Siebel Personalization Administration Guide

Siebel 2018

April 2018

ORACLE

Copyright © 2005, 2018 Oracle and/or its affiliates. All rights reserved.

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

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

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

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

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

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

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

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

Documentation Accessibility

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

Access to Oracle Support

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

Contents

Chapter 7: Testing Personalization Rules

Appendix A: Operators for Building Condition Expressions

Appendix B: Functions

Index

1 What's New in This Release

What's New in Siebel Personalization Administration Guide, Siebel 2018

No new features have been added to this guide for this release. This guide has been updated to reflect only product name changes.

NOTE: Siebel 2018 is a continuation of the Siebel 8.1/8.2 release

2 About Siebel Personalization

Oracle's Siebel Personalization provides an integrated multichannel personalization platform for customizing enterprise-wide interactions with customers, partners, and employees. Personalization is part of the core infrastructure of Siebel Business Applications and is fully integrated into the Siebel architecture.

This section includes the following topics:

Siebel Personalization Components

Table 1 lists the Siebel Personalization main components.

Table 1. Siebel Personalization Components

Component	Description
Personalization rules	Controls visibility and content in user interface elements such as applets and views. Personalization rules are evaluated at run time and use the available information about the user, such as his or her profile and any transaction information. Rules can control the visibility of views and applets or control the contents displayed in an applet. Personalization rules are created and managed in the Personalization Administration screens.
Run-time events	Allows applications to change in real time based on user inputs. Run-time events are triggered in response to user actions. Business managers can configure the actions in response to any event. Run-time events and the actions that trigger them are created and managed in the Administration - Runtime Events screen.

Personalization Features

Siebel Personalization provides the following benefits.

End-User Layout Customization

Users can customize the layout directly from portal-style pages. Table 2 lists how to move hide, and minimize applets.

Table 2. Using the Edit Layout Button to Move, Edit, and Minimize Applets

To	Then
Move applets to various positions in the view	Click Edit Layout. This allows end users to express their preference as to where applets should appear.
Hide applets	Click Hide. Use Edit Layout to show the button again, assuming that the administrator allows this to happen. This allows end users to show only those applets that are most important to them.
Minimize applets	Click Collapse so that they appear on the view, but only the title bar appears. This allows end users to keep information that is infrequently used hidden until it is needed.

Personalization for Another User

Users have access to their own profile when they log in, but they can also read the profile of other users with whom they are interacting.

For example, a call center agent can load the profile of the user with whom he or she is speaking to trigger personalization rules on that user's behalf. This allows personalization to work consistently across all channels.

Personalization Business Rules Designer

The Personalization Business Rules Designer allows business managers to implement personalization rules without learning complex programming languages.

- The Rules Designer uses Siebel Query Language to construct expressions. The if-then rule structure facilitates rule writing.
- The template-based approach allows business managers to write rules by filling out a template wizard with the appropriate key words, while not getting in the way of expert users who just want to type rules in the expression fields.
- The Rules Designer provides choices of rule elements based on the context of the rule. The context comes from the place where the Rules Designer is invoked: action, applet, rule, and so on.

For more information, see About the Personalization Business Rules Designer.

Advanced Testing Environment

Creating personalization rules is by nature a dynamic process. How the system responds to a user depends on who that user is, and what actions the user performs while using the system. Testing personalization rules before deploying them on a live environment allows an administrator to assess whether the rules are having the desired effect.

- The administrator can test the effects of content targeting rules, and personalization events and actions in employee, customer, and partner applications.
- The administrator can load the profiles of dummy users. These users can simulate the effect of personalization rules on hypothetical user profiles.

For more information, see Using Test Mode to Test Personalization Rules.

Personalization Within the Siebel Architecture

Siebel Personalization is installed automatically as part of the Siebel object manager. Personalization is administered in run time, providing maximum control without the necessity of recompiling the repository. This allows business managers to anticipate customer needs based on their actions and react to them in real time.

In the Siebel architecture, there are objects that define the user interface, business objects that define the business rules, a data manager for moving data in and out of the database, and the database itself.

Siebel Personalization forms a layer between the user interface and the business objects. Figure 1 shows the Siebel architecture, which consists of run-time administration, data, and business objects from the Siebel Repository.

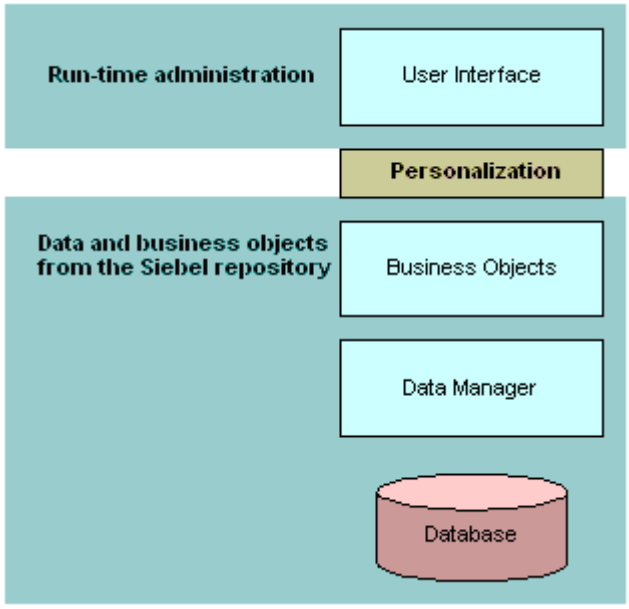


Figure 1. Siebel Architecture

Getting Started with Siebel Personalization

Use the following process to start using Siebel Personalization. Each of these activities is covered in more detail in this document.

NOTE: Siebel Personalization is active in all applications by default.

- Make sure that all persons engaged in personalization administration are assigned a responsibility in the Siebel system that allows them to see all Personalization Administration and Administration - Runtime Events views. This is especially important if you have configured multiple organizations in the Siebel system.

  For more information, see Roles and Responsibilities on page 14.

  If you delegate any authority to channel partners for administering personalization, make sure your LDAP server is configured to expect these logins.

  For information on configuring the LDAP server, see *Siebel Security Guide*.

- Analyze business requirements for personalized behavior. Consider requirements for employees and channel partners as well as customers.

  For more information, see Understanding the Target Audience on page 15.

- Create persistent user profile attributes as necessary to track the user information needed for personalization.

- Create rules to control the visibility of views, applets, and content based on user profile attributes and content attributes.

- Register events that you want to monitor in the Events and Events Aliases view.

- Configure actions to be triggered when events of interest occur.

  For more information, see Process of Creating Action Sets on page 38.

NOTE: When a user is not specifically logged into the Siebel application, the user actually has a session as a guest user. Using this session, an anonymous user's experience can be personalized. For example, using personalization rules an anonymous user can be configured to see a different home page from that of a logged in user.

Roles and Responsibilities

Table 3 lists the main Siebel Personalization roles.

Table 3. Main Siebel Personalization Roles

Role	Description
Business manager or analyst	Analyzes and defines personalization requirements that meet business needs in the most cost-effective manner. Requests personalization behavior based on the experience of employees, customers, or channel partners.
Personalization administrator	Translates the business requirements into personalization rules, rule sets, events, and actions. Also responsible for reviewing and maintaining the implementation. The personalization administrator must have the responsibility associated with the Siebel administrator to have the necessary access to perform these duties.

Personalization Terminology

Table 4 lists some of the terms used in this section.

Table 4. Personalization Terms

Term	Description
Personalization	The ability to provide specific information and application functionality based on known or inferred characteristics of a user. Decisions are made in real time about what to show or not show to a user based on the profile built up over time.
Content	All the visible components of a display, including data, text, images, and structural elements.
Content Targeting	The ability to display information dynamically to users based on their interests and behavior, either explicitly stated or implicitly observed.
Profiling	The ability to collect information about users in order to deliver the most appropriate and effective content. Profiling is accomplished using registration information, by explicit polling, and by real-time observation of user behavior.
Rules-Based Filtering	The ability to show users information based on rules defined by a system administrator or marketing manager.

Personalization and Content Targeting

Siebel Personalization makes it possible to deliver a customized message to a large number of employees, customers, and channel partners across all channels. Driving personalization decisions from a business perspective is important, that is, to understand the business issues you are trying to address before implementing personalization.

Content Targeting Diagram

Siebel Personalization lets you combine together your detailed understanding of your content and your users. Figure 2 illustrates that by using content and users you define content attributes and build the user profile. Both are used to create rules.

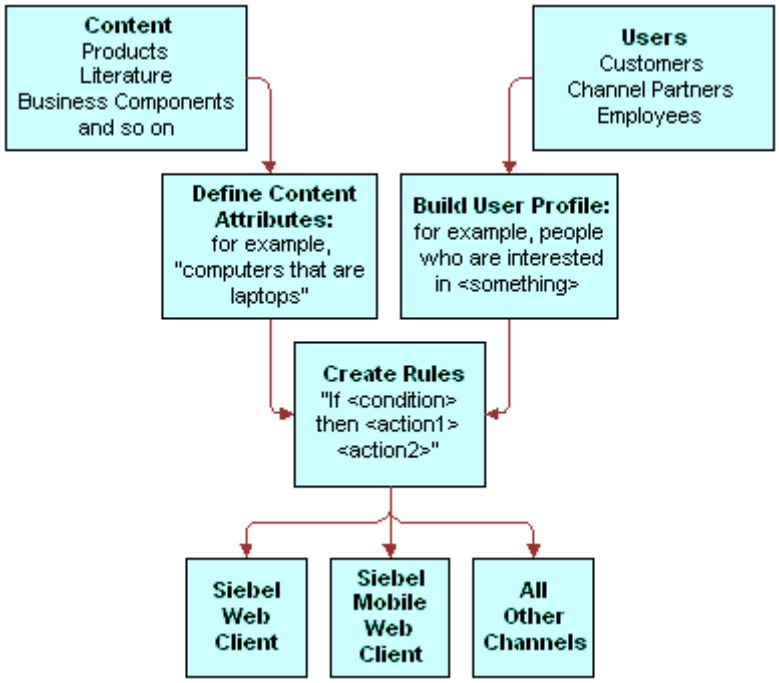


Figure 2. Content Targeting

Understanding the Target Audience

The key to implementing Siebel Personalization is to understand the target audience, whether customer, channel partner, or employee. When the target audience is identified, then the business imperative for that audience must be clearly defined.

You should consider the following:

- What views should be shown or hidden, depending on the user interacting with the system?
- Should certain applets be shown or hidden?
- What opportunities are there to show customized content to users based on their profiles?

- What events take place that should be tracked? Alternatively, are there business requirements for tracking certain events?
- Can benefits be derived from delivering personalized content to employees as well as customers and channel partners?

Once a personalization system is in place, then the business manager and personalization administrator must develop an ongoing strategy to monitor and implement changes in business requirements as well as evaluate the effectiveness of the current personalization plan.

Personalization Usage Scenarios

Users experience Siebel Personalization as content targeted to them, such as on the home page. The personalized home page displays a calendar, and other content specific to the user.

The following are scenarios of using Siebel Personalization:

- Real-Time Product Recommendations Scenario
- Personalization Through All Channels Scenario
- Third-Party Personalization Engines Integration Scenario

Real-Time Product Recommendations Scenario

This section consists of the following topics:

- About the Real-Time Product Scenario
- Implementing Real-Time Product Scenario

About the Real-Time Product Scenario

A company sells products in many areas. A typical customer is likely to be interested in a small subset of those products.

The company wants to:

- Recommend only those products that are of interest to that particular customer.
- Make sure that the products shown are in stock and are profitable for the company.

Implementing Real-Time Product Scenario

Use the following steps to implement this:

- Create or borrow the recommended products user interface (a recommended items applet already exists in).
- Write personalization rules to target content based on user profile attributes or other transaction data, such as past orders. For example, you can show customers products based on the industry they specified when they registered.

- Write personalization rules to recommend products based on customer behavior. For example, when a customer adds a product to his or her Shopping Cart, the ProductInCart persistent or dynamic user profile attribute can be set. Products defined in the Product Administration screen as being related to that product can be displayed to the customer for cross-sell and upsell opportunities.

  The following rule:

  ```
  EXISTS([Related Product] = GetProfileAttr("ProductInCart"))
  ```

  shows products that are related to products in the Siebel eSales Shopping Cart.

- Create a business service to access external data or check for business constraints. For example, you can use an ERP system to determine whether a product is in stock and how well it has been selling recently.
- Write personalization rules to access the business service and to show or hide products based on availability and profitability.

When customers enter the company's Web site, they see products appropriate to their industry or other interests. After adding products to the Shopping Cart, new products are recommended that complement the ones they have already selected. The customers are more likely to buy products that are useful and readily apparent.

Personalization Through All Channels Scenario

The section consists of the following topics:

About the Personalization Through All Channels Scenario

A company wants to provide the same level of personalized service through all its channels, using the same set of personalization rules.

The company has the Siebel eSales Web site that it uses to sell its products. On this Web site the company recommends products to its customers.

The company also runs a call center. When a customer calls the company's call center, the company wants its call center agents to recommend products based on the profile of the caller rather than the profiles of the agents.

The company also wants to recommend the same products to the customer as those recommended on the Web site.

Implementing Personalization Through All Channels Scenario

Use the following steps to implement this:

- Create or borrow the recommended products user interface (a recommended items applet already exists in Siebel eSales).
- Create personalization rules to target content based on the user profiles for both the logged-in user (the call center agent) and the secondary user (caller). These are called the Me and You profiles, respectively.
- Load the secondary user profile (You) when a customer calls.

When the customer calls the company, the call center agent knows the customer's interests, concerns, service request history, and past purchases. The customer gets personalized attention and a more efficient response from the call center agent. The agent can recommend useful products to the customer for cross-sell and upsell opportunities.

Third-Party Personalization Engines Integration Scenario

This section consists of the following topics:

About the Third-Party Personalization Engines Integration Scenario

A company wants to prevent its customers from switching to a competitor's product by giving them special offers.

The company analyzes historical data and discovers a trend that is related to customer defection. Based on this analysis, the company creates a new model that uses relevant customer data to predict the probability of a customer switching to a competitor's product.

You might create this model using a third-party personalization engine. Several such models exist for the pharmaceutical industry.

The company wants to monitor customer actions in real time while running the model. Based on the model's predictions, the company wants to make a special offer to likely defectors to prevent them from switching.

Implementing Third-Party Personalization Engines Integration Scenario

Use the following steps to implement this:

- Analyze historical data and create a new model that accesses the relevant data about customers and predicts the probability of their defection.
- Create a business service that integrates Siebel with the third-party software and runs the predictive model.

- Write personalization rules that trigger the business service in response to customer actions. These run-time personalization events are passed to the third-party engine through the business service interface.
- Write personalization rules to deliver a special offer to a customer based on results from the model.

When the customer enters the Web site, he or she might fill out a survey or just browse the site. A special offer appears in a new browser window. Because the offer is targeted specifically to the customer, he or she takes the time to look at it instead of exiting the Web site.

3 Managing User Profiles

User profiles consist of attributes that you want to track for each user, so time invested in analyzing your audience is well spent.

As you determine your business requirements for personalization, review the standard user profile attributes available and decide if you need to create additional attributes.

This section includes the following topics:

About User Profile Attributes

User profile attributes can include personal information about the user, such as name, address, email address, and hobbies, or a variety of extended information, such as company, industry, position, responsibilities, items purchased, information requested, or service provided.

User profile attributes are configurable. Once modeled, they are available across all channels, whether the user contacts your organization through your Web site, your call center, or any other channel managed through your Siebel system.

About Managing User Profile Attributes

User profile attributes can be either dynamic or persistent, and can be managed in a variety of ways across many channels. Table 5 lists examples of managing user profile attributes.

Table 5. Managing User Profile Attribute Examples

Channel	For example
Self–administration of a user's own attributes	Updating his or her mailing address. These attributes make up the Me user profile.

Table 5. Managing User Profile Attribute Examples

Channel	For example
A call center agent for customers or others contacting your organization by phone	A service request. The profile of the: ■ Call center agent profile is the Me profile. ■ Customer for whom the agent is managing the attributes is called the You profile. The You profile is represented by the current contact in the customer dashboard.
A field or sales representative administering customer or contact profile attributes remotely on a site visit	Then synchronizing the changes with a corporate database. User profile attributes must be specified exactly as they appear in the business component, and are case sensitive. For example, if `Channel` is the correct user profile attribute, it must not be spelled `channel`.

About Persistent User Profile Attributes

The persistent user profile consists of persistent user profile attributes, both individual and organizational. These attributes are stored in the Siebel database.

Persistent profile attributes exist in the base tables or extension tables of the S_PARTY table.

The persistent profile attributes can be:

- Person-based or organization-based. They appear as fields in the Personalization Profile business component.
- Present in any S_PARTY-based business component, but the Siebel Personalization Engine loads only those defined in the Personalization Profile business component. All organization-based attribute names start with an `Org.` prefix.

This section contains the following topics:

About Storing Persistent User Profile Attributes

All persistent user profile attributes are stored in the Personalization Profile business component, which has two sets of fields, as shown in Table 6.

Table 6. Persistent User Profile Attributes

Field	Description
Person-related	All person-related fields are based on the S_PARTY table or its extension tables and joins that point to a person's attributes. Examples of such extension tables are S_CONTACT, S_CONTACT_X, and S_USER. The best way to define a new field is to find a similar one in the person-based business components, such as User and Employee, and use it as a prototype.
Organization-related	All organization-related fields start with Org. and are based on the S_PARTY table or its extension tables and joins that point to an organization's attributes. Examples of such extension tables are S_ADDR_ORG, S_ORG_EXT, and S_POSTN. The best way to define a new field is to find a similar one in the organization-based business components, such as Account and Internal Division, and use it as an example.

Attribute Components Obsolete in Version 7

In version 7 the following attribute components are obsolete:

- Contact Profile Attributes business component
- Account Profile Attributes business component
- Employee Profile Attributes business component
- Internal Division Profile Attributes business component

Personalization Profile Business Component Used in Version 7

In version 6, these business components are used to store persistent user profile attributes, but in version 7, the Personalization Profile business component is used instead. Any reference to the user profiles in those business components have to be modified.

Basic information such as name and email address should be stored as persistent attributes. Your business needs will determine what other attributes are important to store from one session to another. In general, the most frequently used attributes are made part of the persistent profile.

Adding New Persistent User Profile Attributes

You can add new persistent user profile attributes by adding columns to the S_PARTY table or one of its extension tables, and then exposing these columns as business component fields. New profile attributes are available after the repository file is recompiled.

Querying for Persistent User Profile Attributes

When you query for persistent user profile attributes in the Personalization Profile business component, it is important to use the Org. prefix organization-related fields. The following process occurs:

NOTE: A similar process occurs when persistent user profile attributes are saved, for example when exiting the application.

- Position the business component to the record corresponding to the person whose attributes are being read.
- Read all person-related fields.
- Use either the Primary Account Id Internal or Primary Division Id Internal field (whichever is not empty) as an Id field to query for the organization-related fields.

  NOTE: If both the Primary Account Id Internal and Primary Division Id Internal fields are populated, then the Org.* fields are based on the Primary Account Id Internal.

  Because both person and organization records are stored in the S_PARTY table and its extensions, you can use the same business component to query for these fields.
- Read all fields with names Org.*.

Saving Modified Persistent Attributes

Table 7 describes saving modified persistent attributes.

Table 7. Saving Modified Persistent Attributes

If...	Then...
Persistent attributes are modified by a user and the session is updated, either directly or by the user's actions while logged in.	The modified values are not saved to the database until the session is terminated. This is because the profile attributes are cached. Caching profile attributes allows extremely fast access to them when building pages, while providing persistence between sessions.

Table 7. Saving Modified Persistent Attributes

If...	Then...
Persistent attributes are modified in an administration screen and saved, for example, by a call center agent using the Profile view under the User Profile Preferences screen.	The changes are saved to the database at that time.

GetProfileAttr and Personalization

GetProfileAttr only works for fields explicitly defined in the Personalization Profile business component. GetProfileAttr does not work with system fields, which are not explicitly defined in the business component; it returns a NULL value for them.

The Id system field is an exception. This particular system field was added in order to be available to GetProfileAttr, even though it is not in the Personalization Profile business component.

Retrieving Persistent Attributes

Values for persistent attributes are loaded in the session when a user logs in. Persistent attributes are retrieved using `GetProfileAttr()`. For more information on `GetProfileAttr()`, see Profile Functions on page 98.

About Dynamic User Profile Attributes

Dynamic attributes are created at run time and are not based on any business component fields. They are session specific.

Dynamic attributes are set and modified by actions the user takes while navigating the system, and are never written to the database. They allow real-time modification of the user experience. They are related to the state of the user, as determined when he or she entered the application.

NOTE: Do not name a dynamic profile attribute Id, as this system field generates a run time error.

This section contains the following topics:

Dynamic Attributes Set at Run Time

The login process can also automatically enter dynamic attributes. Table 8 lists the dynamic profile attributes that are set by the system at run time.

Table 8. Dynamic Attributes Set at Run Time

Dynamic Profile Attribute	Possible Values
IsStandaloneWebClient	TRUE or FALSE
Is Anonymous	TRUE or FALSE
ApplicationName	Name of the application
ActiveViewName	Name of the active view

About Retrieving Dynamic Attributes

Dynamic attributes are retrieved the same way as persistent attributes, using `GetProfileAttr()`. If the attribute is not set (or does not exist) for a given user, it will return nothing, but will not give an error message.

Dynamic profile attributes can be different for each user. When a dynamic profile attribute is set, it exists for that user only. This works because of how dynamic attributes are set and retrieved: no errors occur if an administrator references dynamic attributes that do not exist. If a dynamic profile attribute does not exist and the user sets it, a new one is created.

NOTE: The ActiveViewName profile attribute is set after the user has been navigated to the view.

Dynamic Profile Attribute Examples

The following are examples of the usage of dynamic profile attributes:

- GetProfileAttr(AttrDoesNotExist) returns NULL.
- SetProfileAttr(NewAttribute,value) creates a new dynamic profile attribute.

About Second User Profiles

The second profile is used if a call center agent wants to personalize the content shown on the screen based upon the caller's profile instead of the agent's own profile.

In Web channel interactions, only one party is involved. However, in certain other channel interactions, for example, there are multiple parties involved, for example the call center and the customer agent. In this situation, it is necessary to load a second user profile.

This section contains the following topics:

About Loading a Second User Profile

Table 9 describes instances of loading a second user profile when the customer profile is open and when the customer is not using CTI.

Table 9. Instances of Loading a Second User Profile

If...	Then...
The customer dashboard is open when the caller contacts the Call Center. **NOTE:** The second profile is loaded if the Call Center agent executes a find for the customer contact and uses the Set Dashboard button to launch the dashboard.	The second profile is automatically loaded when the dashboard is set with the user's credentials.
Customer is not using CTI. **NOTE:** LoadUserAttributes is exposed for the application object. This function takes only one argument: the row-id of the employee or contract.	The customer can write a script or business service to call the function LoadUserAttributes with the contact ID, and use this script to load the second user profile.

Accessing the Second User Profile

You can access the second user profile with personalization rules by using a you.indentifier. Table 10 lists examples of accessing the second user profile.

Table 10. Accessing the Second User Profile Examples

This user profile...	Returns...
GetProfileAttr (me.first name)	The first name of the logged in user
GetProfileAttr (you.firstname)	The first name of the second user

Using LoadUserProfile Example

The following VB example shows a method that loads a user profile into the session. The function is exposed on the Siebel Application Object.

```
Function LoadUserProfile As Integer

sArgs = "0-10N07"

TheApplication.InvokeMethod "LoadUserAttributes", sArgs

End Function
```

This function has only one parameter: the row-id of the person whose profile needs to be loaded.

If this function is called with empty parameters, it unloads the loaded user profile. For example:

```
Function LoadUserProfile As Integer

sArgs = ""

TheApplication.InvokeMethod "LoadUserAttributes", sArgs

End Function
```

Personalization Profile Business Component Restrictions

Personalization Profile business component:

- All MVG- and join-based fields (except extension table–based fields) are read-only. Flag them accordingly in Siebel Tools.
- If adding an MVG field, only its primary value is made visible to the end user.
- Do not use any inner joins in the Personalization Profile business component, as it may prevent the business component from being queried correctly.
- Do not modify the definitions of the following fields, because they are used to implement Siebel Personalization functionality:
  - Primary Account Id
  - Primary Account Id Internal

- Primary Division Id
- Primary Division Id Internal
- Primary Position Id

- Do not modify the definitions of the following joins, as they are used to implement Siebel Personalization functionality:
  - Primary Account
  - Primary Division
  - Primary Position
- For performance reasons, flag fields in Siebel Tools as read-only whenever possible. This will avoid an attempt to save the record if the corresponding attribute is changed.

For information on working with business components and tables, see *Using Siebel Tools*.

Performance Considerations

Figure 3 shows that in Siebel Personalization, a profile attribute consists of a person component and an organization component.

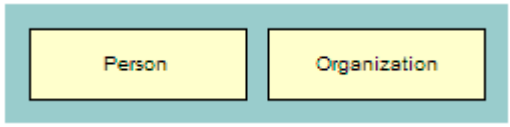


Figure 3. Personalization Profile

When you implement action sets and events associated with the Personalization profile attributes, consider the following:

- It takes two queries to load profile attributes and two queries to save them. One query is for the person, and the second query is for the organization. These attributes are loaded the first time Siebel Personalization accesses a persistent profile attribute.
- If only a *person's* persistent attributes are set, the system will not load and save the organization component of the Personalization profile and vice versa.
- Using only dynamic attributes will save two queries on startup and two on exiting, as the persistent profile would not be loaded or saved.
- If the user does not modify a persistent profile attribute during a session, then two queries are saved, because nothing is saved to the database.

Working with Multiple Value Profile Attributes

Profile attributes with multiple values are stored as MVGs (Multiple Value Group). You configure them in Siebel Tools as ordinary MVG fields on the personalization profile business component. For more information, refer to Adding New Persistent User Profile Attributes.

This section contains the following topics:

Using MVG Profile Attributes to Improve Searches

Siebel Personalization provides a hook for entering dynamic search specifications on applets and business components. Administrators can use these dynamic search specifications to target the content to the user based on their profile information. Personalization supports MVG attributes in order to extend the profile, which you can use to track information on customers that cannot be captured in single value fields. These attributes could be hobbies, interests, and so on.

About the GetProfileAttrAsList Function

The GetProfileAttrAsList function returns the MVG value as a list. You can also use this function in the EXISTS operator to create the right expressions for matching the MVG profile attributes with content within Siebel applications.

Using GetProfileAttrAsList and the EXISTS Operator

Table 11 describes using the GetProfileAttrAsList outside and within the EXISTS operator.

Table 11. Using GetProfileAttrAsList and the Exists Operator

If...	Then...
GetProfileAttrAsList is used outside the Exists operator.	It returns a comma-separated list of the MVG values. For example, if you are using a MVG called State that has the values CA, MA, GA, and CA is primary, then GetProfileAttrAsList (State) would return CA, MA, GA.

Table 11. Using GetProfileAttrAsList and the Exists Operator

If...	Then...
GetProfileAttrAsList is used within the EXISTS operator.	The function returns the value of the profile attributes in the format expected by the operator. For example, a typical usage of the EXISTS operator in this scenario is EXISTS ([Targeted States] = GetProfileAttrAsList("State")). This does a many-to-many match of the MVG Business Component Field Targeted State against the MVG profile attribute State.

GetProfileAttr and the MVG Profile Attributes

The following describes using the GetProfileAttr function and MVG profile attributes.

- Modify the semantics of GetProfileAttr also for MVG profile attributes. So if the GetProfileAttr function is called for an MVG, it returns only the primary value of the MVG.

  For example, for an MVG called State, with the values CA, MA, GA, where CA is primary, GetProfileAttr (State) would return CA.

- Use the EXISTS operator for matching content with MVG profile attribute.

  Table 12 lists the three potential cases when using EXISTS with a profile attribute.

Table 12. Three Cases of Using EXISTS with a Profile Attribute

In this case...	Expression	Description
MVG profile attribute and MVG Business component field	Expression: EXISTS ([Targeted State] = GetProfileAttrAsList("State"))	This expression tests whether any of the states in the MVG profile attribute, State, exist in the BC MVG field Targeted State.
MVG profile attribute and single value business component field	Expression: EXISTS ([State] = GetProfileAttrAsList("State"))	This expression tests whether any of the states in the MVG profile match states in the Targeted State single value BC field.
Single value profile attribute and MVG Business component field	Expression: EXISTS ([Targeted State] = GetProfileAttr("State"))	This expression tests whether single value states in the user profile match states in the MVG Targeted State BC field.

Exists Operator Usage Scenarios

Table 13 lists the usage scenarios using the EXISTS operator.

Table 13. Usage Scenarios Using the EXISTS Operator

Usage Scenario	Example	Expression
Searching for an MVG field in a business component against an MVG profile attribute.	A bookseller site wants to recommend books from all the categories that interest a user. A book can belong to multiple categories, that is, book ABC can belong to action, adventure, and history. Similarly, user XYZ is interested in multiple categories of books, that is, science fiction and action. The personalization engine matches user XYZ to all the books that belong to the categories that the user is interested in, including book ABC. This is done by matching the MVG profile field Interest with the MVG field Categories in the products business component.	Expression: EXISTS ([Categories] = GetProfileAttrAsList( ))
Searching for an MVG field in a business component against a single value profile attribute.	A product is targeted to multiple states and the marketing manager wants to recommend the product to only the people who live in one of the targeted states.	Expression: EXISTS ([Categories] = GetProfileAttrAsList("Interests"))
Searching for a single value field in a business component against an MVG profile attribute (for example, Portfolio Product Id).	A product is targeted to multiple states and the marketing manager wants to recommend the product to only the people who live in one of the targeted states.	Expression: [Product Id] = GetProfileAttrAsList (“Portfolio Product Id”)) **NOTE:** The EXISTS clause is not required when searching for a business component’s single value field against a MVG profile attribute.

4 Tracking Run-Time Events

Tracking run-time events allows the Siebel application to respond in real time to user actions. Business managers can configure the actions in response to any event. For information about tracing run-time events, see Enabling Personalization Event Logging on page 75.

This section includes the following topics:

NOTE: Where a server script is written for an application, applet, or business component, the run-time event runs first.

About Run-Time Events and Action Sets

When a run-time event occurs that is associated with one or more action sets, the personalization engine performs the specified actions. These actions modify content and user profile attributes, with the potential of triggering personalization rules. You do not need to reconfigure or recompile business objects because the interaction occurs between Siebel Personalization and the user interface.

NOTE: Upon application startup, to avoid potential inconsistency in the data available through profile attributes and the data stored in the database, it is recommended that you always force the personalization engine to load the user profile in the start event.

For example, you could watch for customers to remove products from their Siebel eSales Shopping Cart (that is, invoke the EmptyCart method of the Shopping Service business service), and then recommend a substitute product for them to consider.

Definitions of Events

Events are defined by:

- **Object Type.** The type of object to which the event occurs, such as an application, business component, or applet.

- **Object Name.** The name of the application, business component, or applet to which the event occurs.
- **Event.** The specific event that happens to the object. The set of available events is different for different object types.

- **Application event**. Table 14 lists available events.

Table 14. Application Events

Application Events	Description
Login	A user logs in to an application using the Siebel Mobile Web Client. **NOTE:** Logging in to Siebel Web Client will trigger both Login as well as WebLogin runtime events.
Logout	A user logs out from an application using the Siebel Mobile Web Client.
SetAttribute	A profile attribute is set.
ViewActivated	A view is activated.
ViewDeactivated	A view is deactivated.
WebLogin	A user logs in to the Siebel Web Client.
WebLogout	A user logs out from the Siebel Web Client.
WebSessionEnd	A Web session ends.
WebSessionStart	A Web session begins.
WebTimeout	A Web session times out. The following example was traced by setting the component parameter Application Personal Log = <FILENAME> and lists the events that occur when a Web session times out. 1 Object type: Application Object name: Siebel eCustomer Event: TIMEOUT Sub-event: 2 Object type: Application Object name: Siebel eCustomer Event: Logout Sub-event: TIMEOUT 3 Object type: Application Object name: Siebel eCustomer Event: WebSessionEnd Sub-event: 4 Object type: Application Object name: Siebel eCustomer Event: WebTimeout Sub-event:
ApplicationUnload	The ApplicationUnload event is fired on the server side when the client framework is unloaded from the client browser.

- **Business component.** All business component events are named after corresponding Siebel VB BusComp functions. Table 15 lists the available events.

Table 15.Business Component Events

Business Component Events	Is Triggered...
Associate	After an association is created when a record is added to a business component.
ChangeRecord	After the current row changes in the business component.
CopyRecord	After a new row is copied in the business component.
DeleteRecord	After a row is deleted in the business component.
InvokeMethod	After a method is called.
NewRecord	After a new row is added to the business component.
PreAssociate	Before an association is created when a record is added to a business component.
PreCopyRecord	Before a new row is copied in the business component.
PreDeleteRecord	Before a row is deleted in the business component.
PreGetFieldValue	When the value of a business component field is accessed.
PreInvokeMethod	Before a method is called.
PreNewRecord	Triggered before a new row is added to the business component.
PreQuery	Triggered before a query is executed.
PreSetFieldValue	Triggered when a value is written to the business component from the user interface.
PreWriteRecord	Triggered before a row is written to the database.
Query	Triggered after a query is executed.
SetFieldValue	Triggered after a value is written to the business component from the user interface.
WriteRecord	Triggered after a row is written to the database.
WriteRecordNew	Triggered after a new row is written to the database. Fires along with WriteRecord.
WriteRecordUpdated	Triggered after a row is updated in the database. Fires along with WriteRecord.

- **Applet.** Applet events are triggered just after a method is invoked or an applet or record is displayed, as shown in Table 16. For more information on events, see *Using Siebel Tools*.

  Table 16.Applet Events

Applet Event	Description
DisplayApplet	An applet is displayed.
DisplayRecord	A record is displayed.
InvokeMethod	A method is invoked.
PreInvokeMethod	Triggered before a method is called.

- **Subevent.** The subevent represents the following information according the event and object type, as determined by the object type (Table 17).

  Table 17.Subevent Information

When the object type is...	Then...
BusComp or Applet and the event is InvokeMethod or PreInvokeMethod	The subevent is the name of the method to monitor. The method name is found in Siebel Tools, in the Invoke Method property of the control that calls the method.
BusComp and the event is SetFieldValue or PreSetFieldValue	The subevent is the name of the field to monitor.
Application and the event is ViewDeactivated or ViewActivate	The subevent is the name of the view to monitor.
Application and the event is SetAttribute	The subevent is the name of the attribute.

- **Conditional Expression.** Adds additional control. If the conditional expression evaluates to TRUE, the action is triggered when the event occurs.

NOTE: The field referenced in the conditional expression must be activated, either by setting the field to *force active* or by displaying it on the *applet*, to trigger the run-time event.

- **Action Set.** What actions are taken when the event occurs. Action sets are defined in the Action Sets view under Administration - Runtime Events.

  Most actions set or modify user profile attributes. These set expressions store information that is later used to tailor content delivery.

- **Sequence.** The order in which the action set associated with this event in this record executes, relative to other action sets associated with this event. Runtime events with a sequence value of -1, such as a system generated workflow events for workflow processes, are executed first.

Process of Creating Action Sets

Action sets are groups of actions that are triggered by events. They are created in the Action Sets view under Administration - Runtime Events. The following steps are the tasks to create action sets:

1. Creating Action Sets.
2. Creating Actions for Action Sets.
3. Associating Events with Action Sets.

Action Types Supported

The following action types are supported for all events:

- **Attribute Set.** Simple expression that sets user profile attributes. These are: setting to a constant value, auto-increment, auto-decrement, increment by a constant, and decrement by a constant.
- **BusService.** Invokes a method of the specified business service. The input to the business service is a property set with the following properties:
  - **Context.** Business service context, as defined in the action that caused the business service to be called.
  - **ActionSet.** Action set name.
  - **Action.** Action that caused the business service to be called.
  - **EventId.** Row ID of the event that caused the action to occur.
  - **Event Name.** Event that caused the action to occur.
  - **Sub Event.** Sub Event defined in the event that caused the action to occur.
  - **Event Type.** Type of event.
  - **Object Name.** Object Name defined in the event that caused the action to occur.
  - **Business Component Name.** Used if the object type is Applet, the name of the business component on which the applet is based, as shown in Table 18.

Table 18.Business Component Name Used if Object Type is Applet or BusComp

If...	Then...
Applet	The business component name is the name of the business component on which the applet is based.
BusComp	The business component name is the same as the object name.

- **Invoke Method.** Invokes a method on the business component that caused the event to occur.

Creating Action Sets

You create action sets in the Action Sets view under Administration - Runtime Events.

To create an action set

1. Navigate to the Administration - Runtime Events screen > Action Sets view.
2. Create a new record.

   Complete the fields as needed. Some fields are described in the following table.

Field	Description
Active	**Required.** Select the check box to make the action set active.
Enable Export	Select the check box to enable exporting the action set when exporting personalization rules. Run-time events created automatically by workflow have the Enable Export flag set to false.

Creating Actions for Action Sets

You define actions to be triggered when an event occurs. Actions can modify content or user profile attributes, which in turn can modify the content displayed to the user.

Actions are created in the Action Sets view under Administration - Runtime Events.

NOTE: After modifying personalization actions, you must refresh the object manager for the actions to take effect immediately. See About Clearing and Reloading Siebel Personalization.

To create actions for an action set

1. Navigate to the Administration - Runtime Events screen > Action Sets view.
2. Select the action set, and in the Actions list, create a new record.
3. Complete the fields as needed under the More Info tab.

   Some fields are described in the following table.

Field	Comments
Name	**Required.** Name for conveniently referring to the action.
Sequence	**Required.** Number describing the order in which the action occurs. Execution begins with the action with the lowest sequence number, such as -1. Actions with the same sequence number are executed in random order. Actions occur in sequence until all actions are completed.

Field	Comments
Active	**Required.** Check to indicate whether you want the action triggered or not. Inactive actions are ignored when the event occurs. **NOTE:** This is a quick way to turn off an action without changing the start and end dates.
Conditional Expression	If the conditional expression evaluates to TRUE, the action is triggered when the event occurs. **NOTE:** The field referenced in the conditional expression must be activated, either by setting the field to *force active* or by displaying it on the *applet*, to trigger the run-time event.
Action Type	**Required.** Select the type of action from the drop-down menu: ■ **Attribute Set.** Sets a user profile attribute. The most common action type. For example, `ProdName` is set to a product in the user's Shopping Cart. Attribute Set uses the Profile Attribute, Set Operator, Value, Set Minimum, and Set Maximum fields. For example, to create an action in which you specify the profile attribute service: ■ Select Attribute Type from the Action Type field. ■ Enter `AttributeFooValue` in the Profile Attribute field. ■ Enter `Set` in the Value field. ■ **BusService.** Invokes a business service method. You can also use this action type to trigger a workflow process or to get profile attributes. The input argument is a property set. BusService uses the Business Service Name, Business Service Method, and Business Service Context fields. ■ **InvokeMethod.** Invokes a method on the object (applet, business component, or application) that caused the event to occur. InvokeMethod uses the Method Name and Method Argument fields.
Profile Attribute	The user profile attribute to set.

Field	Comments
Set Operator	One of five choices: ■ **Set.** Sets a user attribute to an explicit value. For example, Set DidLogin to Yes. ■ **Increment.** Increments a user attribute by an integer (default value, if Set Expression is blank, is 1). For example, Increment PagesView by 1. ■ **Decrement.** Decrements a user attribute by an integer (default value, if Set Expression is blank, is 1). For example, Decrement PagesView by 10. ■ **Add List.** Adds a value to a comma-delimited list. For example, add the name of the current product to the list of products viewed. ■ **Remove List.** Removes a value from a comma-delimited list.
Value	The expression to use with the set operator. You can use any expression that evaluates to something the Set Operator can use, usually a business component field. Use the Personalization Business Rules Designer to pick the business component field. Set, Add List, and Remove List must evaluate to a single value. Increment and Decrement must evaluate to an integer.

Business Service Context	Parameter to pass to the business service method. While a business service can take many name/value pair parameters, you can only pass one parameter—called Context—in Siebel Personalization.
	An example of a business service context is:
	`Action Type: BusService`
	`Business Service Name: Workflow Process Manager`
	`Business Service Method: RunProcess`
	`Business Service Context: "ProcessName", "My Workflow".`
	Use the following eScript command to retrieve the value from the business service context from a custom build business service:
	`var sValue = Inputs.GetProperty("Context")`
	Expressions like Today() or [<Field Name>] are not calculated. Only the plain string is sent to the business service and the string must be interpreted within the business service. In business services, like Workflow Process Manager, the name value pairs are used as input arguments. However, in a custom business service you must develop this functionality.

Field	Comments
Method Name	Method to invoke on the business component, if the conditional expression evaluates to TRUE and the action type is Invoke Method.
Method Argument	Parameter to pass the business component. You can only pass one argument.

Associating Events with Action Sets

You can set up events to be monitored by the system and to trigger actions when these events occur. The actions then modify content or user profile attributes or invoke business service methods, which can cause personalization rules to fire.

To associate an event with an action set

1. Navigate to the Administration - Runtime Events screen > Events view.
2. Create a new record.

   Complete the fields as needed. Some fields are described in the following table.

Field	Description
Sequence	**Required.** An event can trigger multiple action sets. Enter numbers in this field to control when the action set associated with this event in this record executes, relative to other action sets associated with this event.
Object Name	The name of the application, business component, or applet (depending on the object type) to which the event occurs.
Event	**Required.** Select from the drop-down list. The choices depend on which object type you choose.

Field	Description
Subevent	The subevent represents the following information according the event and object type: When the object type is: ■ BusComp or Applet and the event is InvokeMethod or PreInvokeMethod, the subevent is the name of the method to monitor. The method name is found in Siebel Tools, in the Invoke Method property of the control that calls the method. ■ BusComp and the event is SetFieldValue or PreSetFieldValue, the subevent is the name of the field to monitor. ■ Application and the event is ViewDeactivated or ViewActivated, the subevent is the name of the view to monitor. ■ Application and the event is SetAttribute, the subevent is the name of the attribute.
Conditional Expression	Enter a conditional expression if necessary to add additional control to whether the event is monitored. **NOTE:** The field referenced in the conditional expression must be activated, either by setting the field to *force active* or by displaying it on the *applet*, to trigger the run-time event.

Creating Event Aliases

An event alias is a template that you can use to create commonly occurring events. Event aliases are created in the Event Aliases list under Administration - Runtime Events.

Event aliases consist of a meaningful name and the minimum set of objects that define an event: Object Type, Object Name, Event, and Subevent. When you pick an event alias in the Name field of the Events form, the fields for the objects that define the event are automatically completed.

To create an event alias

1. Navigate to the Administration - Runtime Events screen > Event Aliases view.
2. Create a new record.

   Complete the fields as needed. Some fields are described in the following table.

Field	Description
Object Type	**Required.** Select from the drop-down list: ■ Application ■ BusComp ■ Applet
Object Name	The name of the object to which the event occurs.
Event	**Required.** Select from the drop-down list. The choices depend on which object type you choose.
Subevent	Fill in Subevent only when the object type is BusComp or Applet and the event is InvokeMethod. In this case, the subevent is the name of the method to be monitored.

5 Setting View Visibility

This section describes the process of setting the visibility of views by finding the name of the view, setting the visibility of applets by finding the name of the applet, setting its visibility, and setting the number of rows displayed.

This section includes the following topics:

About Setting View Visibility

This section includes the following topics:

Events Triggering Visibility Flowchart

The sequence of events triggered when a user accesses a view, which determines the visibility of an applet, is shown in Figure 4.

The evaluation of visibility conditions for views and applets is done dynamically. For example, the Shopping Cart applet in Siebel eSales can be hidden initially, and then shown based on user actions in real time.

For more information on conditional expressions, see Conditional Expressions.

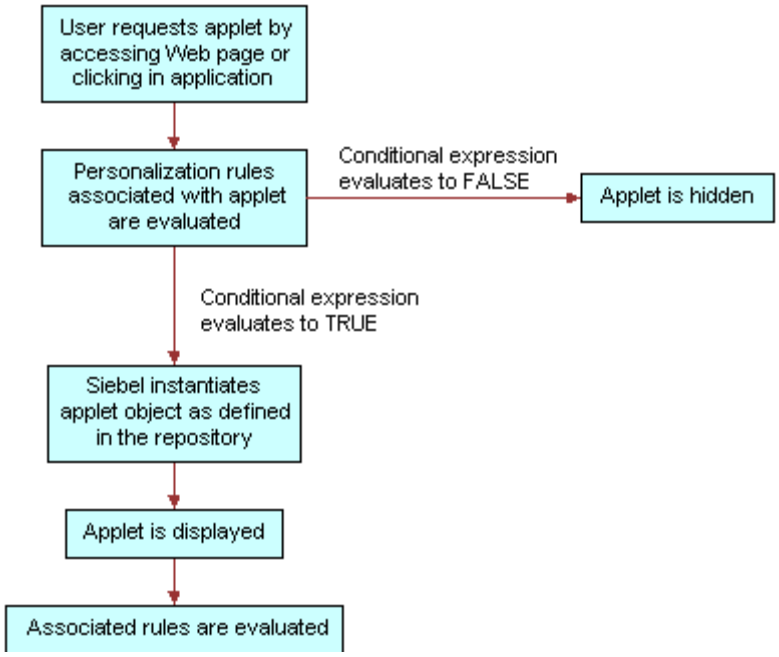


Figure 4. Sequence of Events Determining Applet Visibility

About Writing Visibility Rules

You can only use dynamic and persistent profile attributes to write visibility rules. However, you can create dynamic profile attributes that drive the visibility of views and applets in response to events that are based on other data, such as transactions and external data.

About View Visibility

View visibility is also affected by responsibility. You can only use Siebel Personalization to control the visibility of views and applets that are accessible to a user. For example, you cannot write personalization rules to show administration views to users without administrator responsibility.

For more information on responsibilities, see *Siebel Applications Administration Guide*.

Importance of the Repository

When working with views or applets personalization, it is important that you have the most up-to-date repository. The reason is that the views and applets available to you are derived from the repository of the application in which you are working, which may not be the same as the server repository.

Process of Setting View Visibility

You control which views users see in the Views list under Administration - Personalization. For example, if you wanted to hide certain views from channel partners, use the Views list. View visibility is controlled by the conditional expression: if the conditional expression evaluates to TRUE, then the view is shown.

To set up view visibility, perform the following tasks:

- Finding the Name of a View
- Setting the Visibility of a View on page 48

Finding the Name of a View

See also Process of Setting View Visibility and Setting the Visibility of a View.

Before you can set the visibility of a view, you must know its name. Table 19 describes how to find the name of the view depending on the application.

Table 19. Finding a View Based on the Application

If it is...	Then...
employee application	Navigate to the view and choose Help > About View. This displays information about the view, its applets, and its business components.
a customer application	Navigate to the view, right-click, and then choose View Source. In the <script> section of the source file, the name of the view is shown after SWEView: in each instance of action. This method also works for employee applications.
Siebel Tools	Choose the Screen object, query for the screen containing the view whose name you want to know, and then select the Screen View object. This displays the views in the Name field. This method works for all applications.

Setting the Visibility of a View

This procedure creates a conditional expression that determines whether the view is shown. View visibility is set in the Views list under Administration - Personalization.

To set the visibility of a view

1. Navigate to the Administration - Personalization screen > Views view.
2. Query for the desired view.
3. If that view is not available, create a new record.
4. In the Name field, query for the desired view.

   Complete the fields as needed. Some fields are described in the following table.

Field	Description
Start Date	The date after which the conditional expression is evaluated. If Start Date is blank, the conditional expression is evaluated continuously or until the End Date is reached.
End Date	The date after which the conditional expression is not evaluated. The conditional expression is: ■ Evaluated continuously after the Start Date if the End Date is blank. ■ Always evaluated if both Start and End Date are blank.
Conditional Expression	Enter the expression as text or click the icon to use the Personalization Business Rules Designer. If the expression evaluates to: ■ TRUE, then the view displays when the screen is reloaded. ■ FALSE, then the view does not display. Because of the way that views are cached, conditional expressions should not be used to dynamically display views. When data changes the result of a conditional expression, the user interface does not automatically reload to show a view. A view that is hidden will not display dynamically once the conditional expression evaluates to true. The view will need to be manually reloaded before the hidden view will appear. For information on conditional expressions, see Conditional Expressions on page 58.

Process of Setting Applet Visibility

You control whether or not an applet is displayed to a user and what content is shown in the Applets view under Administration - Personalization. For example, you can hide the Shopping Cart in Siebel eSales when it is empty. Like view visibility, applet visibility is controlled by the conditional expression, for example the conditional expression for Salutation Applet (eMarketing).

To set up applet visibility, perform the following tasks:

- Finding the Name of an Applet on page 49.
- Setting the Visibility of an Applet on page 49.

For more information on applet visibility, see Hiding an Applet Based on a Field Value and Setting the Number of Rows Displayed in an Applet.

Finding the Name of an Applet

See also Process of Setting Applet Visibility and Hiding an Applet Based on a Field Value, and Setting the Number of Rows Displayed in an Applet.

Before you can set the visibility of an applet, you must know its name.

To find the name of an applet

1. Find the name of the view in which the applet appears (see Finding the Name of a View on page 47).
2. In Siebel Tools, select the View object.
3. Query for the name of the view in which the applet appears.
4. Expand the View object, and then select View Web Template.
5. Expand the View Web Template object, and then select View Web Template Item.

   The applets are listed under View Web Template Items.

Setting the Visibility of an Applet

See also Process of Setting Applet Visibility, Hiding an Applet Based on a Field Value, and Setting the Number of Rows Displayed in an Applet.

This procedure creates a conditional expression that determines whether the applet is shown. Applet visibility is set in the Applets view under Administration - Personalization.

To set the visibility of an applet

1. Navigate to the Administration - Personalization screen > Applets view.
2. Query for the desired applet.

3 If that applet is not available, create a new record.

4 In the Name field, query for the desired applet.

Complete the fields as needed. Some fields are described in the following table.

Field	Description
Start Date	The date after which the conditional expression is evaluated. If Start Date is blank, the conditional expression is evaluated continuously or until the End Date is reached.
End Date	The date after which the conditional expression is not evaluated. The conditional expression is: ■ Evaluated continuously after the Start Date if End Date is blank. ■ Always evaluated if both Start and End Date are blank.
Conditional Expression	Enter the expression as text or click the button to use the Expression Editor. If the expression evaluates to: ■ TRUE, then the applet displays. ■ FALSE, then the applet does not display. ■ Blank, then it is equivalent to an expression that is always TRUE. For example: ■ Hide or display an applet based on a user's responsibility, by entering GetProfileAttr("Primary Responsibility Name") = "Admin" ■ Use the OR conditional expression for an either or condition, by entering GetProfileAttr('Me.Is Anonymous') <> 'FALSE' OR GetProfileAttr('Login Name') = 'GUESTCP' Where <> means Not equal to FALSE For information on conditional expressions, see Conditional Expressions on page 58.

Hiding an Applet Based on a Field Value

Hiding an applet is set in the Administration - Runtime Events.

See also Process of Setting Applet Visibility and Setting the Number of Rows Displayed in an Applet.

To hide an applet based on a field value

1. Navigate to the Administration - Runtime Events screen > Action Sets view.
2. Create a new record and complete the following fields.
   - Name
   - Start Date
   - End Date
   - Action check box
3. Scroll down to the Actions form and create a record.

   Complete the following fields.

Field	Description
Name	Name for the event. For example, SetProfileAttr.
Sequence	An event can trigger multiple action sets. For example, 1. Enter numbers in this field to control when the action set associated with this event in this record executes, relative to other action sets associated with this event.
Active	Select the box. This indicates that the action is triggered. Inactive actions are ignored when the event occurs. **NOTE:** This is a quick way to turn off an action without changing the start and end dates.
Action Type	Select Attribute Set from the drop-down menu.
Set Operator	Select Set from the drop-down menu.
Value	Select the <product name>.
Profile Attribute	The user profile attribute to set, for example Product Name Profile.

4. From the Show drop-down list, select Events.

5 Create a new record and complete the following fields.

Field	Description
Sequence	Enter a number for the order in which the action occurs. For example, 1. Execution begins with the action with the lowest sequence number. Actions with the same sequence number are executed in random order. Actions occur in sequence until all actions are completed.
Object Type	Select Applet from the drop-down list.
Object Name	Name of the applet. For example, Contact Asset Mgmt- Asset List Applet.
Event	Select InvokeMethod from the drop-down list. The choices depend on which object type you choose.
Subevent	Enter Drilldown as the subevent.
Action Set Name	Select an action set to run when the event occurs. For example, SetProductName.

6 Save the record.

Setting the Number of Rows Displayed in an Applet

The number of rows displayed in applets is set in the [SWE] section of the configuration (.cfg) file for the application.

For example:

```
[SWE]

NumberOfListRows = 7
```

Changing the Number of Rows Displayed in an Applet

Change the value of the `NumberOfListRows` parameter in the `[SWE]` section of the application configuration file to the number of rows you want to see displayed.

See also Process of Setting Applet Visibility.

6 Targeting Content by Using Expressions

This section describes how to target content by using expressions. It also describes how to create rule sets and associate rule sets with applets. Salutation applets are also described because they use inclusion expressions of rules to display a text message to the user.

This section includes the following topics:

Process of Content Targeting

Table 20 lists the four steps to content targeting.

Table 20. Content Targeting Steps

Step	Description
Content tagging	Business managers tag content with attributes that describe for whom the content is most appropriate. For example, a business manager might tag a company's products by industry focus or value classification.
User profiling	Users enter information about themselves for use in evaluating business rules. Each piece of information is called a profile attribute. For example, a profile attribute could keep track of the industries in which a customer is interested. For more information on profile attributes, see Chapter 3, Managing User Profiles.

Table 20. Content Targeting Steps

Step	Description
Business rules administration	Business managers create rules that govern which content is shown to which users. These rules are written using an if-then syntax and allow business managers to make changes to the business logic during run time. You can write rules to examine tagged content or the actions of users in the system.
Association with applets	Business rules are associated with Siebel applets at run time, and only the content matching the rules is shown to the appropriate user.

Actions to Control the Content

Controlling the content shown to users involves the following actions:

- Creating rule sets to control the content shown in applets.
- Creating the individual rules needed to accomplish the objectives of the rule sets.
- Associating the rule sets with applets and setting the evaluation sequence. Remember that rule sets can be shared by multiple applets.
- Creating a conditional expression for each rule set associated with an applet that prevents the rule set from being evaluated whenever there is no content available.

  The rule set conditional expression can also create complex Boolean logic within the rule set. The values of profile attributes can turn rules on and off.

Rule Sets and Rules Flow Chart

You create both rule sets and rules in the Rule Sets view. Figure 5 shows the relationship of rules and rule sets to applets and views.

This section contains the following topics:

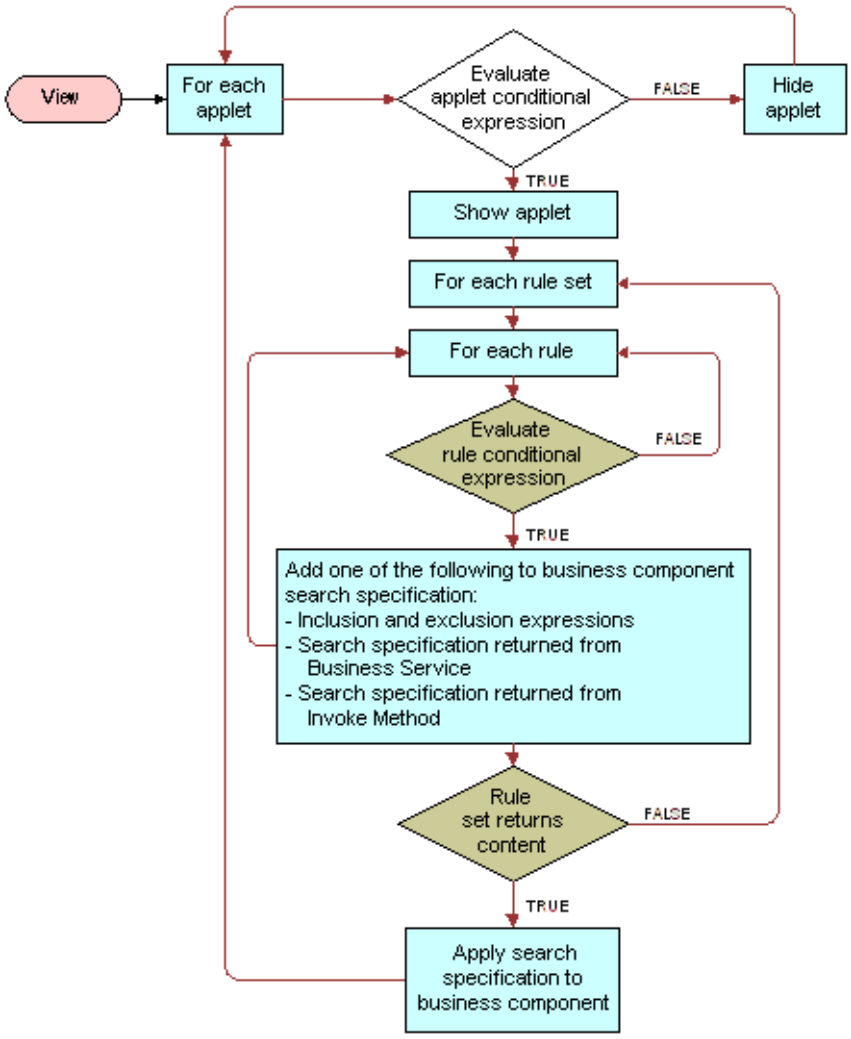


Figure 5. Relationship of Rules and Rule Sets to Applets and Views

Evaluating Rule Sets

Table 21 lists the sequence in which rule sets are evaluated.

Table 21. Sequence of Evaluating Rule Sets

If...	Then...
A rule set is evaluated and no content or records are returned.	The next rule set is evaluated and ■ Processing continues until a rule set returns content. ■ As soon as a rule set returns content, processing stops. ■ Subsequent rule sets in the sequence are not evaluated.
The conditional expression associated with the rule set evaluates to FALSE.	

Creating Complex Evaluation Flow

You can also use conditional expressions to set up Boolean logic (IF, AND, OR) among the rule sets associated with a view or an applet to create a more complex flow of evaluation.

The results of the evaluation of all the rules in the set are returned to the underlying business component associated with the applet in the form of a search expression. This search expression controls the content displayed in the applet.

Rule Sets Best Practices

Limiting the number of rule sets is good practice. This is because each rule set is a query built by adding all the rules together. The rules contribute to the final query, but do not cause a query by themselves. Therefore, you can create as many rules as you like, but create rules sets carefully.

The sequence you set for rule set evaluation is important. If the system must evaluate many rule sets before getting one that returns content, performance can be affected noticeably.

Managing Rule Set Overhead

One way of managing the overhead on executing rule sets is to add a conditional expression to the rule set that evaluates to FALSE when the rule set is unlikely to return any records. This action skips the rule set whenever it is unlikely to find content.

For example, as shown in Table 22, rule set 1 is based on the user's age as calculated from his or her birthday and rule sets 2 and 3 are based on other information.

Table 22. Example of Managing Rule Sets

If...	Then...
The user has not entered his or her birthday.	His or her age cannot be calculated.
You put a conditional expression on rule set 1 that evaluates to FALSE if the birthday attribute is blank.	The rule set is not evaluated.

About Expressions and Expression Types

Expressions set the basic parameters for controlling the content to users. You can create expressions by entering the expression as text in the appropriate field or by using the Personalization Business Rules Designer.

The types of expressions are:

- Conditional Expressions on page 58
- Search Expressions on page 58

About Building Expressions

Expressions are written in Siebel Query Language entered as text. You build expressions from the following basic elements:

- Functions.
- Operators
- Profile attributes
- Business component fields

For more information on these elements, see Personalization Business Rules Designer Contents on page 60.

Creating Expressions

To create expressions you can either:

- Enter the expression as text directly in the expression field, or
- Click the icon in any selected expression field to launch the Personalization Business Rules Designer.

The maximum size for any expression (conditional, include, exclude, and business service context) is 4096, which is limited by the database configuration.

For more information on:

- Creating expressions using the Personalization Business Rules Designer, see About the Personalization Business Rules Designer on page 59.
- Siebel Query Language, see Operators for Building Condition Expressions on page 83 which details the formulas and operators available.

Conditional Expressions

See also About Creating Rule Sets and Rules.

Conditional expressions control whether an applet or view is displayed. What content is displayed is controlled through rule sets based on content and user profile attributes. You must write conditional expressions to evaluate to TRUE or FALSE. Conditional expressions trigger rules when they evaluate to TRUE, the view or applet is displayed or the rule set and rules are processed.

Example:

```
GetProfileAttrAsInt(“Number of Visits”) >= 7
```

Interpretation of Numbers in Expressions

Enclose numbers, such as telephone numbers, in quotation marks, for example:`‘650-477-5000’`. This prevents numbers from being interpreted as numeric values.

Search Expressions

Search expressions consist of inclusion and exclusion expressions. Inclusion and exclusion expressions relate content attributes to user profile attributes in order to control what content each user sees. Each expression is part of a rule that is part of a rule set that you can link to an applet for the purpose of controlling content.

Business components have existing visibility settings, for example, My visibility, All visibility, or Sales Rep visibility. There may be other search specifications configured in Siebel Tools. When a rule is evaluated at run time, its inclusion and exclusion expression are added to the business component as additional search specifications.

A rule can only return content that is within the scope of the business component’s inherent visibility or search specifications as set in Siebel Tools.

Example:

```
EXISTS([Related Product] = GetProfileAttr(“ProductInCart”))
```

shows products that are related to products in the Siebel eSales Shopping Cart.

About the Personalization Business Rules Designer

The Personalization Business Rules Designer allows business managers to implement personalization rules without learning complex programming languages.

This section contains the following topics:

Displaying the Personalization Business Rules Designer

See also Personalization Business Rules Designer Contents, Process of Creating Business Rules, and Writing a Business Rule.

The Personalization Business Rules Designer (Figure 6) appears when you click the icon in any selected expression field.

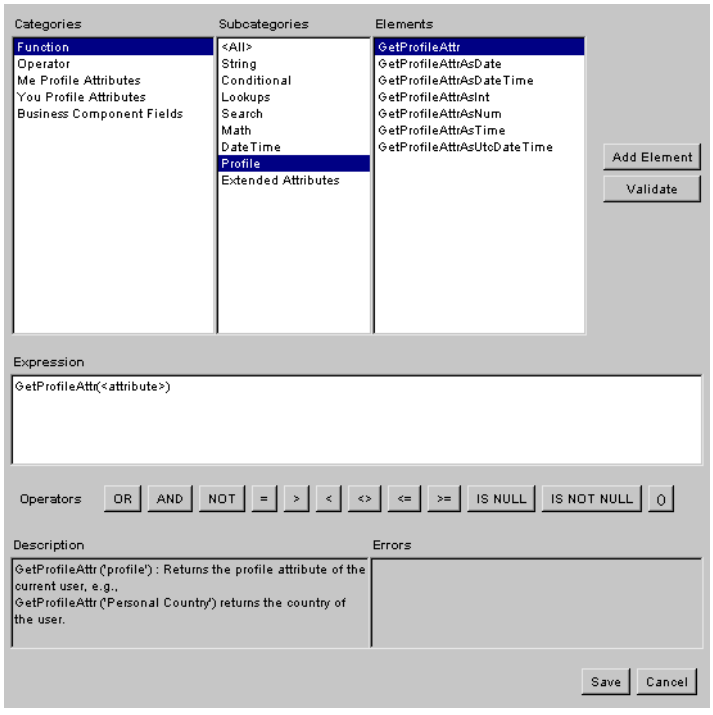


Figure 6. Personalization Business Rules Designer

Personalization Business Rules Designer Contents

The contents of the Personalization Business Rules Designers are:

- **Categories.** Contains the main categories available for building business rules:
  - **Function.** Siebel Query Language functions.

    For more information, see Functions on page 87.
  - **Operator.** Siebel Query Language operators.

    For more information, see Operators for Building Condition Expressions on page 83.
  - **Me Profile Attributes.** The profile attributes of the primary user, such as a Siebel eSales customer or Siebel Call Center agent.

    Table 23 lists valid subcategories.

Table 23. Me Profile Attribute Subcategories

Subcategory	Description
ShowAsString	GetProfileAttr is used in the expression.
ShowAsNumber	GetProfileAttrAsNum is used in the expression.
ShowAsInteger	GetProfileAttrAsInt is used in the expression.
ShowAsDate	GetProfileAttrAsDate is used in the expression.
ShowAsTime	GetProfileAttrAsTime is used in the expression.
ShowAsDateTime	GetProfileAttrAsDateTime is used in the expression.

For more information on GetProfileAttr functions, see Profile Functions on page 98.

- **You Profile Attributes.** The profile attributes of the secondary user, such as a customer calling a call center agent. Uses the same subcategories as Me Profile Attributes.
- **Business Component Fields.** These depend on the context of the rule (Table 24), that is, where the Rules Designer is invoked. This category will or will not have data in its elements, depending on the following contexts.

Table 24. Context of Business Component Fields

Context	Data Shown
View	None
Applet	Fields in the business component on which the applet is based
Rule set not associated with an applet	None
Rule set associated with one applet	Fields in the business component on which the applet is based

Table 24. Context of Business Component Fields

Context	Data Shown
Rule set associated with more than one applet, and the applets are based on the same business component	Fields in the business component on which the applets are based
Rule set associated with more than one applet, and the applets are based on different business components	None
Applet event	Fields in the business component on which the applet is based
Application event	None
Business component event	Business component fields
Action set not associated with an applet event or business component event	None
Action set associated with one applet event	Fields in the business component on which the applet is based
Action set associated with one business component event	Business component fields
Action set associated with more than one applet event or business component event	None

- **Subcategories.** Contains subcategories of the selected category. Its context is controlled by the Categories box and it controls the context of the Elements box.
- **Elements.** Contains all of the elements used to build business rules. Its context is controlled by the Categories and Subcategories boxes.
- **Expression.** Displays the business rule. You can type and edit in the box, as well as add elements to it.
- **Quick bar (Operators).** Provides buttons for frequently used operators.
- **Description.** Gives a description of the element selected in the Elements box.
- **Errors.** Displays errors in expressions or highlighted parts of expressions when you click Validate.

Process of Creating Business Rules

See also Displaying the Personalization Business Rules Designer, Personalization Business Rules Designer Contents, and Writing a Business Rule.

Use the following process to build your business rule using the Personalization Business Rules Designer:

1. Select an element, and then click Add Element.

   NOTE: Double-clicking an element has the same effect as Add Element. You can also type text in the Expressions box.

   The Rules Designer automatically provides the correct syntax for elements. For example, adding the Left string function displays `Left(<string>,<n>)` in the Expressions box.

2. Fill in arguments in the element, and then enclose it in parentheses if necessary.
3. Add other elements and complete them.
4. Connect elements using the Quick bar or by typing.
5. Click Validate to check your expression for errors.

   If part of an expression is highlighted, only the highlighted string is evaluated. Errors are displayed in the Errors box.

6. Save the business rule and return to the expression field.

   This also validates the entire rule.

If...	Then...
The rule is valid.	The rule is saved and the Rules Designer is closed.
The rule has any errors.	They are displayed and the Rules Designer is not closed.

Writing a Business Rule

See also Displaying the Personalization Business Rules Designer, Personalization Business Rules Designer Contents, and Process of Creating Business Rules.

This section shows how to write a business rule using the Personalization Business Rules Designer.

Writing Rule Example

The following example shows how to write this rule:

```
EXISTS([Related Product] = GetProfileAttr(“ProductInCart”))
```

which shows products that are related to products in the Siebel eSales Shopping Cart.

Writing a Personalization Rule

This procedure describes how to write a personalization rule.

To write a personalization rule

1. Click the icon in a selected expression field.
2. Choose Operator from Categories, Pattern Matching from Subcategories, and EXISTS from Elements, and then click Add Element.

   The Expression box displays EXISTS(<condition>).
3. Click <condition> to highlight it.
4. Choose Business Component Fields from Categories, <All> from Subcategories, and Related Product from Elements, and then click Add Element.

   The Expression box displays EXISTS([Related Product]).
5. Place the cursor after [Related Product] in the Expression box, then click = in the Quick bar.

   The Expression box displays EXISTS([Related Product]=).
6. Place the cursor after =.
7. Choose Function from Categories, Profile from Subcategories, GetProfileAttr from Elements, and then click Add Element.

   The Expression box displays:

   ```
   EXISTS([Related Product]= GetProfileAttr(<attribute>))
   ```
8. Click <attribute> to highlight it, and then type “ProductInCart”.

   The Expression box displays the completed rule:

   ```
   EXISTS([Related Product] = GetProfileAttr(“ProductInCart”))
   ```
9. Save the rule to validate it and put it in the expression field.

About Creating Rule Sets and Rules

Rule sets control the display of content. You can associate multiple rule sets with individual applets, and you can associate an individual rule with multiple applets. Each rule set can contain multiple rules. You create rule sets in the Rule Sets view under Administration - Personalization.

After you have created a rule set, you need to create the individual rules that make up the set and accomplish the objective for matching content to the user.

This section consists of the following topics:

Creating a Rule Set

This procedure describes the process of creating a rule set.

To create a rule set

1. Navigate to the Administration - Personalization screen > Rule Sets view.
2. Create a new record.

NOTE: After modifying personalization rules, you must refresh the object manager for the rules to take effect immediately. See About Clearing and Reloading Siebel Personalization.

Creating a New Rule

This procedure describes the process of creating a new rule.

To create a new rule

1. Navigate to the Administration - Personalization screen > Rule Sets view.
2. Select a rule set.
3. In the Rules list, create a new record.

   Complete the fields as needed. Some fields are described in the following table.

Field	Description
Sequence	**Required.** Enter numbers in this field to set the rules evaluation order. Evaluation begins with the rule with the lowest sequence number and continues until all rules are evaluated, except that -1 is evaluated last. Rules with the same sequence number are evaluated in random order. The sequence number for rules does not have the same significance as for rule sets, because all the rules in a rule set are evaluated whether content is returned or not. Rule sequence is very important, however, for the order of the text strings displayed in salutation applets.

Field	Description
Rule Type	Select a rule type from the drop-down menu: ■ **Expressions.** Evaluates inclusion and exclusion expressions directly. ■ **BusService.** Invokes a business service and expects a property set in return, which is passed on to the business component. The input argument to Business Service is a property set with four properties: Context, BusComp, RuleSet, and Rule. ■ **Invoke Method.** Invokes a method on the business component to get a string, which is the search specification. The input argument to Invoke Method is a string.
Active	Check the box to use the rule.
Conditional Expression	**Optional.** Use the conditional expression to control the evaluation of the rule, or to set up Boolean logic for evaluation of all the rules in the rule set.
Include Expression	Use with the Expressions rule type. An expression that sets parameters to include content.
Exclude Expression	Use with the Expressions rule type. An expression that sets parameters to exclude content.
Business Service Name	Name of the business service to invoke, if the conditional expression evaluates to TRUE and the rule type is BusService.
Business Service Method	Method to invoke on the business service.
Business Service Context	Parameter to pass to the business service method. While a business service can take many name/value pair parameters, you can only pass one parameter—called Context—in Siebel Personalization. An example of a business service context is `UserType=Partner;AccountState=Gold`
Method Name	Method to invoke on the business component, if the conditional expression associated with the rule evaluates to TRUE and the rule type is Invoke Method.
Method Argument	Parameter to pass the business component. You can only pass one argument.

Associating Rule Sets With Applets

This procedure describes how to use the Rule Sets list to associate existing rule sets with applets.

For information on creating rule sets, see About Creating Rule Sets and Rules.

To associate rule sets with an applet

1. Navigate to the Administration - Personalization screen > Applets view.
2. Select an applet.
3. Scroll down to the Rules Sets list, and create a new record.
4. In the Name field, select a rule set.

   Complete the fields as needed. Some fields are described in the following table.

Field	Description
Sequence	Choose the order in which to evaluate the rule sets.
Start Date	The date after which the conditional expression is evaluated. If Start Date is blank, the conditional expression is evaluated continuously or until the End Date is reached.
End Date	The date after which the conditional expression is not evaluated. If End Date is blank, the conditional expression is evaluated continuously after the Start Date. If both Start and End Date are blank, the conditional expression is always evaluated.
Conditional Expression	Processes the rule set if the expression evaluates to TRUE.

About Salutation Applets

See also Hyperlinking Salutation Messages.

The salutation applet is a specialized applet that uses search specifications, defined in the inclusion expressions of rules, as message text displayed to the user. The salutation applet allows business managers to write customized messages using free-form HTML.

Siebel applications use the salutation applet to display information about the user or the session. You can set the parameters of this applet to display a variety of information taken from the user's profile or actions performed during the session.

This applet can greet the user by name, indicate how long it has been since the user last visited the site, and present the user with information about specific products or services that match known interests or previous behavior.

Hyperlinking Salutation Messages

See also About Salutation Applets.

Use the Language() function to localize salutation messages. The salutation messages can include hyperlinks to other screens, views, and applet drilldown objects.

This section includes the following topics:

- Hyperlinking Salutation Messages to Screens
- Hyperlinking Salutation Messages to Views
- Hyperlinking to Siebel Employee Relationship Management Views
- Hyperlinking Salutation Messages to Applets
- Two Salutation Applets in One View

Hyperlinking Salutation Messages to Screens

Use the command SWEPersonalizationGotoScreen(‘ScreenName’); in the hyperlink tag.

Enclose the name of the screen in single quotes (‘), and represent spaces in the name by plus signs (+).

The following is an example of the command syntax:

```
<A href=JavaScript:SWEPersonalizationGotoScreen('Accounts+
Screen');>
```

Hyperlinking Salutation Messages to Views

Use the command SWEPersonalizationGotoview(‘ViewName’); in the hyperlink tag.

Enclose the name of the view in single quotes (‘), and represent spaces in the name by plus signs (+).

The following is an example of the command syntax:

```
<A href=JavaScript:SWEPersonalizationGotoview('User+Profile
+View+(eApps)');>
```

Hyperlinking to Siebel Employee Relationship Management Views

You can also use this command to hyperlink to Siebel Employee Relationship Management views.

The syntax is the following:

```
<A href=JavaScript:SWEPersonalizationGotoview('My+View
+Name');>
```

or

```
<A href=JavaScript:SWEPersonalizationGotoview('My+View
+Name','&SWEmyExtra1=true&SWEMyExtra2=false');>
```

Make sure there are no spaces:

- Between the two argument quotes and the comma separating them.
- In the myExtra parameters.

The ampersand for the first myExtra parameter is optional. You can also use the following syntax:

```
<A href=JavaScript:SWEPersonalizationGotoview('My+View
+Name','SWEmyExtra1=true&SWEMyExtra2=false');>
```

There is no dup argument checking mechanism. If your myExtra parameters contain an argument that SWE has already generated, undesired behavior might occur.

Hyperlinking Salutation Messages to Applets

Use the command `SWEPersonalizationDrillDown ('`*`ViewName`*`','`*`AppletName`*`','`*`FieldName`*`','`*`RowId`*`',{`*`ParentRowIds`*`});` in the hyperlink tag where:

- *`ViewName`* is the name of the view where the drilldown object is defined.
- *`AppletName`* is the name of the applet on which the drilldown object is defined.
- *`FieldName`* is the business component field name on which the drilldown object exists.
- *`ParentRowIds`* is an array of strings that contains the Row IDs of the parent records. This array is optional, and is only used when drilling down in a child applet.

Enclose names in single quotes (‘), and represent spaces in the names by plus signs (+).

Command Syntax Example

The following is an example of the command syntax:

```
<A href=JavaScript:SWEPersonalizationDrillDown('Opportunity+
List+View','Opportunity+List+Applet','Name','1-45XHZ');
```

Two Salutation Applets in One View

See also About Salutation Applets and Hyperlinking Salutation Messages.

To have two salutation applets in the same view, do the following:

1. Copy the Salutation (eApps) business component and rename it.
2. Add the copy to the business object associated with the view.
3. Create a new applet in the view based on the copy.

Process of Adding a Message to the Salutation Applet

See also About Salutation Applets and Hyperlinking Salutation Messages.

The following is the process of adding a message to the salutation applet:

1. Select the salutation applet.
2. Select the rule set associated with the salutation applet.
3. Add a rule to the rule set.
4. Provide a name, sequence, conditional expression, and inclusion expression (message) for the rule.

NOTE: For salutation applets (based on the eApps business component), if HTML tags are used in Personalization rules, the personalization engine passes them on to the browser. This allows users to define formatted messages. The eApps business component also supports the LongDate method for getting the current date and time.

Modifying the Siebel eService Salutation Applet

See also About Salutation Applets, Hyperlinking Salutation Messages and Process of Adding a Message to the Salutation Applet.

Use the following procedure to add a message to the Siebel eService salutation applet that will request the user to register his or her product.

This section includes the following topics:

Adding a Message to the eService Salutation Applet

This procedure describes how to modify the eService Salutation applet.

To add a message to the eService Salutation applet

1. Navigate to the Administration - Personalization screen > Applets view.
2. Select Salutation Applet (eService).
3. Click the name of the rule set associated with Salutation Applet (eService).
4. Scroll down to the rule list to the Message Self-Service rule, and add a new rule to the list.
5. Fill in the following fields under the More Info tab below the Rules list.

   The rule set is saved and the new rule is made active. Reloading Siebel Personalization is not necessary.

Field	Value
Name	Message Register.
Sequence	13. This evaluates the rule after the Message Self-Service rule.
Rule Type	Expressions.
Active	Select the check box.
Conditional Expression	((GetProfileAttr(“Me.Is Anonymous”) IS NULL) OR (GetProfileAttr(“Me.Is Anonymous”) = 'FALSE')) AND (GetProfileAttr (“Full Name”) IS NOT NULL) This shows the expression only to registered users.
Description	Register your products. This field is optional.
Include Expression	 <img src='images/bullet_green.gif' align='top' > <b> Please register your products. This free-form HTML text displays a bullet with the message in bold.

Viewing the Results of Modifying the eSalutation Applet

Log in to eService as SADMIN. Figure 7 is an example of the Siebel eService Salutation Applet after adding a new personalization rule.

Figure 7. Siebel eService Salutation Applet After Adding a New Personalization Rule

7 Testing Personalization Rules

This section explains the ways in which you test your personalization rules and includes the following topics:

- Ways to Test Personalization Rules
- About Test Mode
- About Using the Log File
- About Exporting and Importing Personalization Data
- About Clearing and Reloading Siebel Personalization

Ways to Test Personalization Rules

Testing your personalization rules is important for evaluating their effectiveness and correctness. Table 25 lists the features that allow you to test your personalization rules in a development environment before they are deployed in a production environment.

Table 25. Features for Testing Personalization Rules in a Production Environment

Feature	Description
Test mode	For evaluating personalization rules. See About Test Mode, Setting Up the Test Mode, and Using Test Mode to Test Personalization Rules.
Log file	For recording personalization events. See About Using the Log File, Enabling Personalization Event Logging, and Testing Siebel Personalization.
XML export	For exporting personalization objects to an XML file. Later you can export this file into a production environment. See About Exporting and Importing Personalization Data, Exporting Personalization Data as an XML File, and Importing Personalization Data.
Reload personalization	For updating the object manager with changes made to personalization rules. See About Clearing and Reloading Siebel Personalization, Reloading Siebel Personalization for the Current Object Manager, and Reloading Siebel Personalization for Other Object Managers.

About Test Mode

The Test view under Personalization Administration allows you to set up a test mode to test personalization rules. The test mode allows you to set up the profiles of test users, specify the test application name and view, and then launch a new instance of the application to test personalization rules.

This section contains the following:

- Setting Up the Test Mode
- Using Test Mode to Test Personalization Rules on page 73

Setting Up the Test Mode

See also About Test Mode and Using Test Mode to Test Personalization Rules.

The following are the steps involved in testing the personalization setup:

1. Specify the login and password of the primary user.

   The primary user is the person logged into the application, for example a Siebel Call Center agent or Siebel eSales registered user.

   Click Open to load a saved setup file.

2. Specify the login of the secondary user, if desired.

   An example of a secondary user is a customer speaking by telephone with the call center agent.

3. Click Load.

   This loads the persistent user profile attributes of the primary user into the Primary User Attributes list and those of the secondary user into the Secondary User Attributes list.

4. Edit the persistent user profile attributes for both users.

   All changes made to the persistent user profile attributes are written only to the test setup, not to the Siebel database.

5. Enter dynamic user profile attributes for both users.
6. Specify the application to test by filling in the URL for launching the application.
7. If you want to test the personalization of a specific view, specify the view.
8. Click Test to get instructions for opening a new instance of the application to test personalization rules.
9. Test the personalization rules.
10. Click Save.

Using Test Mode to Test Personalization Rules

See also About Test Mode and Using Test Mode to Test Personalization Rules.

You can test personalization rules in a staging environment before being used in the production environment. This is done in the Test view under Personalization Administration.

To test personalization rules using Test Mode

1. Navigate to the Administration - Personalization screen > Test view.
2. Either:
   - Click Open to use a saved test setup, or
   - Fill in the following fields, and click Load

Field	Description
Primary User Login	**Required.** User who is logged in to the application, such as a call center agent.
Primary User Password	Required.
Secondary User Login	**Optional.** Someone who interacts with the primary user, such as a customer speaking with the call center agent.

Test Application	Path to the test application.
	■ For the Siebel Web Client use a URL, for example:
	`http://<machine_name>/callcenter`
	■ For the Siebel Mobile Web Client use a command string, for example:
	`D:\<Siebel_install_dir>\BIN\siebel.exe /l language_code /c "D:\<Siebel_install_dir>\bin\ENU\config_file.cfg" /d data_source`
	where:
	■ *<Siebel_install_dir>* is the full path to the client installation directory
	■ *language_code* is the three-letter code for the language, for example, ENU for US English and
	■ *config_file* is the application configuration file, for example, uagent for Siebel Call Center and siebel for Siebel Sales
	■ *data_source* is the database to which to connect: Local, Sample, or ServerDataSrc

Field	Description
Test View	**Optional.** If not specified, the screen is blank when the test application launches. You have to navigate to a view.

3 Select an attribute to edit its value.

- The persistent user profile attributes of the:
  - Primary user are loaded into the Primary User Attributes list with a `Me.` prefix.
  - Secondary user are loaded into the Secondary User Attributes list with a `You.` prefix.
  - Name and Source fields are read-only for persistent user profile attributes.
- Person-related attributes have the value `Person` in the Source field.
- Organization-related attributes have the value `Organization`.

4 Add a new record under Primary User Attributes or Secondary User Attributes to add a new dynamic user profile attribute.

5 Fill in the Name and Value fields.

The Source field is read-only and has the value Dynamic.

6 Click Test under the Test tab and take the following action.

If...	Then...
You are using a connected Web Client (accessed though an appropriate URL).	The Test Mode dialog box appears, which prompts users to open a new browser window (by using a desktop icon or the Start menu, if using Windows). The Test Mode dialog box provides a URL to paste into the new browser window.
You are using a Mobile Web Client.	A new instance of the application opens. Go to Step 9.

7 Copy the URL shown in the Test Mode dialog box, and then click OK.

8 Open a new browser window, and then paste the URL into it.

A new instance of the specified application launches.

If the test view was not specified, the screen is blank and you have to navigate to a view.

9 After testing personalization rules, click Save in the Test view to save the test setup.

About Using the Log File

You can write a record of personalization events that occur to a log file. Use this log for testing and diagnosing personalization rules.

Running personalization with the log turned on does have a performance impact. Also, log files can become extremely large, which has an impact on disk space. You should not enable the log in production environments for longer than it takes to acquire the information or do the testing you need.

This section includes the following topics:

Enabling Personalization Event Logging

See also About Using the Log File and Testing Siebel Personalization.

The following procedure describes how to enable Personalization event logging.

To enable Personalization Event logging

1 Take one of the following actions, as appropriate.

If...	Then...

You are using the Siebel Mobile Web Client	Add an entry to the [Siebel] section of the configuration file for your application as follows:
	`PersonalizationLog = <directory path for log file + file name>`
	For example,
	`PersonalizationLog = C:\Temp\Personalization.txt`

You are using the Siebel Web Client	Navigate to the Administration - Server Configuration screen > Servers view, and click the Components tab.

2 Select the appropriate object manager from the components list, for example Call Center Object Manager (ENU), then click the Parameters subtab.

3 Select the Application Personal Log parameter from the advanced parameters list, and edit the following fields:

NOTE: To display the advanced parameters list, click the Advanced button in the Component Parameters applet.

Field	Value
Current Value	Directory path for the log file
Value on Restart	Directory path for the log file

Hidden Application Personal Log Parameter

In Siebel CRM version 7.8, the Application Personal Log parameter is hidden and can only be accessed from the server manager command line.

You can use following commands for hidden parameters:

- To List: `list advanced parameters for comp <comp_alias>`
- To Change: `change param <ParamName>=<Paramvalue> for comp <comp_alias>`
- To Delete: `Delete parameter override for comp <comp_alias> param <ParamName>`

To make sure personalization and runtime events are *on* and a log can be obtained, perform the steps in the following procedure.

To enable Personalization Event Logging when the Application Personal Log parameter is hidden

1. Log in to server manager and issue the following commands:

   ```
   spool c:\list_of_hidden_parameters.txt
   list advanced parameters for comp UCMObjMgr_enu
   spool off
   ```

   This generates the following file at the following location:

   `c:\list_of_hidden_parameters.txt`

   You can change the file name and location.

2. Open the file and check the parameter values:

   - Set CFGEnableRuntimeEvents to True:

     `change param CFGEnableRuntimeEvents=True for comp UCMObjMgr_enu`

   - Set tCFGEnablePersonalization to True (this is the equivalent of Application Enable Personalization):

     `change param CFGEnablePersonalization=True for comp UCMObjMgr_enu`

   - Set CFGPersonalLog to the full path and filename to generate the log:

     `change param CFGPersonalLog=c:\personalization.log for comp UCMObjMg`

3. After the changes, restart the server and gateway.

Testing Siebel Personalization

This section includes the following topics:

- Changing the Test Parameters
- Using the Log File to Test Siebel Personalization

See also About Using the Log File and Enabling Personalization Event Logging.

Changing the Test Parameters

You can use the EnablePersonalization and EnableRuntimeEvents object manager parameters to test Siebel Personalization. These parameters are found in the `srvrdefs.dat` file.

To change these parameters

- Navigate to the Administration - Server Configuration screen > Servers view > Parameters view.

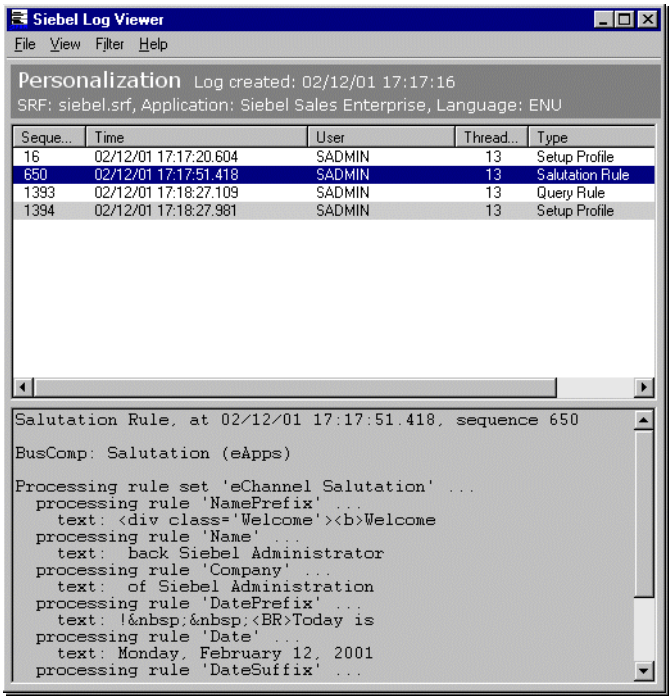
NOTE: You can also add these parameters to the application configuration file.

Using the Log File to Test Siebel Personalization

The following procedure describes using the log file to test Siebel Personalization.

To test Siebel Personalization using the log file

1. Run the personalized application with the log enabled.
2. Open the log file using the graphical application SSFLOG.EXE, which is located in the <*Siebel_install_dir*>\bin directory, and shown in the following illustration.



3 Examine the log file under the following conditions.

Condition	What to Look For
EnablePersonalization = FALSE EnableRuntimeEvents = FALSE	■ Persistent user profile attributes are not loaded. ■ Features based on personalization rules do not work. ■ No run-time events occur.
EnablePersonalization = FALSE EnableRuntimeEvents = TRUE	■ Features based on personalization rules do not work. ■ Persistent user profile attributes are not loaded. ■ Run-time events occur.
EnablePersonalization = TRUE EnableRuntimeEvents = FALSE	■ Features based on run-time events do not work. Some personalization features also might not work, as they might depend on events. ■ Persistent user profile attributes are loaded. ■ Run-time events do not occur.
EnablePersonalization = TRUE EnableRuntimeEvents = TRUE	■ Siebel Personalization works properly. ■ Persistent user profile attributes are loaded. ■ Run-time events occur.

About Exporting and Importing Personalization Data

You can export personalization rules, events, and actions as an XML file for later importation into another Siebel environment.

This section includes the following topics:

Exporting Personalization Data as an XML File

See also About Exporting and Importing Personalization Data and Importing Personalization Data.

This procedure describes how to export all personalization data rules, events, and actions in one XML file.

To export personalization data as an XML file

1. Navigate to the Administration - Personalization screen > Views view.
2. Select any Personalization Administration view.
3. Click the menu button, and then choose XML Export.
4. Select Save.

   The default file name is personalization.xml.

Importing Personalization Data

This section includes a description of how the XML import works and a procedure on how to import all personalization data rules, events, and actions in one XML file.

Process of How the XML Import Works

The import utility uses the business component's primary user key sequence to match the record in the XML file and the record in database. For example, for the business component Personalization Rule Set, the user key is Name. Table 26 lists the consequences if the user key is change or the sequence number is changed.

NOTE: You can find out the user key column from the table definition. For the Personalization Rule Set business component, the associated table, S_CT_RULE_SET has a single user key column NAME.

Table 26. User Key or Sequence Number Changes and the Consequences

If...	Then...
The value for the user key itself (Name field) is changed.	The import utility does not find an existing record matching the Name field in the XML file, thus considering it being a new record. A new record is created.
The sequence number is changed.	It is an update, not an insert, since Sequence number is not part of the user key, and the import utility finds the record matching field Name in the database.

NOTE: The import utility has an implicit rule, that is, a record is updated only if the LAST_UPD timestamp in the XML file is newer than in the database. This prevents an old XML file from flushing all the newer configurations.

See also About Exporting and Importing Personalization Data and Exporting Personalization Data as an XML File.

Importing Personalization Data from an XML File

The following procedure describes how to import personalization data from an XML file.

NOTE: This process can take several minutes.

To import personalization data from an XML file

1. Navigate to the Administration - Personalization screen > Views view.
2. Click the menu button, and choose XML Import.

   When the process is finished, a message displays how many records had conflicts and were inserted, updated, and skipped.

About Clearing and Reloading Siebel Personalization

The Siebel object manager caches personalization data for improved performance. You must reload the personalization data into the object manager to see the effect of personalization rule changes.

This section consists of the following topics:

- Reloading Siebel Personalization for the Current Object Manager
- Reloading Siebel Personalization for Other Object Managers

Reloading Siebel Personalization for the Current Object Manager

For the current object manager, reloading Siebel Personalization is performed in the Views, Applets, or Rule Sets view under Administration - Personalization.

See also Reloading Siebel Personalization for Other Object Managers.

To clear and reload Siebel Personalization

1. Navigate to the Administration - Personalization screen > Views view.
2. Click the menu button, and choose Reload Personalization.

Reloading Siebel Personalization for Other Object Managers

To propagate changes in Siebel Personalization to other object managers, use the Enterprise Configuration view under Administration - Server Configuration.

For more information on working with object managers, see the *Siebel Installation Guide* for the operating system you are using.

See also Reloading Siebel Personalization for the Current Object Manager.

To clear and reload Siebel Personalization

1. Navigate to the Administration - Server Configuration screen > Enterprises view.
2. Click the Component Definitions view tab.
3. In the Component Definitions list at the top of the view, select the object manager for which to clear and reload personalization rules, for example Call Center Object Manager.
4. Click the menu button under the upper Component Definitions list and choose Start Reconfiguration.

   The value of the Definition State field for that object manager changes to Reconfiguring.
5. Select the desired object manager again, click the menu button under the upper Component Definitions list, and then choose Commit Reconfiguration.

   When the definition state changes to Active, the personalization data has been reloaded.

A Operators for Building Condition Expressions

This section describes the operators that are available for building condition expressions specific to Siebel Personalization. It includes the following topics.

- Arithmetic Operators
- Comparison Operators
- Logical Operators
- Pattern Matching Operators

For more information on operators, see *Using Siebel Tools*.

Arithmetic Operators

Table 27 lists arithmetic operators.

See also Comparison Operators, Logical Operators, and Pattern Matching Operators.

Table 27. Arithmetic Operators

Operator	Description
+	Add
–	Subtract
*	Multiply
/	Divide
^	Power

Comparison Operators

Table 28 lists comparison operators.

See also Arithmetic Operators, Logical Operators, and Pattern Matching Operators.

Table 28. Comparison Operators

Operator	Description
=	Equal to a specific value.
< >	Not equal to a specific value.
>	Greater than a specific value.
<	Less than a specific value.
> =	Greater than or equal to a specific value.
< =	Less than or equal to a specific value.
~	NOT. Used to exclude values.
IS NULL	Finds blank fields. Using the = operator without any accompanying value does not give you the same results as IS NULL.
IS NOT NULL	Finds non-blank fields.

Logical Operators

Table 29 lists logical operators.

See also Arithmetic Operators, Comparison Operators, and Pattern Matching Operators.

Table 29. Logical Operators

Operator	Description
AND	All conditions connected by AND must be TRUE.
OR	At least one condition connected by OR must be TRUE.
NOT	The condition modified by NOT must be FALSE.

Pattern Matching Operators

Table 30 lists pattern matching operators.

See also Arithmetic Operators, Comparison Operators, and Logical Operators.

Table 30. Pattern Matching Operators

Operator	Description
*	(asterisk) Wildcard for any number of characters before or after the value entered. Used in combination with LIKE.
?	(question mark) Wildcard for a single character. Used in combination with LIKE.
LIKE	Combine with a string and one or more wildcard characters to find all values with string. For example: [Product Line] LIKE *enti?m* returns records where the Product Line field contains the string enti?m such as The Pentium I Line and Pentiums.
NOT LIKE	Combine with a string and one or more wildcard characters to find all values without string.
EXISTS	Finds a value from a multivalue group. EXISTS also finds values in any child of the MVG. For example: EXISTS ([Product Component] = 4 Meg Ram) returns records in which attribute Product Component (an MVG) referred to is a child record with the value 4 Meg Ram.

B Functions

This section describes the operators that are available for building condition expressions and the functions that are specific to Siebel Personalization. It includes the following topics:

NOTE: Siebel Query Language parses literal strings incorrectly. For example, the expression “ ” = “X”, where “ ” is an empty string, throws the following error: *“ ” is not a field name*.

String Functions

Table 31 lists string functions. See also Functions.

Table 31. String Functions

Function	Description
Left (string, integer)	Returns the left-most *n* characters in the text string or field. For example: `Left (“Adams”, 2)` returns “Ad”.

Table 31. String Functions

Function	Description

Mid (string, start [, integer])	Returns *n* characters from the string starting from the start location.
	For example:
	`Mid (“Adams”, 2, 2)`
	returns “da”
	`Mid (“Adams”, 2)`
	returns “dams”.

Right (string, integer)	Returns the right-most *n* characters in the text string or field. For example: `Right (“Adams”, 2)` returns “ms”.
Len (string)	Returns the length of character string. For example: `Len (“foo”)` returns 3.

InStr (string1, string2 [, start] [, compare])	The position of the first character of string2 in string1.
	[,start] is used to specify the position in string1 to begin the search, where the first character in the string is 1. If start is negative, InStr searches backwards.
	[,compare] is used to specify if the comparison of strings is case sensitive. Enter 0 for a case-sensitive search, enter 1 for a case-insensitive search.
	For example:
	`InStr (“foo”, “oo”)`
	returns 2.
	`InStr (“bda”, “Adams”, 2)`
	returns 2.
	`InStr (“BDA”, “Adams”, 2, 1)`
	returns 2.
	If the string is not found, the function returns 0.

Table 31. String Functions

Function	Description
InList (string1, comma-separated list)	Returns if string1 occurs one or more times in a comma-separated list. For example: `InList(“foo”, “abc,boo,foo”)` returns TRUE.

Conditional Functions

Table 32 lists the conditional functions. See also Functions.

Table 32. Conditional Functions

Function	Description
IfNull (expr1, expr2)	Returns expr1 if expr1 is not NULL or returns expr2 if expr1 is NULL. IfNull is the return type of its first argument even if the first argument is NULL. For example: `IfNull (“”, “foo”)` returns “foo”.
IIf (testExpr, expr1, expr2)	If testExpr is TRUE, returns expr1’s value, otherwise returns expr2’s value. IfNull is the return type of its first argument even if the first argument is NULL. The second argument is converted to the type of the first argument before its value is returned. For example: `IIf ([Last Name] IS NULL, “foo”, “boo”)` returns foo if [Last Name] is NULL.
IsPrimary ()	For MVGs, returns if the current record is the primary record. IsPrimary () returns TRUE if the current address in an MVG address field is the primary address.

Table 32. Conditional Functions

Function	Description
BCHasRows (BO, BC, search_expr, visibility)	Returns TRUE if the business component BC that is a part of business object BO has any rows after applying the search_expr and visibility. Evaluates to TRUE or FALSE depending on whether a given business component would return any records given the search specification and visibility. For example: `BCHasRows (“Contact”, “Contact”, “”, “Organization”)` returns TRUE if there are 1 or more rows of contacts. For example, you could hide an applet based on how many records it displayed. However, hiding and showing detail applets using BCHasRows is not possible.

Lookup Functions

Table 33 lists lookup functions. See also Functions.

NOTE: The Lookup functionality is based on the LOV record's LANG_ID column matching the language code of the currently active language.

Table 33. Lookup Functions

Function	Description
Lookup (type, value)	Finds a row in the List of Values table (S_LST_OF_VAL) where the TYPE column matches the type argument and the VALUE column matches the value argument. The function returns the value of the ORDER_BY column for that row. For example: `Lookup (“MR_MS”, “Ms.”)` returns “2”.

Table 33. Lookup Functions

Function	Description
LookupExpr (type, value_expr)	Searches the rows in the List of Values table (S_LST_OF_VAL) where the TYPE column matches the type argument. LookupExpr evaluates the contents of the VALUE column treated as an expression. Returns the value of the ORDER_BY column for the first row for which the expression evaluates to TRUE. For example: `LookupExpr (“MR_MS”, “M*”)` returns “1”.
LookupName (type, lang_ind_code)	Finds a row in the List of Values table (S_LST_OF_VAL) where the TYPE column matches the type argument, the NAME column matches the lang_ind_code argument, and the LANG_ID column matches the language code of the currently active language. LookupName returns the language-independent code (the NAME column) for the row. For example: `LookupName (“MR_MS”, “Ms.”)` returns “Ms.”
ParentFieldValue (field_name)	The value of the field_name field in the parent business component. The result does not change if the parent row is updated. The parent business component field must be exported by using Link Specification. For example: for the Opportunity business component `ParentFieldValue(“Account”)` returns “Name”.
FieldValue (field_name)	Return value of field in the business component as a string. For example: for Contacts business component for Henry Kim. `FieldValue (“Last Name”)` returns “Kim”.

Table 33. Lookup Functions

Function	Description
LookupValue (type, lang_ind_code)	Finds a row in the List of Values table (S_LST_OF_VAL) where the TYPE column matches the type argument, the NAME column matches the lang_ind_code argument, and the LANG_ID column matches the language code of the currently active language. LookupValue returns the display value for the specified lang_ind_code, and if not found, the lang_ind_code itself as the value. For example: `LookupValue (“MR_MS”, “Ms.”)` returns “Ms.”

Translation Functions

Table 34 lists the translation functions. See also Functions.

Table 34. Translation Functions

Function	Description
LookupTransVal (lang_ind_code, table, type)	For a multilingual list of values, given a language independent code, translation table (must be S_LST_OF_VAL), and LOV type, returns the display value. For example: `LookupTransVal (“Mr.”, “S_LST_OF_VAL”, “MR_MS”` returns “Mr.”
LookupTransCode (display_value, table, type)	For a multilingual list of values, given a display value, translation table (must be S_LST_OF_VAL), and LOV type, returns the language independent code. For example: `LookupTransCode (“Mr.”, “S_LST_OF_VAL”, “MR_MS”` returns “Mr.”
LookupTranslation ([field])	For the new multilingual data feature. Takes a field as an argument, and if the field has a translation, returns the translation. Otherwise returns the value of the field. For example: `LookupTranslation([Name])` returns “Sam” if the current language is English.

Table 34. Translation Functions

Function	Description
Preference (category, pref_name)	Finds the user preference for the category and preference name specified. For example: `Preference (“User Interface”, “StyleSheet”)` returns the user-specified style sheet preference.
LoginName()	Returns the login ID of the logged in user.

Search Functions

Table 35 lists the search functions. See also Functions.

Table 35. Search Functions

Function	Description
FindOneOf (string1, string2)	Returns 1-based index of the first instance in string1 of a character in string2. For example: `FindOneOf (“abcdef”, “xyzc”)` returns 3.
FindNoneOf (string1, string2)	Returns 1-based index of the first instance in string1 which does not match any character in string2. For example: `FindNoneOf (“abcdef”, “xyzc”)` returns 1.

Math Functions

Table 36 lists the math function. See also Functions.

Table 36. Math Functions

Function	Description
Sum (mvfield)	Sums the values from a field in child records. You must define the child record that is being summed from as a multivalue field that is part of a multivalue group. The multivalue group is associated with the business component of the field that is being summed. For example: `Sum ([Number of Employees])` gives the sum of all the employees at different locations for a company.
Count (mvfield)	Returns the number of rows in the multi-value group defined by the mvfield. For example: `Count([Number of Employees])` gives the number of employees at different locations for a company.
ToChar ([field_name], 'format')	Returns a string that represents a number or date in a format specified by the optional format parameter. For example: `ToChar([Start Date], 'MM/DD/YYYY')` returns the starting date of a record as a string in MM/DD/YYYY format.
Min (mvfield)	Returns the minimum value from a field in child records. You must define the child record being examined as a multivalue field that is part of a multivalue group. The mutlivalue group is associated with the business component of the field being evaluated. For example: `Min ([Number of Employees])` gives the minimum number of employees of all the locations.

Table 36. Math Functions

Function	Description
Max (mvfield)	Returns the Maximum value from a field in child records. You must define the child record being examined as a multivalue field that is part of a multivalue group. The mutlvalue group is associated with the business component of the field being evaluated. For example: `Max ([Number of Employees])` gives the maximum number of employees of all the locations.

InvokeServiceMethod (name, method, context, returnProperty)	Returns the value of the return property from the returnProperty set of the specified business service, after invoking the method with the context.
	
	For example:
	
	`InvokeServiceMethod (“BusServ”, “PersonalizationMethod”, “Key1=a,Key2=2”, “ReturnProperty”)`
	
	invokes the business service method PersonalizationMethod in business service BusServ, passes it the context Key1=a,Key2=2, and returns the value set by the business service in the property ReturnProperty.
	
	To use the InvokeServiceMethod function in client mode, the business service name that is called by the InvokeServiceMethod function must be added as the BusinessServiceQueryAccessList parameter value under the [Siebel] section of the application configuration file as follows:
	
	`[Siebel]`
	`BusinessServiceQueryAccessList = BusSvc1,BusSvc2,...`
	
	Likewise, in Server mode, the component parameter BusinessServiceQueryAccessList must also be set to specify the list.

LN (num)	Returns the natural log of the num. For example: `LN (10)` returns 2.30.

Table 36. Math Functions

Function	Description
GetNumBCRows (BO, BC, search_expr, visibility)	Returns the number of rows business component BC has, which is part of business object BO, after applying the search_expr and visibility. For example: `GetNumBCRows (“Contact”, “Contact”, “A*”, “Organization”)` returns the number of rows that match the criteria.

Date and Time Functions

Table 37 lists date and time functions. See also Functions.

Table 37. Date and Time Functions

Function	Description
JulianDay ()	Equal to the Oracle (and Sagent) Julian Day for all dates in the 20th and 21st centuries. Assigns an absolute numeric value to every date in order to perform calculations on the dates.
JulianMonth ()	Equal to the JulianYear() * 12 + currentMonth, where January = 1.
JulianQtr ()	Equal to the JulianYear () * 4 + currentQuarter, where currentQuarter = [(currentMonth -1) / 3+1] rounded down to the next integer.
JulianWeek ()	JulianDay () / 7, rounded down to the next integer.
JulianYear ()	Equal to the current year + 4713.
Today ()	Retuns the current date. For example: 2/15/2001 Today() does not do the UTC (universal time code) conversion and TimeStamp() does do the conversion. **NOTE:** Today() and Timestamp() functions return different results. For more information, see Timestamp ().

Table 37. Date and Time Functions

Function	Description
Timestamp ()	Returns today's date and time. For example: 2/15/2001 11:15:22 TimeStamp() does the UTC (universal time code) conversion and Today() does not do the conversion. **NOTE:** Today() and Timestamp() functions may return different results. For example: ■ Add these fields to a standard report. ■ On the machine that hosts the Report Server and Siebel Server, change the time zone to one different from the user machine. For example, 10/03/03 1:45 A.M. (EST) would be 10/02/03 10:45 P.M. on the user's machine. When running this report from the user's machine (Web client), while the Timestamp() field shows the correct time in the user's time zone, the Today() field displays 10/03/03, even though it is still 10/02/03 for the user.
UtcConvert (utc_date_time, time_zone	This function converts a local time (in the current user's logged in user's time zone) to another local time in the specified time zone. For example, if the user is on Pacific time the time is converted to eastern time: `UtcConvert("12/14/2000 5:07:05 PM", "Eastern Standard Time")` returns "12/14/2000 00:07:05 PM"

Profile Functions

Table 38 list the profile functions. See also Functions.

Table 38. Profile Functions

Function	Description

GetProfileAttr(‘profile’)	Returns the profile attribute of the current user.
	
	For example:
	
	`GetProfileAttr ('Personal Country')`
	
	returns the country of the user.
	
	GetProfileAttr only works for fields explicitly defined in the Personalization Profile business component. The function does not work with system fields, which are not explicitly defined in the business component; it returns a NULL value for them.
	
	An exception is the Id system field. This particular system field is available to GetProfileAttr even though it is not in the Personalization Profile business component.
	
	If this function is called for an MVG, it returns just the primary value of the MVG. For example, if the MVG State has the values CA, MA, and GA, where CA is primary:
	
	`GetProfileAttr ('State')`
	
	returns CA.

GetProfileAttr(“org.country”)	Returns one of the following: ■ Where there is an account associated with the logged in user, `GetProfileAttr("org.country")` returns the Country of the contact account’s primary address. ■ Where there is no account associated with the logged in user, then `GetProfileAttr("org.country")` returns the Country of the user organization’s primary address. **NOTE:** To ensure that you retrieve the organization name in instances where the employee is associated with an account as a contact, it is recommended that you map an additional attribute on the Personalization Profile business component (which you can use to retrieve the organization name).

Table 38. Profile Functions

Function	Description
GetProfileAttrAsInt(‘profile’)	Returns the profile attribute of the current user as an integer. For example: `GetProfileAttrAsInt ('Age')` returns the age of the user as an integer.

GetProfileAttrAsList (‘profile’)	Returns the MVG value as a list. You can use GetProfileAttrAsList in the EXISTS operator to create expressions that match MVG profile attributes with appropriate Siebel content.
	
	If GetProfileAttrAsList is used:
	
	■ Outside the Exists operator, then it returns a comma-separated list of the MVG values. For example, if the MVG State has the values CA, MA, and GA, where CA is primary it returns CA, MA, GA.
	
	`GetProfileAttrAsList ('State')`
	
	■ For a single value field, then it returns the profile attribute value.
	
	■ Within the EXISTS operator, then it returns the value profile attributes in the form expected by the user. For example, you could use the EXISTS operator this way:
	
	`EXISTS ([Targeted States] = GetProfileAttrAsList (“State”)]`
	
	This action would match the MVG business component field Targeted State against the MVG profile attribute State.

GetProfileAttrAsNum (‘profile’)	Returns the profile attribute of the current user as a number. For example: `GetProfileAttrAsNum ('Age')` returns the age of the user as a number.
GetProfileAttrAsDate (‘profile’)	Returns the profile attribute of the current user as a date. For example: `GetProfileAttr ('Birth Day')` returns the birthday of the user as a date.

Table 38. Profile Functions

Function	Description
GetProfileAttrAsTime (‘profile’)	Returns the profile attribute of the current user as time. For example: `GetProfileAttr ('Last Login Time')` returns the last login time of the user as time.
GetProfileAttrAsDateTime (profile_attrib)	Returns the profile attribute of the current user as date and time. For example: `GetProfileAttr ('Last Login Date')` returns the last login date and time of the user as date and time.
GetProfileAttrAsUtcDateTime (profile_attrib)	Returns the profile_attrib formatted in the UtcDateTime type. For example: `GetProfileAttrAsUtcDateTime ('Date')` returns date time in UTC format.

Attribute Functions

Table 39 lists attribute functions. See also Functions.

Table 39. Attribute Functions

Function	Description
XAIsClass (classname)	Returns TRUE if the current object belongs to the class classname including subclasses. For example: `XAIsClass (“”Car””)` returns TRUE for Honda Accord.
GetXAVal (attributename)	Returns the value for the specified string-type attribute for the current object if any. For example: `GetXAVal (“”Seat Type””)` returns leather for a luxury car with leather seats.

Table 39. Attribute Functions

Function	Description
GetXAValAsInt (attributename)	Returns the value for the specified integer-type attribute for the current object if any. For example: `GetXAValAsInt (""Doors"")` returns 4 for a four-door sedan.
GetXAValAsNum (attributename)	Returns the value for the specified number-type attribute for the current object if any. For example: `GetXAValAsNum (""Doors"")` returns 4 for a four-door sedan.
GetXAValAsDate (attributename)	Returns the value for the specified date-type attribute for the current object if any. For example: `GetXAValAsDate (""Release Date"")` returns the release date attribute for the row as date.

Other Functions

Table 40 lists other functions. See also Functions.

Table 40. Other Functions

Function	Description
SystemPreference (“Pref”)	Get value of system preference. These values are found in the Administration - Application screen > System Preferences view. For example: `SystemPreference (""Auto Mgr Calendar Access"")` returns TRUE.
LoginId ()	Returns the Login ID of the logged in user.

Table 40. Other Functions

Function	Description
BCName ()	Returns the name of the active business component. For example: `BCName ()` returns ““Account””.
Currency ()	Returns the currency for the position of the logged in user. For example: USD
LocalCurrency ()	Returns the currency for the machine. For example: JPY
Language ()	Language code (for example, ENU) that is the active client language setting. The language code is set by either the Language parameter in the .cfg file, or by the /L parameter when starting Oracle’s Siebel application.
RowidToRowidNum ()	Converts the Row ID of a record to a number. For example: `RowidToRowidNum ()` returns the unique integer translation of Rowid.

Index

S

T

U

V

X