)`

SetSharedGlobal Method for an Application	Sets a shared global variable.	`Dim SiebApp as SiebelDataControl`
		`Dim boolVal as Boolean`
		`boolVal =`
		`SiebApp.SetSharedGlobal(varName As String, value As String)`

Trace Method for an Application	Appends a message to the trace file.	`Dim SiebApp as SiebelDataControl`
		`Dim boolVal as Boolean`
		`boolVal = siebApp.TraceOn(msg As String)`

TraceOff Method for an Application	Turns off tracing.	`Dim SiebApp as SiebelDataControl`
		`Dim boolVal as Boolean`
		`boolVal =siebApp.TraceOff`

TraceOn Method for an Application	Turns on tracing.	`Dim SiebApp as SiebelDataControl`
		`Dim boolVal as Boolean`
		`boolVal = siebApp.TraceOn(fileName As String, category As String, src As String)`

Business Component Methods for COM Data Control

Table 142 describes a summary of business component methods you can use with COM Data Control. It does not include object interface methods that Siebel CRM does not call directly from a business component instance. For information about methods that Siebel CRM calls with the InvokeMethod method on a business component, see "Business Component Invoke Methods" on page 252.

Table 142. Summary of Business Component Methods for COM Data Control

Method	Description	Format
ActivateField Method for a Business Component	Activates a field.	`Dim busComp as SiebelBusComp` `BusComp.ActivateField(fieldName as String)`
ActivateMultipleFields Method for a Business Component	Activates multiple fields.	`Dim busComp as SiebelBusComp` `busComp.ActivateMultipleFields(oPropSet as SiebelPropertySet)`
Associate Method for a Business Component	Creates a new many-to-many relationship for the parent object through an association business component.	`Dim busComp as SiebelBusComp` `busComp.Associate(whereIndicator as Integer)`

BusObject Method for a Business Component	Returns the name of the business object that the business component references.	`Dim busComp as SiebelBusComp`
		`Dim busObject as SiebelBusObject`
		`Set busObject = busComp.BusObject`

ClearToQuery Method for a Business Component	Clears the current query but does not clear sort specifications on a business component.	`Dim busComp as SiebelBusComp` `busComp.ClearToQuery`
DeactivateFields Method for a Business Component	Deactivates the fields that are currently active from the SQL query statement of a business component.	`Dim busComp as SiebelBusComp` `busComp.DeactivateFields`
DeleteRecord Method for a Business Component	Removes the current record from a business component.	`Dim busComp as SiebelBusComp` `busComp.DeleteRecord`
ExecuteQuery Method for a Business Component	Returns a set of business component records.	`Dim buscomp as SiebelBusComp` `buscomp.ExecuteQuery(cursorMode As Integer) As Boolean`
ExecuteQuery2 Method for a Business Component	Returns a set of business component records. Allows you to control the number of records Siebel CRM returns.	`Dim buscomp as SiebelBusComp` `buscomp.ExecuteQuery2(cursorMode As Integer, ignoreMaxCursorSize As Integer) As Boolean`

Table 142. Summary of Business Component Methods for COM Data Control

Method	Description	Format

FirstRecord Method for a Business Component	Moves the record pointer to the first record in a business component, making that record the current record.	`Dim busComp as SiebelBusComp`
		`Dim bIsRecord as Boolean`
		`bIsRecord = busComp.FirstRecord`

GetFieldValue Method for a Business Component	Returns the value of a field from the current record of a business component.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetFieldValue(FieldName`
		`as String)`

GetFormattedFieldValue Method for a Business Component	Returns a field value that is in the same format that the Siebel client uses.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetFormattedFieldValue`
		`(FieldName as String)`

GetLastErrCode Method for a Business Component	Returns the error code for the error that Siebel CRM logged most recently.	`Dim errCode As Integer`
		`Dim SiebApp as SiebelDataControl`
		`errCode=siebApp.GetLastErrCode`

GetLastErrText Method for an Application	Returns the text message for the error that Siebel CRM logged most recently.	`Dim busComp as SiebelBusComp`
		`Dim sErr as String`
		`sErr = busComp.GetLastErrText`

GetMultipleFieldValues Method for a Business Component	Returns a value for each field specified in a property set.	`Dim busComp as SiebelBusComp`
		`busComp.GetMultipleFieldValues`
		`(oFieldNames as`
		`SiebelPropertySet, oFieldValues`
		`as SiebelPropertySet)`

GetMVGBusComp Method for a Business Component	Returns the multivalue group business component that is associated a business component field.	`Dim busComp as SiebelBusComp`
		`Dim mVGBusComp as SiebelBusComp`
		`set mVGBusComp =`
		`busComp.GetMVGBusComp(FieldName`
		`as String)`

GetNamedSearch Method for a Business Component	Returns the name of a search specification.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetNamedSearch(SearchName`
		`as String)`

GetPicklistBusComp Method for a Business Component	Returns the name of the pick business component that is associated with a field in the current business component.	`Dim busComp as SiebelBusComp`
		`Dim pickBusComp as SiebelBusComp`
		`Set pickBusComp =`
		`busComp.GetPicklistBusComp(Field-`
		`Name as String)`

GetSearchExpr Method for a Business Component	Returns the current search expression that is defined for a business component.	`Dim busComp as SiebelBusComp`
		`Dim sExpr as String`
		`sExpr = busComp.GetSearchExpr`

Table 142. Summary of Business Component Methods for COM Data Control

Method	Description	Format
GetSearchSpec Method for a Business Component	Returns the search specification that is defined for a business component.	```Dim busComp as SiebelBusComp
Dim sSpec as String
sSpec = busComp.GetSearchSpec(FieldName as String)```
GetUserProperty Method for a Business Component	Returns the value of a user property.	```Dim buscomp as SiebelBusComp
Dim retStr as String
retStr = buscomp.GetUserProperty(prop As String) As String```
GetViewMode Method for a Business Component	Returns the visibility mode for a business component.	```Dim busComp as SiebelBusComp
Dim iMode as Integer
iMode = busComp.GetViewMode```
InvokeMethod Method for a Business Component	Calls a method.	```Dim busComp as SiebelBusComp
Dim sReturn as String
sReturn = busComp.InvokeMethod(methodName as String, methArg1, methArg2, methArgN as String or StringArray)```
LastRecord Method for a Business Component	Moves the record pointer to the last record in a business component.	```Dim busComp as SiebelBusComp
Dim bReturn as Boolean
bReturn = busComp.LastRecord```
Name Method for a Business Component	Returns the name of a business component.	```Dim busComp as SiebelBusComp
Dim sName as String
sName = busComp.Name```
NewRecord Method for a Business Component	Adds a new record to a business component.	```Dim busComp as SiebelBusComp
busComp.NewRecord(whereIndicator as Integer)```
NextRecord Method for a Business Component	Moves the record pointer to the next record in a business component, making that record the current record.	```Dim busComp as SiebelBusComp
Dim bReturn as Boolean
bReturn = busComp.NextRecord```
ParentBusComp Method for a Business Component	Returns the name of a parent business component.	```Dim busComp as SiebelBusComp
Dim parentBusComp as SiebelBusComp
Set parentBusComp = busComp.ParentBusComp```
Pick Method for a Business Component	Places the currently chosen record in a pick business component into the appropriate fields of the parent business component.	```Dim busComp as SiebelBusComp
busComp.Pick```

Table 142. Summary of Business Component Methods for COM Data Control

Method	Description	Format

PreviousRecord Method for a Business Component	Moves the record pointer to the previous record in a business component, making that record the current record.	`Dim busComp as SiebelBusComp`
		`Dim bReturn as Boolean`
		`bReturn = busComp.PreviousRecord`

RefineQuery Method for a Business Component	Refines a query.	`Dim busComp as SiebelBusComp` `busComp.RefineQuery`
SetFieldValue Method for a Business Component	Sets a new value in a field in the current record of a business component.	`Dim busComp as SiebelBusComp` `busComp.SetFieldValue(`*FieldName*` as String, `*FieldValue*` as String)`
SetFormattedFieldValue Method for a Business Component	Sets a new value in a field in the current record of a business component. It accepts the field value in the current local format.	`Dim busComp as SiebelBusComp` `busComp.SetFormattedFieldValue(`*FieldName*` as String, `*FieldValue*` as String)`
SetMultipleFieldValues Method for a Business Component	Sets new values in the fields of the current record of a business component.	`Dim busComp as SiebelBusComp` `busComp.SetMultipleFieldValues (oPropSet as SiebelPropertySet)`
SetNamedSearch Method for a Business Component	Sets a named search specification on a business component.	`Dim busComp as SiebelBusComp` `busComp.SetNameSearch(`*searchName*` as String, `*searchSpec*` as String)`
SetSearchExpr Method for a Business Component	Sets a search expression for a business component.	`Dim busComp as SiebelBusComp` `busComp.SetSearchExpr(`*searchSpec*` as String)`
SetSearchSpec Method for a Business Component	Sets the search specification for a business component.	`Dim busComp as SiebelBusComp` `busComp.SetSearchSpec(`*FieldName*` as String, searchSpec as String)`
SetSortSpec Method for a Business Component	Sets the sort specification for a business component.	`Dim busComp as SiebelBusComp` `busComp.SetSortSpec(`*sortSpec*` as String)`
SetUserProperty Method for a Business Component	Sets the value of a named user property.	`Dim buscomp as SiebelBusComp` `buscomp.SetUserProperty(`*property-Name*` as String, `*newValue*` as String)`

SetViewMode Method for a Business Component	Sets the visibility type for a business component.	`Dim buscomp as SiebelBusComp`
		`Dim boolVal as Boolean`
		`boolVal = buscomp.SetViewMode(mode As Integer)`

Table 142. Summary of Business Component Methods for COM Data Control

Method	Description	Format
UndoRecord Method for a Business Component	Reverses any unsaved modifications made to the record.	`Dim busComp as SiebelBusComp` `busComp.UndoRecord`
WriteRecord Method for a Business Component	Saves to the Siebel database any modifications made to the current record.	`Dim busComp as SiebelBusComp` `busComp.WriteRecord`

Business Object Methods for COM Data Control

Table 143 describes a summary of business object methods you can use with COM Data Control.

Table 143. Summary of Business Object Methods for COM Data Control

Method	Description	Format

GetBusComp Method for a Business Object	Returns the name of a business component.	`Dim busObject as SiebelBusObject`
		`Dim busComp as SiebelBusComp`
		`set busComp =`
		`BusObject.GetBusComp(BusCompName as String)`

GetLastErrCode Method for a Business Object	Returns the error code for the error that Siebel CRM logged most recently.	`Dim busObject as SiebelBusObject`
		`Dim iErr as Integer`
		`iErr = busObject.GetLastErrCode`

GetLastErrText Method for a Business Object	Returns the text message for the error that Siebel CRM logged most recently.	`Dim busObject as SiebelBusObject`
		`Dim sErr as String`
		`sErr = busObject.GetLastErrText`

Name Method for a Business Object	Returns the name of a control.	`Dim busObject as SiebelBusObject`
		`Dim sName as String`
		`sName = busObject.Name`

Business Service Methods for COM Data Control

Table 144 describes a summary of business service methods you can use with COM Data Control.

Table 144. Summary of Business Service Methods for COM Data Control

Method	Description	Format
GetFirstProperty Method for a Business Service	Returns the name of the first property of a business service.	```Dim oService as SiebelService Dim sName as String sName = oService.GetFirstProperty()```
GetNextProperty Method for a Business Service	Returns the name of the next property of a business service.	```Dim oService as SiebelService Dim sName as String sName = oService.GetNextProperty()```
GetProperty Method for a Business Service	Returns the value of a property.	```Dim oService as SiebelService Dim sValue as String sValue = oService.GetProperty(propName as String)```
Name Method for a Business Service	Returns the name of a business service.	```Dim oService as SiebelService Dim sName as String sName = oService.Name```
InvokeMethod Method for a Business Service	Calls a method.	```Dim oService as SiebelService oService.InvokeMethod(methodName as String, InputArguments as SiebelPropertySet, OutputArguments as SiebelPropertySet)```
PropertyExists Method for a Business Service	Returns a Boolean value that indicates if the property that the argument identifies exists.	```Dim oService as SiebelService Dim propExists as Boolean propExists = oService.PropertyExists(propName as String)```
RemoveProperty Method for a Business Service	Removes a property from a business service.	```Dim oService as SiebelService oService.RemoveProperty(propName as String)```
SetProperty Method for a Business Service	Sets a value for a property of a business service.	```Dim oService as SiebelService oService.SetProperty(propName as String, propValue as String)```

Property Set Methods for COM Data Control

Table 145 describes a summary of property set methods you can use with COM Data Control.

Table 145. Summary of Property Set Methods for COM Data Control

Method	Description	Format

AddChild Method for a Property Set	Adds child property sets to a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim iIndex as Integer`
		`iIndex = oPropSet.AddChild(`
		`childObject as Property Set)`

Copy Method for a Property Set	Returns a copy of a property set.	`Dim oPropSet1 as SiebelPropertySet`
		`Dim oPropSet2 as SiebelPropertySet`
		`oPropSet2 = oPropSet1.Copy()`

GetChild Method for a Property Set	Returns a child property set of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim oPropSet1 as SiebelPropertySet`
		`oPropSet1 = oPropSet.GetChild(index`
		`as Integer)`

GetChildCount Method for a Property Set	Returns the number of child property sets that exist for a parent property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim iCount as Integer`
		`iCount = oPropSet.GetChildCount()`

GetFirstProperty Method for a Property Set	Returns the name of the first property in a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropName as String`
		`sPropName =`
		`oPropSet.GetFirstProperty()`

GetNextProperty Method for a Property Set	Returns the name of the next property in a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropName as String`
		`sPropName =`
		`oPropSet.GetNextProperty()`

GetProperty Method for a Property Set	Returns the value of a property.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropVal as String`
		`sPropVal =`
		`oPropSet.GetProperty(propName as`
		`String)`

GetPropertyCount Method for a Property Set	Returns the number of properties that exist in the current level in the hierarchy.	`Dim oPropSet as SiebelPropertySet`
		`Dim count as Long`
		`count = oPropSet.GetPropertyCount`

GetType Method for a Property Set	Returns the value of the type attribute of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sTypeVal as String`
		`sTypeVal = oPropSet.GetType()`

GetValue Method for a Property Set	Returns the value of the value attribute of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sValVal as String`
		`sValVal = oPropSet.GetValue()`

Table 145. Summary of Property Set Methods for COM Data Control

Method	Description	Format
InsertChildAt Method for a Property Set	Inserts a child property set in a parent property set at a specific location.	`Dim oPropSet as SiebelPropertySet` `oPropSet.InsertChildAt(childObject as SiebelPropertySet, index as Long)`

PropertyExists Method for a Property Set	Returns a Boolean value that indicates if the property that the argument identifies exists.	`Dim oPropSet as Property Set`
		`Dim propExists as Boolean`
		`propExists = oPropSet.PropertyExists(propName as String)`

RemoveChild Method for a Property Set	Removes a child property set from a parent property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.RemoveChild(index as Long)`
RemoveProperty Method for a Property Set	Removes a property from a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.RemoveProperty(propName as String)`
Reset Method for a Property Set	Removes every property and child property set from a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.Reset()`
SetProperty Method for a Property Set	Sets a value in the property of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetProperty(propName as String, propValue as String)`
SetType Method for a Property Set	Sets the value for the type attribute of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetType(value as String)`
SetValue Method for a Property Set	Sets the value for the value attribute of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetValue(value as String)`

11 Web Client Automation Server Quick Reference

This chapter describes summary information for the Web Client Automation Server. It includes the following topics:

- Siebel HTML Application Methods for the Web Client Automation Server on page 395
- Siebel Service Methods for the Web Client Automation Server on page 396
- Property Set Methods for the Web Client Automation Server on page 397

Siebel HTML Application Methods for the Web Client Automation Server

Table 146 describes a summary of Siebel HTML application methods you can use with the Web Client Automation Server. It does not include object interface methods that Siebel CRM does not call directly from an application instance. For information about methods that Siebel CRM calls with the InvokeMethod method on the application, see "LoadObjects Method for an Application" on page 150.

Table 146. Summary of Siebel HTML Application Methods for the Web Client Automation Server

Method	Description	Format

GetLastErrCode Method for an Application	Returns the error code for the error that Siebel CRM logged most recently.	`Dim siebelApp As SiebelHTMLApplication`
		`Dim iErr as Long`
		`iErr = siebelApp.GetLastErrCode`

GetLastErrText Method for an Application	Returns the text message for the error that Siebel CRM logged most recently.	`Dim siebelApp As SiebelHTMLApplication`
		`Dim sText as String`
		`sText = siebelApp.GetLastErrText`

GetService Method for an Application	Locates a business service. If this business service is not already running, then Siebel CRM starts it.	`Dim siebelApp As SiebelHTMLApplication`
		`Dim svc As SiebelService`
		`Set svc =`
		`siebelApp.GetService(ServiceName`
		`as String)`

Table 146. Summary of Siebel HTML Application Methods for the Web Client Automation Server

Method	Description	Format

Name Method for an Application	Returns the name of the Siebel application.	`Dim siebelApp As SiebelHTMLApplication`
		`Dim name as String`
		`name = siebelApp.Name`

NewPropertySet Method for an Application	Creates a new property set.	`Dim siebelApp As SiebelHTMLApplication`
		`Dim propSet as SiebelPropertySet`
		`Set propSet = siebelApp.NewPropertySet`

Siebel Service Methods for the Web Client Automation Server

Table 147 describes a summary of Siebel service methods you can use with the Web Client Automation Server.

Table 147. Summary of Siebel Service Methods for the Web Client Automation Server

Method	Description	Format

GetNextProperty Method for a Business Service	Returns the name of the next property of a business service.	`Dim svc As SiebelService`
		`Dim iErr as Long`
		`iErr = svc.GetLastErrCode`

InvokeMethod Method for a Business Service	Calls a method.	`Dim svc As SiebelService` `svc.InvokeMethod(MethodName as String, inputPropSet as SiebelPropertySet, outputPropSet as SiebelPropertySet)`

Name Method for a Business Service	Returns the name of a business service.	`Dim svc As SiebelService`
		`Dim name as String`
		`name = svc.Name`

Property Set Methods for the Web Client Automation Server

Table 148 describes a summary of the property set methods you can use with the Web Client Automation Server.

Table 148. Summary of Property Set Methods for the Web Client Automation Server

Method	Description	Format
AddChild Method for a Property Set	Adds child property sets to a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.AddChild(childObject as SiebelPropertySet)`

Copy Method for a Property Set	Returns a copy of a property set.	`Dim oPropSet1 as SiebelPropertySet`
		`Dim oPropSet2 as SiebelPropertySet`
		`Set oPropSet2 = oPropSet1.Copy`

GetChild Method for a Property Set	Returns a child property set of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim oChildPropSet as SiebelPropertySet`
		`Set oChildPropSet = oPropSet.GetChild(index as Long)`

GetChildCount Method for a Property Set	Returns the number of child property sets that exist for a parent property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim iCount as Long`
		`iCount = oPropSet.GetChildCount`

GetFirstProperty Method for a Property Set	Returns the name of the first property in a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropName as String`
		`sPropName = oPropSet.GetFirstProperty`

GetLastErrCode Method for a Property Set	Returns the error code for the error that Siebel CRM logged most recently.	`Dim oPropSet as SiebelPropertySet`
		`Dim iErr as Long`
		`iErr = oPropSet.GetLastErrCode`

GetLastErrText Method for a Property Set	Returns the text message for the error that Siebel CRM logged most recently.	`Dim oPropSet as SiebelPropertySet`
		`Dim sText as String`
		`sText = oPropSet.GetLastErrText`

GetNextProperty Method for a Property Set	Returns the name of the next property in a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropName as String`
		`sPropName = oPropSet.GetNextProperty`

GetProperty Method for a Property Set	Returns the value of a property.	`Dim oPropSet as SiebelPropertySet`
		`Dim sValue as String`
		`sValue = oPropSet.GetProperty(propName as String)`

GetPropertyCount Method for a Property Set	Returns the number of properties that exist in the current level in the hierarchy.	`Dim oPropSet as SiebelPropertySet`
		`Dim iCount as Long`
		`iCount = oPropSet.GetPropertyCount`

Table 148. Summary of Property Set Methods for the Web Client Automation Server

Method	Description	Format

GetType Method for a Property Set	Returns the value of the type attribute of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim type as String`
		`type = oPropSet.GetType`

GetValue Method for a Property Set	Returns the value of the value attribute of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sValue as String`
		`sValue = oPropSet.GetValue`

InsertChildAt Method for a Property Set	Inserts a child property set in a parent property set at a specific location.	`Dim oPropSet as SiebelPropertySet`
		`oPropSet.InsertChildAt(childObject`
		`as SiebelPropertySet, index as Long)`

PropertyExists Method for a Property Set	Returns a Boolean value that indicates if the property that the argument identifies exists.	`Dim oPropSet as SiebelPropertySet`
		`Dim bool as Boolean`
		`bool =`
		`oPropSet.PropertyExists(propName as`
		`String)`

RemoveChild Method for a Property Set	Removes a child property set from a parent property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.RemoveChild(index as Long)`

RemoveProperty Method for a Property Set	Removes a property from a property set.	`Dim oPropSet as SiebelPropertySet`
		`oPropSet.RemoveProperty(propName as`
		`String)`

Reset Method for a Property Set	Removes every property and child property set from a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.Reset`

SetProperty Method for a Property Set	Sets a value in the property of a property set.	`Dim oPropSet as SiebelPropertySet`
		`oPropSet.SetProperty(propName as`
		`String, propValue as String)`

SetType Method for a Property Set	Sets the value for the type attribute of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetType(value as String)`
SetValue Method for a Property Set	Sets the value for the value attribute of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetValue(value as String)`

12 Mobile Web Client Automation Server Quick Reference

This chapter describes summary information for the Mobile Web Client Automation Server. It includes the following topics:

- Application Methods for the Mobile Web Client Automation Server on page 399
- Business Component Methods for the Mobile Web Client Automation Server on page 402
- Business Object Methods for the Mobile Web Client Automation Server on page 406
- Business Service Methods for the Mobile Web Client Automation Server on page 407
- Property Set Methods for the Mobile Web Client Automation Server

Application Methods for the Mobile Web Client Automation Server

Table 149 describes a summary of application methods you can use with the Mobile Web Client Automation Server. It does not include object interface methods that Siebel CRM does not call directly from an application instance. For information about methods that Siebel CRM calls with the InvokeMethod method on an application, see "LoadObjects Method for an Application" on page 150.

Table 149. Summary of Application Methods for the Mobile Web Client Automation Server

Method	Description	Format

ActiveBusObject Method for an Application	Returns the name of the business object that the active view references.	`Dim application as SiebelWebApplication`
		`Dim busObject as SiebelBusObject`
		`Set busObject = application.ActiveBusObject`

ActiveViewName Method for an Application	Returns the name of the active view.	`Dim application as SiebelWebApplication`
		`Dim sView as String`
		`sView = application.ActiveViewName`

CurrencyCode Method for an Application	Returns the currency code that is associated with the division of the user position.	`Dim application as SiebelWebApplication`
		`Dim sCur as String`
		`sCur = Application.CurrencyCode`

EnableExceptions Method for an Application	Enables or disables native COM error handling.	`Dim application as SiebelWebApplication` `application.EnableExceptions(bEnable as Boolean)`

Table 149. Summary of Application Methods for the Mobile Web Client Automation Server

Method	Description	Format
GetBusObject Method for an Application	Creates a new instance of a business object.	```Dim application as SiebelWebApplication Dim busObject as SiebelBusObject set busObject = application.GetBusObject(busobjName as String)```
GetLastErrCode Method for an Application	Returns the error code for the error that Siebel CRM logged most recently.	```Dim application as SiebelWebApplication Dim iErr as Integer iErr = application.GetLastErrCode```
GetLastErrText Method for an Application	Returns the text message for the error that Siebel CRM logged most recently.	```Dim application as SiebelWebApplication Dim sText as String sText = application.GetLastErrText```
GetProfileAttr Method for an Application	Returns the name of an attribute in a user profile.	```Dim application as SiebelWebApplication Dim profValue as String profValue = application.GetProfileAttr(profName as String)```
GetService Method for an Application	Locates a business service. If this business service is not already running, then Siebel CRM starts it.	```Dim application as SiebelWebApplication Dim oService as SiebelService set oService = Application.GetService(serviceName as String)```
GetSharedGlobal Method for an Application	Returns the shared global variables.	```Dim application as SiebelWebApplication Dim name as String name = application.GetSharedGlobal (sName as String)```
InvokeMethod Method for an Application	Calls a method.	```Dim application as SiebelWebApplication application.InvokeMethod(methodName as String, methArg1, methArg2, methArgN as String or StringArray)```
Login Method for an Application	Allows external applications to log in to the Mobile Web Client Automation Server.	```Dim application as SiebelWebApplication Dim sErr as String sErr = application.Login(connectString as String, userName as String, password as String)```
LoginId Method for an Application	Returns the login ID of the user who started the Siebel application.	```Dim application as SiebelWebApplication Dim sID as string sID = application.LoginId```
LoginName Method for an Application	Returns the login name of the user who started the Siebel application.	```Dim application as SiebelWebApplication Dim sUser as String sUser = application.LoginName```

Table 149. Summary of Application Methods for the Mobile Web Client Automation Server

Method	Description	Format

Logoff Method for an Application	Disconnects the Siebel client from the Siebel Server.	`Dim application as SiebelWebApplication`
		`Dim status as Boolean`
		`Status = application.Logoff`

NewPropertySet Method for an Application	Creates a new property set.	`Dim application as SiebelWebApplication`
		`Dim propset As SiebelPropertySet`
		`set propset = application.NewPropertySet`

PositionId Method for an Application	Returns the position ID of the user position.	`Dim application as SiebelWebApplication`
		`Dim sRow as String`
		`sRow = application.PositionId`

PositionName Method for an Application	Returns the name of the current user position.	`Dim application as SiebelWebApplication`
		`Dim sPosition as String`
		`sPosition = application.PositionName`

SetPositionId Method for an Application	Sets the active position to a Position ID.	`Dim application as SiebelWebApplication`
		`Dim posId as String`
		`Dim status as Boolean`
		`status =`
		`application.SetPositionId(posId)`

SetPositionName Method for an Application	Sets the active position to a position name.	`Dim application as SiebelWebApplication`
		`Dim posName as String`
		`Dim status as Boolean`
		`status =`
		`application.SetPositionName(posName)`

SetProfileAttr Method for an Application	Personalization uses this method to set a value for an attribute in a user profile.	`Dim oApplication as SiebelWebApplication`
		`Dim bool as Boolean`
		`bool = oApplication.SetProfileAttr(name as String, value as String)`

SetSharedGlobal Method for an Application	Sets a shared global variable.	`Dim application as SiebelWebApplication`
		`Dim bool as Boolean`
		`bool =`
		`application.SetSharedGlobal(varName as String, value as String)`

Trace Method for an Application	Appends a message to the trace file.	`Dim application as SiebelWebApplication` `application.Trace(message as String)`

TraceOff Method for an Application	Turns off tracing.	`Dim application as SiebelWebApplication`
		`Dim bool as Boolean`
		`bool = application.TraceOff`

TraceOn Method for an Application	Turns on tracing.	`Dim application as SiebelWebApplication`
		`Dim bool as Boolean`
		`bool = application.TraceOn(filename as String, type as String, Selection as String)`

Business Component Methods for the Mobile Web Client Automation Server

Table 150 describes a summary of business component methods you can use with the Mobile Web Client Automation Server. It does not include object interface methods that Siebel CRM does not call directly from a business component instance. For information about methods that Siebel CRM calls with the InvokeMethod method on a business component, see "Business Component Invoke Methods" on page 252.

Table 150. Summary of Business Component Methods for the Mobile Web Client Automation Server

Method	Description	Format
ActivateField Method for a Business Component	Activates a field.	Dim busComp as SiebelBusComp BusComp.ActivateField(*fieldName* as String)
ActivateMultipleFields Method for a Business Component	Activates multiple fields.	Dim buscomp as SiebelBusComp buscomp.ActivateMultipleFields (oPropSet as SiebelPropertySet)
Associate Method for a Business Component	Creates a new many-to-many relationship for the parent object through an association business component.	Dim busComp as SiebelBusComp busComp.Associate(*whereIndiSiebel cator* as Integer)
BusObject Method for a Business Component	Returns the name of the business object that the business component references.	Dim busComp as SiebelBusComp Dim busObject as SiebelBusObject Set BusObject = busComp.BusObject
ClearToQuery Method for a Business Component	Clears the current query but does not clear sort specifications on a business component.	Dim busComp as SiebelBusComp Dim bool as Boolean bool = busComp.ClearToQuery
DeactivateFields Method for a Business Component	Deactivates the fields that are currently active from the SQL query statement of a business component.	Dim busComp as SiebelBusComp Dim bool as Boolean bool = busComp.DeactivateFields
DeleteRecord Method for a Business Component	Removes the current record from a business component.	Dim busComp as SiebelBusComp Dim bool as Boolean bool = busComp.DeleteRecord
ExecuteQuery Method for a Business Component	Returns a set of business component records.	Dim busComp as SiebelBusComp Dim bool as Boolean bool = busComp.ExecuteQuery(*cursorMode* as Integer)

Table 150. Summary of Business Component Methods for the Mobile Web Client Automation Server

Method	Description	Format

ExecuteQuery2 Method for a Business Component	Returns a set of business component records. Allows you to control the number of records Siebel CRM returns.	`Dim busComp as SiebelBusComp`
		`Dim bool as Boolean`
		`bool =`
		`busComp.ExecuteQuery2(`*cursorMode*
		`as Integer,` *ignoreMaxCursorSize* `as`
		`Boolean)`

FirstRecord Method for a Business Component	Moves the record pointer to the first record in a business component, making that record the current record.	`Dim busComp as SiebelBusComp`
		`Dim bIsRecord as Boolean`
		`bIsRecord = busComp.FirstRecord`

GetAssocBusComp Method for a Business Component	Returns the name of the association business component.	`Dim busComp as SiebelBusComp`
		`Dim AssocBusComp as SiebelBusComp`
		`Set AssocBusComp =`
		`busComp.GetAssocBusComp`

GetFieldValue Method for a Business Component	Returns the value of a field from the current record of a business component.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetFieldValue(`*FieldName* `as`
		`String)`

GetFormattedFieldValue Method for a Business Component	Returns a field value that is in the same format that the Siebel client uses.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetFormattedFieldValue(`*Fi*
		eldName `as String)`

GetLastErrCode Method for a Business Component	Returns the error code for the error that Siebel CRM logged most recently.	`Dim buscomp as SiebelBusComp`
		`Dim iErr as Integer`
		`iErr = buscomp.GetLastErrCode`

GetLastErrText Method for a Business Component	Returns the text message for the error that Siebel CRM logged most recently.	`Dim busComp as SiebelBusComp`
		`Dim sErr as String`
		`sErr = busComp.GetLastErrText`

GetMultipleFieldValues Method for a Business Component	Returns a value for each field specified in a property set.	`Dim buscomp as SiebelBusComp`
		`buscomp.GetMultipleFieldValues(oP`
		`ropSet as SiebelPropertySet,`
		`PValues as SiebelPropertySet)`

GetMVGBusComp Method for a Business Component	Returns the multivalue group business component that is associated a business component field.	`Dim busComp as SiebelBusComp`
		`Dim mVGBusComp as SiebelBusComp`
		`set mVGBusComp =`
		`busComp.GetMVGBusComp(`*FieldName* `as`
		`String)`

GetNamedSearch Method for a Business Component	Returns the name of a search specification.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetNamedSearch(`*SearchName*
		`as String)`

Table 150. Summary of Business Component Methods for the Mobile Web Client Automation Server

Method	Description	Format

GetPicklistBusComp Method for a Business Component	Returns the name of the pick business component that is associated with a field in the current business component.	`Dim busComp as SiebelBusComp`
		`Dim pickBusComp as SiebelBusComp`
		`Set pickBusComp =`
		`busComp.GetPicklistBusComp(FieldName as String)`

GetSearchExpr Method for a Business Component	Returns the current search expression that is defined for a business component.	`Dim busComp as SiebelBusComp`
		`Dim sExpr as String`
		`sExpr = busComp.GetSearchExpr`

GetSearchSpec Method for a Business Component	Returns the search specification that is defined for a business component.	`Dim busComp as SiebelBusComp`
		`Dim sSpec as String`
		`sSpec =`
		`busComp.GetSearchSpec(FieldName as String)`

GetUserProperty Method for a Business Component	Returns the value of a user property.	`Dim busComp as SiebelBusComp`
		`Dim sValue as String`
		`sValue =`
		`busComp.GetUserProperty(propertyName as String)`

GetViewMode Method for a Business Component	Returns the visibility mode for a business component.	`Dim busComp as SiebelBusComp`
		`Dim iMode as Integer`
		`iMode = busComp.GetViewMode`

InvokeMethod Method for a Business Component	Calls a method.	`Dim busComp as SiebelBusComp`
		`Dim sReturn as String`
		`sReturn = busComp.InvokeMethod(`
		`methodName as String, methArg1,`
		`methArg2, methArgN as String or`
		`StringArray)`

LastRecord Method for a Business Component	Moves the record pointer to the last record in a business component.	`Dim busComp as SiebelBusComp`
		`Dim bReturn as Boolean`
		`bReturn = busComp.LastRecord`

Name Method for a Business Component	Returns the name of a business component.	`Dim busComp as SiebelBusComp`
		`Dim sName as String`
		`sName = busComp.Name`

NewRecord Method for a Business Component	Adds a new record to a business component.	`Dim busComp as SiebelBusComp`
		`Dim bool as Boolean`
		`bool =`
		`busComp.NewRecord(whereIndicator as Integer)`

NextRecord Method for a Business Component	Moves the record pointer to the next record in a business component, making that record the current record.	`Dim busComp as SiebelBusComp`
		`Dim bReturn as Boolean`
		`bReturn = busComp.NextRecord`

Table 150. Summary of Business Component Methods for the Mobile Web Client Automation Server

Method	Description	Format

ParentBusComp Method for a Business Component	Returns the name of a parent business component.	`Dim busComp as SiebelBusComp`
		`Dim parentBusComp as SiebelBusComp`
		`Set parentBusComp =`
		`busComp.ParentBusComp`

Pick Method for a Business Component	Places the currently chosen record in a pick business component into the appropriate fields of the parent business component.	`Dim busComp as SiebelBusComp` `busComp.Pick`

PreviousRecord Method for a Business Component	Moves the record pointer to the previous record in a business component, making that record the current record.	`Dim busComp as SiebelBusComp`
		`Dim bReturn as Boolean`
		`bReturn = busComp.PreviousRecord`

RefineQuery Method for a Business Component	Refines a query.	`Dim busComp as SiebelBusComp` `busComp.RefineQuery`
SetFieldValue Method for a Business Component	Sets a new value in a field for the current record of a business component.	`Dim busComp as SiebelBusComp` `busComp.SetFieldValue(`*`FieldName`*` as String, `*`FieldValue`*` as String)`
SetFormattedFieldValue Method for a Business Component	Sets a new value in a field in the current record of a business component. It accepts the field value in the current local format.	`Dim busComp as SiebelBusComp` `busComp.SetFormattedFieldValue(`*`FieldName`*` as String, `*`FieldValue`*` as String)`
SetMultipleFieldValues Method for a Business Component	Sets new values in the fields of the current record of a business component.	`Dim buscomp as SiebelBusComp` `buscomp.SetMultipleFieldValues(oPropSet as SiebelPropertySet)`
SetNamedSearch Method for a Business Component	Sets a named search specification on a business component.	`Dim busComp as SiebelBusComp` `busComp.SetNamedSearch(`*`searchName`*` as String, `*`searchSpec`*` as String)`
SetSearchExpr Method for a Business Component	Sets the search expression for a business component.	`Dim busComp as SiebelBusComp` `busComp.SetSearchExpr(`*`searchSpec`*` as String)`
SetSearchSpec Method for a Business Component	Sets the search specification for a business component.	`Dim busComp as SiebelBusComp` `busComp.SetSearchSpec(`*`FieldName`*` as String, `*`searchSpec`*` as String)`
SetSortSpec Method for a Business Component	Sets the sort specification for a business component.	`Dim busComp as SiebelBusComp` `busComp.SetSortSpec(`*`sortSpec`*` as String)`

Table 150. Summary of Business Component Methods for the Mobile Web Client Automation Server

Method	Description	Format
SetUserProperty Method for a Business Component	Sets the value of a user property in a business component.	`Dim busComp as SiebelBusComp busComp.SetUserProperty(`*propertyName* ` as String, `*newValue*` as String)`
SetViewMode Method for a Business Component	Sets the visibility type for a business component.	`Dim buscomp as SiebelBusComp buscomp.SetViewMode(mode As Integer)`
UndoRecord Method for a Business Component	Reverses any unsaved modifications made to the record.	`Dim busComp as SiebelBusComp busComp.UndoRecord`
WriteRecord Method for a Business Component	Saves to the Siebel database any modifications made to the current record.	`Dim busComp as SiebelBusComp busComp.WriteRecord`

Business Object Methods for the Mobile Web Client Automation Server

Table 151 describes a summary of business object methods you can use with the Mobile Web Client Automation Server.

Table 151. Summary of Business Object Methods for the Mobile Web Client Automation Server

Method	Description	Format
GetBusComp Method for a Business Object	Returns the name of a business component.	`Dim busObject as SiebelBusObject Dim busComp as SiebelBusComp set busComp = busObject.GetBusComp(`*BusCompName*` as String)`
GetLastErrCode Method for a Business Object	Returns the error code for the error that Siebel CRM logged most recently.	`Dim busobject as SiebelBusObject Dim iErr as Integer iErr = busobject.GetLastErrCode`
GetLastErrText Method for a Business Object	Returns the text message for the error that Siebel CRM logged most recently.	`Dim busobject as SiebelBusObject Dim sValue as String sValue= busobject.GetLastErrText`
Name Method for a Business Object	Returns the name of the business object.	`Dim busObject as SiebelBusObject Dim sName as String sName = busObject.Name`

Business Service Methods for the Mobile Web Client Automation Server

Table 152 describes a summary of business service methods you can use with the Mobile Web Client Automation Server.

Table 152. Summary of Business Service Methods for the Mobile Web Client Automation Server

Method	Description	Format

GetFirstProperty Method for a Business Service	Returns the name of the first property of a business service.	`Dim oService as SiebelService`
		`Dim sName as String`
		`sName = oService.GetFirstProperty`

GetNextProperty Method for a Business Service	Returns the name of the next property of a business service.	`Dim oService as SiebelService`
		`Dim sName as String`
		`sName = oService.GetNextProperty`

GetProperty Method for a Business Service	Returns the value of a property.	`Dim oService as SiebelService`
		`Dim sValue as String`
		`sValue = oService.GetProperty(propName as String)`

InvokeMethod Method for a Business Service	Calls a method.	`Dim oService as SiebelService` `oService.InvokeMethod(methodName as String, InputArguments as SiebelPropertySet, OutputArguments as SiebelPropertySet)`

Name Method for a Business Service	Returns the name of a business service.	`Dim oService as SiebelService`
		`Dim sName as String`
		`sName = oService.Name`

PropertyExists Method for a Business Service	Returns a Boolean value that indicates if the property that the argument identifies exists.	`Dim oService as SiebelService`
		`Dim bool as Boolean`
		`bool = oService.PropertyExists(propName as String)`

RemoveProperty Method for a Business Service	Removes a property from a business service.	`Dim oService as SiebelService` `oService.RemoveProperty propName as String`
SetProperty Method for a Business Service	Sets a value for a property of a business service.	`Dim oService as SiebelService` `oService.SetProperty(propName as String, propValue as String)`

Property Set Methods for the Mobile Web Client Automation Server

Table 153 describes a summary of the property set methods you can use with the Mobile Web Client Automation Server.

Table 153. Summary of Property Set Methods for the Mobile Web Client Automation Server

Method	Description	Format
AddChild Method for a Property Set	Adds child property sets to a property set.	`Dim oPropSet as SiebelPropertyset` `oPropSet.AddChild(childObject as SiebelPropertySet)`

Copy Method for a Property Set	Returns a copy of a property set.	`Dim oPropSet1 as SiebelPropertyset`
		`Dim oPropSet2 as SiebelPropertyset`
		`set oPropSet2 = oPropSet1.Copy`

GetChild Method for a Property Set	Returns a child property set of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim childPropSet as SiebelPropertySet`
		`set childPropSet = oPropSet.GetChild(index as Long)`

GetChildCount Method for a Property Set	Returns the number of child property sets that exist for a parent property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim iCount as Long`
		`iCount = oPropSet.GetChildCount`

GetFirstProperty Method for a Property Set	Returns the name of the first property in a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropName as String`
		`sPropName = oPropSet.GetFirstProperty`

GetLastErrCode Method for a Property Set	Returns the error code for the error that Siebel CRM logged most recently.	`Dim oPropSet as SiebelPropertySet`
		`Dim iErr as Integer`
		`iErr = oPropSet.GetLastErrCode`

GetLastErrText Method for a Property Set	Returns the text message for the error that Siebel CRM logged most recently.	`Dim oPropSet as SiebelPropertySet`
		`Dim sValue as String`
		`sValue = oPropSet.GetLastErrText`

GetNextProperty Method for a Property Set	Returns the name of the next property in a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropName as String`
		`sPropName = oPropSet.GetNextProperty`

GetProperty Method for a Property Set	Returns the value of a property.	`Dim oPropSet as SiebelPropertySet`
		`Dim sPropVal as String`
		`sPropVal = oPropSet.GetProperty(propName as String)`

GetPropertyCount Method for a Property Set	Returns the number of properties that exist in the current level in the hierarchy.	`Dim oPropSet as SiebelPropertySet`
		`Dim lCount as Long`
		`lCount = oPropSet.GetPropertyCount`

Table 153. Summary of Property Set Methods for the Mobile Web Client Automation Server

Method	Description	Format

GetType Method for a Property Set	Returns the value of the type attribute of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sTypeVal as String`
		`sTypeVal = oPropSet.GetType`

GetValue Method for a Property Set	Returns the value of the value attribute of a property set.	`Dim oPropSet as SiebelPropertySet`
		`Dim sValVal as String`
		`sValVal = oPropSet.GetValue`

InsertChildAt Method for a Property Set	Inserts a child property set in a parent property set at a specific location.	`Dim oPropSet as SiebelPropertySet` `oPropSet.InsertChildAt(childObject as SiebelPropertySet, index as Long)`

PropertyExists Method for a Property Set	Returns a Boolean value that indicates if the property that the argument identifies exists.	`Dim oPropSet as SiebelPropertySet`
		`Dim bool as Boolean`
		`bool = oPropSet.PropertyExists(propName as String)`

RemoveChild Method for a Property Set	Removes a child property set from a parent property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.RemoveChild(index as Long)`
RemoveProperty Method for a Property Set	Removes a property from a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.RemoveProperty(propName as String)`
Reset Method for a Property Set	Removes every property and child property set from a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.Reset`
SetProperty Method for a Property Set	Sets a value in the property of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetProperty(propName as String, propValue as String)`
SetType Method for a Property Set	Sets the value for the type attribute of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetType(value as String)`
SetValue Method for a Property Set	Sets the value for the value attribute of a property set.	`Dim oPropSet as SiebelPropertySet` `oPropSet.SetValue(value as String)`

13 Siebel Java Data Bean Quick Reference

This chapter describes summary information for the Siebel Java Data Bean. It includes the following topics:

- Data Bean Methods for Siebel Java Data Bean on page 411
- Business Component Methods for Siebel Java Data Bean on page 413
- Business Object Methods for Siebel Java Data Bean on page 417
- Business Service Methods for Siebel Java Data Bean on page 417
- Property Set Methods for Siebel Java Data Bean on page 418
- Siebel Exception Methods for Siebel Java Data Bean

For more information about Siebel Java Data Bean, see the Javadoc files that reside in the Siebel_JavaDoc.jar file. This file is typically located in the \siebsrvr\CLASSES folder.

Data Bean Methods for Siebel Java Data Bean

Table 154 describes a summary of Data Bean methods you can use with Siebel Java Data Bean.

Table 154. Summary of Data Bean Methods for Siebel Java Data Bean

Method	Description	Format
Attach Method for an Application	Allows an external application to reconnect to an existing Siebel session.	`boolean attach(String sessionID) throws SiebelException`
CurrencyCode Method for an Application	Returns the currency code that is associated with the division of the user position.	`String currencyCode()`
Detach Method for an Application	Returns a string that contains the Siebel session ID.	`String detach() throws SiebelException`
GetBusObject Method for an Application	Creates a new instance of a business object.	`SiebelBusObject getBusObject(String boName) throws SiebelException`
GetProfileAttr Method for an Application	Returns the name of an attribute in a user profile.	`String getProfileAttr(String attrName) throws SiebelException`

Table 154. Summary of Data Bean Methods for Siebel Java Data Bean

Method	Description	Format
GetService Method for an Application	Locates a business service. If this business service is not already running, then Siebel CRM starts it.	`SiebelService getService(string serviceName) throws SiebelException`
InvokeMethod Method for an Application	Calls a method.	`String invokeMethod(String name, String[] args) throws SiebelException`
Login Method for an Application	Allows an external application to log in to the COM Data Server, COM Data Control, or Siebel Java Data Bean, and to access Siebel objects.	`boolean login(String connString, String userName, String passWord) throws SiebelException`
LoginId Method for an Application	Returns the login ID of the user who started the Siebel application.	`String loginId()`
LoginName Method for an Application	Returns the login name of the user who started the Siebel application.	`String loginName()`
Logoff Method for an Application	Disconnects the Siebel client from the Siebel Server.	`boolean logoff() throws SiebelException`
NewPropertySet Method for an Application	Creates a new property set.	`SiebelPropertySet newPropertySet()`
PositionId Method for an Application	Returns the position ID of the user position.	`String positionId()`
PositionName Method for an Application	Returns the name of the current user position.	`String positionName()`
SetPositionId Method for an Application	Sets the active position to a Position ID.	`boolean setPositionId(String posId) throws SiebelException`
SetPositionName Method for an Application	Sets the active position to a position name.	`boolean setPositionName(String posName) throws SiebelException`
SetProfileAttr Method for an Application	Personalization uses this method to set a value for an attribute in a user profile.	`boolean setProfileAttr(String attrName, String attrValue) throws SiebelException`
Trace Method for an Application	Appends a message to the trace file.	`boolean trace(String message) throws SiebelException`

Table 154. Summary of Data Bean Methods for Siebel Java Data Bean

Method	Description	Format
TraceOff Method for an Application	Turns off tracing.	`boolean traceOff() throws SiebelException`
TraceOn Method for an Application	Turns on tracing.	`boolean traceOn(String filename, String Category, String selection)throws SiebelException`

Business Component Methods for Siebel Java Data Bean

Table 155 describes a summary of business component methods you can use with Siebel Java Data Bean. It does not include object interface methods that Siebel CRM does not call directly from a business component instance. For information about methods that Siebel CRM calls with the InvokeMethod method on a business component, see "Business Component Invoke Methods" on page 252.

Table 155. Summary of Business Component Methods for Siebel Java Data Bean

Method	Description	Format
ActivateField Method for a Business Component	Activates a field.	`boolean activateField(String fieldName) throws SiebelException`
ActivateMultipleFields Method for a Business Component	Activates multiple fields.	`boolean activateMultipleFields(SiebelPropertySet psFields) throws SiebelException`
Associate Method for a Business Component	Creates a new many-to-many relationship for the parent object through an association business component.	`boolean associate(boolean isInsertBefore) throws SiebelException`
BusObject Method for a Business Component	Returns the name of the business object that the business component references.	`SiebelBusObject busObject() throws SiebelException`
ClearToQuery Method for a Business Component	Clears the current query but does not clear sort specifications on a business component.	`boolean clearToQuery() throws SiebelException`
DeactivateFields Method for a Business Component	Deactivates the fields that are currently active from the SQL query statement of a business component.	`boolean deactivateFields()`

Table 155. Summary of Business Component Methods for Siebel Java Data Bean

Method	Description	Format
DeleteRecord Method for a Business Component	Removes the current record from a business component.	`boolean deleteRecord() throws SiebelException`
ExecuteQuery Method for a Business Component	Returns a set of business component records.	`boolean executeQuery(boolean cursorMode) throws SiebelException` If using the ExecuteQuery method with Siebel Java Data Bean, use True for ForwardOnly and False for ForwardBackward.
ExecuteQuery2 Method for a Business Component	Returns a set of business component records. Allows you to control the number of records Siebel CRM returns.	`boolean executeQuery2(boolean cursorMode, boolean ignoreMaxCursorSize) throws SiebelException`
FirstRecord Method for a Business Component	Moves the record pointer to the first record in a business component, making that record the current record.	`boolean firstRecord() throws SiebelException`
GetFieldValue Method for a Business Component	Returns the value of a field from the current record of a business component.	`String getFieldValue(String fieldName) throws SiebelException`
GetFormattedFieldValue Method for a Business Component	Returns a field value that is in the same format that the Siebel client uses.	`String getFormattedFieldValue(String fieldName) throws SiebelException`
GetMultipleFieldValues Method for a Business Component	Returns values for the fields specified in a property set.	`boolean getMultipleFieldValues(SiebelPropertySet Src, SiebelPropertySet result) throws SiebelException`
GetMVGBusComp Method for a Business Component	Returns the multivalue group business component that is associated a business component field.	`SiebelBusComp getMVGBusComp(String fieldName) throws SiebelException`
GetNamedSearch Method for a Business Component	Returns the name of a search specification.	`String getNamedSearch(String searchName) throws SiebelException`
GetPicklistBusComp Method for a Business Component	Returns the name of the pick business component that is associated with a field in the current business component.	`SiebelBusComp getPicklistBusComp(String fieldName) throws SiebelException`
GetSearchExpr Method for a Business Component	Returns the current search expression that is defined for a business component.	`String getSearchExpr() throws SiebelException`

Table 155. Summary of Business Component Methods for Siebel Java Data Bean

Method	Description	Format
GetSearchSpec Method for a Business Component	Returns the search specification that is defined for a business component.	`String getSearchSpec(String fieldName) throws SiebelException`
GetUserProperty Method for a Business Component	Returns the value for the specified property.	`String getUserProperty(String property) throws SiebelException`
GetViewMode Method for a Business Component	Returns the visibility mode for a business component.	`int getViewMode()`
InvokeMethod Method for a Business Component	Calls a method.	`String invokeMethod(String methodName, String[] methArg1, methArg2, methArgN) throws SiebelException`
LastRecord Method for a Business Component	Moves the record pointer to the last record in a business component.	`boolean lastRecord() throws SiebelException`
Name Method for a Business Component	Returns the name of a business component.	`String name()`
NewRecord Method for a Business Component	Adds a new record to a business component.	`boolean newRecord(boolean isInsertBefore) throws SiebelException`
NextRecord Method for a Business Component	Moves the record pointer to the next record in a business component, making that record the current record.	`boolean nextRecord() throws SiebelException`
ParentBusComp Method for a Business Component	Returns the name of a parent business component.	`SiebelBusComp parentBusComp() throws SiebelException`
Pick Method for a Business Component	Places the currently chosen record in a pick business component into the appropriate fields of the parent business component.	`boolean pick() throws SiebelException`
PreviousRecord Method for a Business Component	Moves the record pointer to the previous record in a business component, making that record the current record.	`boolean previousRecord() throws SiebelException`
RefineQuery Method for a Business Component	Refines a query.	`boolean refineQuery() throws SiebelException`

Table 155. Summary of Business Component Methods for Siebel Java Data Bean

Method	Description	Format
Release Method for a Business Component	Releases a business component and the resources for this business component that exist on the Siebel Server.	`void release()`
SetFieldValue Method for a Business Component	Sets a new value in a field for the current record of a business component.	`boolean setFieldValue(String fieldName, String fieldValue) throws SiebelException`
SetFormattedFieldValue Method for a Business Component	Sets a new value in a field in the current record of a business component. It accepts the field value in the current local format.	`boolean setFormattedFieldValue(String fieldName, String fieldValue) throws SiebelException`
SetMultipleFieldValues Method for a Business Component	Sets new values to the multiple fields specified in the property set for the current record of a business component.	`boolean setMultipleFieldValues(SiebelPropertySet psFields) throws SiebelException`
SetNamedSearch Method for a Business Component	Sets a named search specification on a business component.	`boolean setNamedSearch(String searchName, String searchText) throws SiebelException`
SetSearchExpr Method for a Business Component	Sets the search expression for a business component.	`boolean setSearchExpr(String searchExpr) throws SiebelException`
SetSearchSpec Method for a Business Component	Sets the search specification for a business component.	`boolean setSearchSpec(String fieldName, String searchSpec) throws SiebelException`
SetSortSpec Method for a Business Component	Sets the sort specification for a business component.	`boolean setSortSpec(String sortSpec) throws SiebelException`
SetUserProperty Method for a Business Component	Sets the value of a user property in a business component.	`boolean setUserProperty(String propName, String propVal)`
SetViewMode Method for a Business Component	Sets the visibility type for a business component.	`boolean setViewMode(int mode) throws SiebelException`
UndoRecord Method for a Business Component	Reverses any unsaved modifications made to the record.	`boolean undoRecord() throws SiebelException`
WriteRecord Method for a Business Component	Saves to the Siebel database any modifications made to the current record.	`boolean writeRecord() throws SiebelException`

Business Object Methods for Siebel Java Data Bean

Table 156 describes a summary of business object methods you can use with Siebel Java Data Bean.

Table 156. Summary of Business Object Methods for Siebel Java Data Bean

Method	Description	Format
GetBusComp Method for a Business Object	Returns the name of a business component.	`SiebelBusComp getBusComp(String busCompName) throws SiebelException`
Name Method for a Business Object	Returns the name of the business object.	`String name()`
Release Method for a Business Object	Releases a business object and the resources for this business object on the Siebel Server.	`void release()`

Business Service Methods for Siebel Java Data Bean

Table 157 describes a summary of business service methods you can use with Siebel Java Data Bean.

Table 157. Summary of Business Service Methods for Siebel Java Data Bean

Method	Description	Format
Business Service Methods	Returns the name of the first property of a business service.	`String getFirstProperty()`
GetNextProperty Method for a Business Service	Returns the name of the next property of a business service.	`String getNextProperty()`
GetProperty Method for a Business Service	Returns the value of a property.	`String getProperty(String propName) throws SiebelException`
InvokeMethod Method for a Business Service	Calls a method.	`boolean invokeMethod(String methodName, SiebelPropertySet inputPropertySet, SiebelPropertySet outputPropertySet) throws SiebelException`
Name Method for a Business Service	Returns the name of a business service.	`String Name()`
PropertyExists Method for a Business Service	Returns a Boolean value that indicates if the property that the argument identifies exists.	`boolean propertyExists(String propName) throws SiebelException`

Table 157. Summary of Business Service Methods for Siebel Java Data Bean

Method	Description	Format
Release Method for a Business Service	Releases a business service and the resources that this business service uses on the Siebel Server.	`void release()`
RemoveProperty Method for a Business Service	Removes a property from a business service.	`void removeProperty(String propName) throws SiebelException`
SetProperty Method for a Business Service	Sets a value for a property of a business service.	`void setProperty(String propName, String propValue) throws SiebelException`

Property Set Methods for Siebel Java Data Bean

Table 158 describes a summary of property set methods you can use with Siebel Java Data Bean.

Table 158. Summary of Property Set Methods for Siebel Java Data Bean

Method	Description	Format
AddChild Method for a Property Set	Adds child property sets to a property set.	`int addChild(SiebelPropertySet propertySet)`
Copy Method for a Property Set	Returns a copy of a property set.	`SiebelPropertySet copy(SiebelPropertySet propertySet)`
GetByteValue Method for a Property Set	Returns a byte array if a byte value is set.	`public byte[] getByteValue()`
GetChild Method for a Property Set	Returns a child property set of a property set.	`SiebelPropertySet getChild(int index)`
GetChildCount Method for a Property Set	Returns the number of child property sets that exist for a parent property set.	`int getChildCount()`
GetFirstProperty Method for a Property Set	Returns the name of the first property in a property set.	`String getFirstProperty()`
GetNextProperty Method for a Property Set	Returns the name of the next property in a property set.	`String getNextProperty()`
GetProperty Method for a Property Set	Returns the value of a property.	`String getProperty(String propertyName)`

Table 158. Summary of Property Set Methods for Siebel Java Data Bean

Method	Description	Format
GetPropertyCount Method for a Property Set	Returns the number of properties that exist in the current level in the hierarchy.	`int GetPropertyCount()`
GetType Method for a Property Set	Returns the value of the type attribute of a property set.	`String getType()`
GetValue Method for a Property Set	Returns the value of the value attribute of a property set.	`String getValue()`
InsertChildAt Method for a Property Set	Inserts a child property set in a parent property set at a specific location.	`boolean insertChildAt(SiebelPropertySet propertySet, int index)`
PropertyExists Method for a Property Set	Returns a Boolean value that indicates if the property that the argument identifies exists.	`boolean propertyExists(String propertyName)`
RemoveChild Method for a Property Set	Removes a child property set from a parent property set.	`boolean removeChild(int index)`
RemoveProperty Method for a Property Set	Removes a property from a property set.	`boolean removeProperty(String propertyName)`
Reset Method for a Property Set	Removes every property and child property set from a property set.	`boolean reset()`
SetByteValue Method for a Property Set	Sets the value portion of a property set.	`public void setByteValue(byte[] value)`
SetProperty Method for a Property Set	Sets a value in the property of a property set.	`boolean setProperty(String propertyName, String propertyValue)`
SetType Method for a Property Set	Sets the value for the type attribute of a property set.	`boolean setType(String type)`
SetValue Method for a Property Set	Sets the value for the value attribute of a property set.	`boolean setValue(String value)`

Siebel Exception Methods for Siebel Java Data Bean

Table 159 describes a summary of Siebel exception methods that you can use with Siebel Java Data Bean. The Siebel Java Data Bean is one of Oracle's Siebel Object Interfaces.

Table 159. Summary of Siebel Exception Methods for Siebel Java Data Bean

Method	Description	Format
GetErrorCode Method	Returns a numeric error code.	`int getErrorCode()`
GetErrorMessage Method	Returns an error message.	`String getErrorMessage()`

Index