Siebel REST API Guide

Siebel Innovation Pack 2016, Rev. A
July 2016

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

Index

1 What's New in This Release

What's New in Siebel REST API Guide, Siebel Innovation Pack 2016, Rev. A

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

Table 1. New Product Features in Siebel REST API Guide, Siebel Innovation Pack 2016, Rev. A

Topic	Description
"About Configuring OAuth 2.0 for Authentication" on page 17	New topic. Describes how to configure the OAuth 2.0 protocol for authentication with the Siebel REST API.

What's New in Siebel REST API Guide, Siebel Innovation Pack 2016

Table 2 lists the changes described in this version of the documentation to support this release of the software.

Table 2. New Product Features in Siebel REST API Guide, Siebel Innovation Pack 2016

Topic	Description
"Overview of Using the Siebel REST API" on page 7	New Chapter. This chapter provides an overview of the Siebel CRM Representational State Transfer (REST) application programming interface (API).
"Getting Started with the Siebel REST API" on page 23	New Chapter. This chapter describes how to get started with using the Siebel REST API.
"Using the Siebel REST API" on page 33	New Chapter. This chapter describes the Siebel REST API requests and responses for REST API calls to access Siebel CRM resources.

2 Overview of Using the Siebel REST API

This chapter provides an overview of the Siebel CRM Representational State Transfer (REST) application programming interface (API). It includes the following topics:

About Siebel CRM REST API

REST (Representational State Transfer) is a software architecture style that provides a convenient and consistent approach to requesting and modifying data. In the Siebel CRM RESTful system, resources are stored on the Siebel Server; a client sends a request using an HTTP verb (such as GET, POST, PUT, or DELETE) that the Siebel Server perform a particular action (such as querying, inserting, upserting, or deleting a Siebel CRM resource), and the Siebel Server performs the action and sends a response.

The Siebel REST API is implemented as part of the existing EAI component group, so enabling the EAI component group will enable the REST related components on the Siebel Server. The Siebel REST API exposes Siebel Business Objects, Siebel Business Services, and Siebel Repository Objects. For more information about Siebel Business Objects, Siebel Business Services, and Siebel Repository Objects see *Configuring Siebel Business Applications*.

The following are the salient aspects of the Siebel REST API that are aligned with general best practices of REST APIs:

- A base URI to access Siebel Server resources, for example:

  `http://Server Name:port/siebel-rest/`

- Support for JSON resource representations.

- Operations on Siebel CRM resources are mapped to semantically similar HTTP methods, such as GET, PUT, POST, AND DELETE.
- Hypertext links to Siebel CRM Child Business Components in the case of the Data API resources in the REST API response.

About Siebel REST API Architecture

The Siebel CRM REST API services are deployed as a WAR file (siebel-rest.war) on the Siebel Java Web Container. The Java Web Container is embedded inside the Siebel CRM server. Siebel REST API requests flow through the Siebel Java Web Container. The Siebel Java Web Container routes the REST API requests to the appropriate Siebel CRM Server or Siebel Application Object Manager session.The Siebel REST Proxy Object Manger proxies for the Java Web Container, allowing management as a Siebel Component. The Java Web Container is managed by the Siebel Rest Proxy Object Manager, which uses the Server Management Infrastructure to manage the Java Web Containers. The Java Web Container hosts the Siebel CRM REST API infrastructure and provides the Siebel CRM REST API interface.

If you plan to expose access for all Siebel channels under a single Web domain, then a reverse proxy can be configured in front of both the pre-existing Siebel Web Server and the new Siebel Java Web Container. Also, if more than one Java Web Container is provisioned, then the Java Web Containers would need to be load balanced, and that can be done by configuring a Web Server with load balancing capability. For more information about reverse proxy and load balancing, see *Siebel Installation Guide* for the operating system you are using, *Siebel Deployment Planning Guide*, and *Siebel System Administration Guide*.

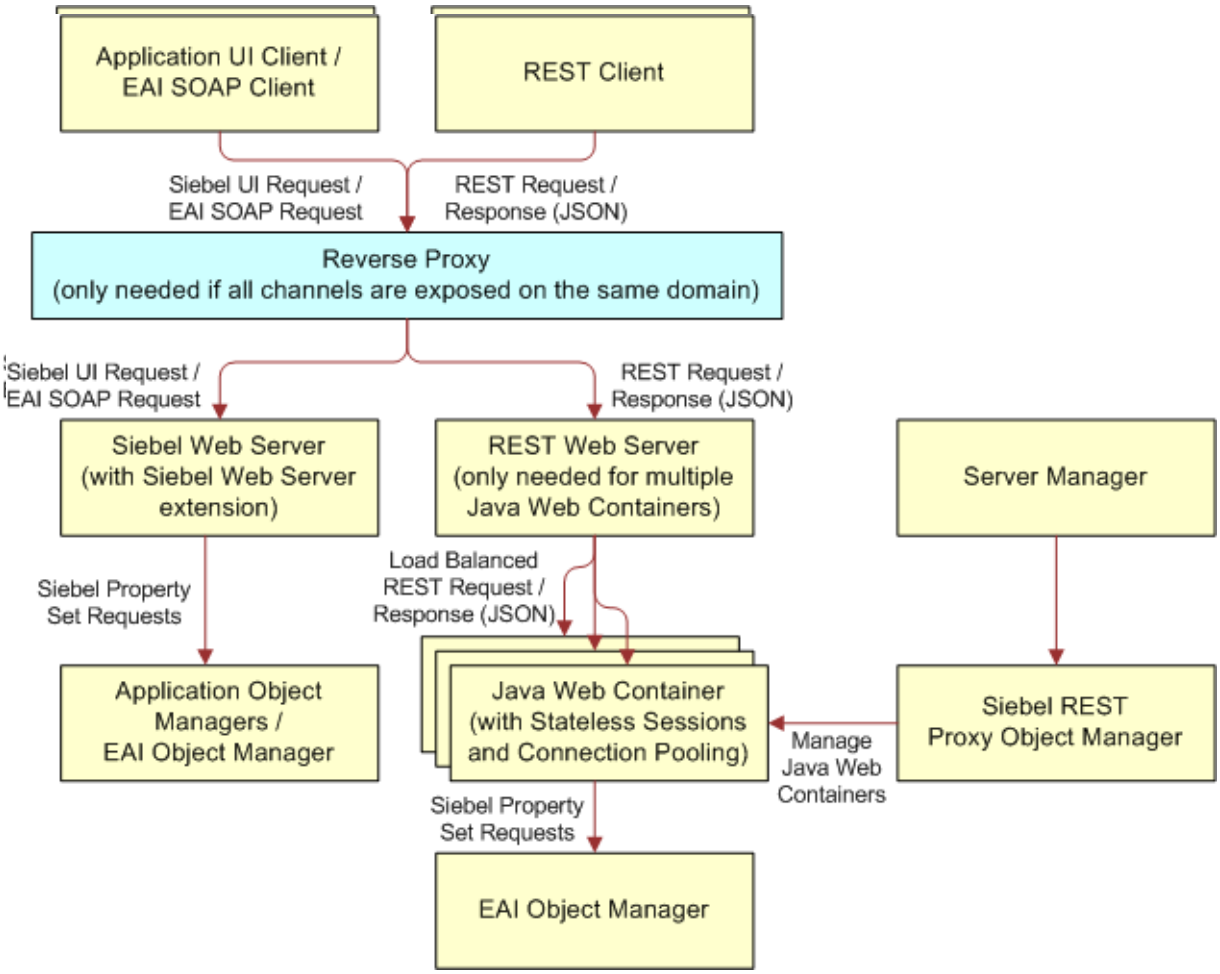

Figure 1. Siebel REST API Architecture

Figure 1 shows the high-level architecture of implementation REST support on the Siebel Server.

- Application UI Client/EAI SOAP Client. The SOAP client used to access the Siebel CRM Server.
- REST Client. The REST API client used to access the Siebel CRM Server.
- Reverse Proxy (Optional). The Siebel REST API uses a different Web server than the existing Siebel User Interface and SOAP Enterprise Application Integration channels. If you plan to expose access for all Siebel channels under a single Web domain, then a reverse proxy can be configured in front of both the pre-existing Siebel Web Server and the new Siebel Java Web Container. For more information about configuring a reverse proxy, see "Configuring the Reverse Proxy" on page 28
- Siebel Web Server. The existing Siebel Web Server that hosts Siebel CRM.

  - Application Object Managers. The existing object managers serving the Siebel User Interface and SOAP Enterprise Application Integration sessions.
- REST Web Server. The server that hosts the Siebel REST API.
- Server Manager. The server manager serves the Siebel REST Proxy Object Manager.
- Java Web Container. The Java Web Container routes the REST API requests to the appropriate Siebel Application Object Manager session.
  - Siebel Rest Proxy Object Manager. The Siebel Rest Proxy Object Manager is a background Siebel Server component used to manage the Java Web Container instances so that the existing Siebel Server Management Infrastructure can start, stop, and restart the instances.
  - Stateless Sessions and Connection Pooling. User Sessions are stateless and reusable. A pool of connections to the Siebel CRM server sessions are maintained by the Java Web Container. To optimize response times, session connections to the Siebel CRM server are held by the Java Web Container for a user until the Siebel Server session times out due to inactivity or is pre-empted by a request from another user when the Connection Pool reaches the maximum size configured.
  - Siebel EAI Object Manager. The Siebel Object Manager serves Siebel CRM REST API requests as well as legacy SOAP requests.

About Siebel CRM REST API Requests and Responses

A request can include the following information:

- A request URI. For more information about URI formats, see "About Siebel CRM REST API URI Formats" on page 11.
  - The request URI contains the base URI signifying what category of Siebel resources to invoke. Siebel resources include: Business Objects, Repository Objects, and Business Services.
  - The request URI contains the object type of the invoked Siebel CRM resource. For example, Account Business Component under Account Business Object. For more information about the supported Siebel CRM resources, "About Siebel CRM REST API Supported Resources" on page 13.
- The HTTP method that you use to perform a REST API operation (query, insert, upsert, or delete) on the Siebel CRM Server. For more information about supported HTTP methods, see "About Supported HTTP Methods" on page 14
- Header information to define the parameters of the interaction with the Siebel CRM Server and the information and format you want in the response. For more information about supported HTTP headers, see "About Supported HTTP Header Fields" on page 14.

After the Siebel CRM Server processes the request, the server sends back a response. The response body contains the results of the REST API call and also additional information based on what was specified in the request.

The response can include the following information:

- A HTTP status code that indicates whether the request was successful or failed. For more information about HTTP status codes, see "About Standard HTTP Status Codes and Error Messages" on page 15
- A list of Siebel proprietary field and value pairs.
- If the request failed, then the response body includes additional information about the error. For additional information about HTTP error status codes, see "About Standard HTTP Status Codes and Error Messages" on page 15.
- Hypertext links to Siebel CRM child resources, in the REST API response, such as links to Child Business Components in the case of the Data API. For more information about links, see "About Siebel CRM REST API Response Links" on page 16.

About Siebel CRM REST API URI Formats

The Siebel CRM REST API exposes Siebel CRM Business Objects, Business Services, and Repository Objects.

Siebel CRM REST API executes resource requests for each resource category using the following HTTP URI formats:

- Siebel CRM REST API basic URI for Siebel Business Objects:

  `http://Server Name:port/siebel-rest/version/data/`

- Siebel CRM REST API basic URI for Siebel Business Services:

  `http://Server Name:port/siebel-rest/version/service/`

- Siebel CRM REST API URI for a Siebel Repository resource:

  `http://ServerName:port/siebel-rest/version/workspace/`

Where:

- Server Name:port: Indicates the name of the server and port hosting the Siebel REST API services.
- siebel-rest: Indicates the product name for the REST API.
- version: Indicates the current version number, 1.0, of the REST API.
- data: Indicates the requested resource is a Siebel Business Object. For more information about Siebel Business Objects, see *Configuring Siebel Business Applications*.
- service: Indicates the request resource is a Siebel Business Service. For more information about Siebel Business Services, see *Configuring Siebel Business Applications*.
- workspace: Indicates the request resource is Siebel Repository Data. For more information about workspaces, see *Using Siebel Tools*. For more information about Siebel Repository Objects see *Configuring Siebel Business Applications*.

About URI Parameters

When constructing a URI, there are a number of optional parameters available to manage response results. Some parameters are described in Table 3.

Table 3. URI Parameters

Parameter	Description
PageSize	Integer that tells the Siebel Server how many records to return. If you query for all the Siebel contacts whose last name starts with the letter A and you do not want to get too many records (for performance reasons), then you can restrict the number of records returned. You restrict the number of records returned by setting the PageSize parameter to a reasonable number. All records that match the search criteria are returned. For example, PageSize=20 returns only twenty contact records, even if more exist in the Siebel database. If fewer records exist that match that search criteria, then all records are returned (but no more than twenty). **NOTE:** It is recommended that you retrieve the lowest number of records required for any one call. The more records that are returned, the larger the message and the slower the response. The maximum number of records cannot exceed 100.
StartRowNum	Used when there is a need to start returning records at a specific row. For example, StartRowNum=100 starts at row 100 of the record set. The first number in a record set is zero, therefore, this request starts at record 99 (given you start counting from one for the first record). This parameter is useful for paging through a record set N records at a time. For example, if there are 100 records in a record set, but you want to retrieve only ten at a time, then enter StartRowNum=0 and PageSize=10 on the first call, then StartRowNum=10 on the next call, then StartRowNum=20 on the next call, and so on.

The syntax for using URI parameters is the parameter name followed by an equal sign (=) with the value of the parameter, and each parameter is separated from other parameters by an ampersand (&). For example, if you want to set the PageSize parameter to 100 and the startrow parameter to 0 (zero), then you enter: Pagesize=100&StartRowNum=0

About Siebel CRM REST API Supported Resources

A REST API resource is a piece of information, such as a data record or a collection of records. Each Siebel CRM REST API resource is identified by a named URI, and it is accessed using standard HTTP methods. For more information about URI formats, see "About Siebel CRM REST API URI Formats" on page 11. For more information about standard HTTP methods, see "About Supported HTTP Methods" on page 14.

The Siebel CRM REST API supports the following Siebel resources and collections:

- Siebel Business Service methods that have been configured for Siebel REST API access for users with a given Siebel Responsibility. For more information about configuring Siebel Business Services, see "Configuring Business Service Methods for RESTful Access" on page 28.
- Siebel Repository Object Types and Siebel Repository Object Instances. All repository objects that are supported for access through workspaces are accessible through the Siebel Repository REST API. To determine if a repository object is workspace-enabled see the section on editing workspace-enabled repository objects in *Using Siebel Tools*.
- Siebel Business Component records. The Siebel Business Components under the following Siebel Business Objects are available through the Siebel REST API:
  - Access Group
  - Account
  - Action
  - Asset Management
  - Campaign
  - Catalog
  - Contact
  - Correspondence
  - Employee
  - Expense
  - Fund
  - Household
  - Incentive Compensation Plan
  - Internal Product
  - List Mgmt
  - Offer
  - Opportunity
  - Order Entry
  - Payments

- Position
- Price List
- Project
- Proposal
- Quote
- Service Agreement
- Service Request
- Solution
- Territory Management
- Time Sheet
- Usage Pattern Tracking

About Supported HTTP Methods

Table 4 contains the HTTP methods supported by the Siebel REST API and the corresponding Siebel CRM operation.

Table 4. Supported HTTP Methods

HTTP Verb	Siebel Operation	Description
GET	Query	The GET method retrieves a Siebel CRM resource.
POST	Insert	The POST method creates a new Siebel CRM resource.
PUT	Upsert	The PUT method upserts a Siebel CRM resource.
DELETE	Delete	The DELETE method deletes a Siebel CRM resource.

About Supported HTTP Header Fields

Certain HTTP header fields define the operating parameters of the REST API transaction with Siebel CRM.

Table 5 contains the HTTP header fields supported by the Siebel REST API.

Table 5. Supported HTTP Header Fields

HTTP Header Field Name	Description	Example
Authorization	The HTTP request header field that indicates the type of authorization. Options include: ■ Basic, if the Authentication type configured in siebsrvr.properties is Basic or SSO ■ Bearer, if the Authentication type configured in siebsrvr.properties is OAuth	Authorization: Basic
Content-Type	The HTTP request and response header field that indicates the content type of the message body. The Siebel REST API supports only JSON encoding for the request body. The Content-Type field is used with POST and PUT requests. When submitting a POST or PUT request, you typically supply a body with the request. You can indicate the format of the response by setting the HTTP Content-Type header on the request.	Content-Type: application/json

About Standard HTTP Status Codes and Error Messages

Siebel CRM REST API uses standard HTTP status codes to indicate the success or failure of API calls. When an error occurs or when a response is successful, the response header contains an HTTP code and the response body usually contains a message accompanying the HTTP response code with additional information about the error.

Table 6 contains the standard HTTP status codes used by the Siebel REST API.

Table 6. Standard HTTP Status Codes and Error Messages

HTTP Code	Message	Description
200	OK	The request successfully executed and the response has content.
204	No Content	The request successfully executed, but the content is not available. For example, the content was deleted.
401	Unauthorized	The request did not have valid authorization credentials.
404	Not Found	The requested resource was not found because of an invalid object name.
405	Not Allowed	The request is not allowed.
406	Not Acceptable	The resource identified by the request is capable of generating only response entities that have content characteristics that are not acceptable according to the accept headers sent in the request.
415	Unsupported Media Type	The data format of the request body, specified in the Content-Type header, is not supported by the targeted resource.
500	Internal Server Error	The server encountered an unexpected error, preventing it from fulfilling the request.

About Siebel CRM REST API Response Links

A link in the Siebel CRM REST API response contains a location of a resource and metadata about that resource.

The Siebel CRM REST API response can include the following types of links:

- **self link.** The original URL that generated the response.
- **canonical link.** The URL for the same resource as the top level resource. If you are already viewing the resource as a top-level resource, then this URL is the same as self. If this resource is not available as a top-level resource, then this link shows child resource context.
- **parent link.** The URL for the parent resource details. This URL of the parent resource is returned in the response only when retrieving details about a child resource.

- **child link.** The URL for the child resource details. The URL returns the path to retrieve each child collection for this record. The href attribute contains the child value. A response can return several child links.
- **association link.** The URL of a specific resource included in the response. There can be many association links.

Each Siebel CRM REST API response link can include the following types of attributes:

- **rel.** Indicates the relationship of the linked resource to the current resource that contains the list of links. Values include: Self, Parent, Child, and Canonical.
- **href.** Indicates the fully qualified location URL of the linked resource.

About User Authentication

Before making REST API calls, users must request a session with the Siebel CRM server. Siebel CRM supports the following mechanisms that the client uses to authentication user credentials:

- Basic authentication over SSL
- OAuth user using OAuth 2.0. For more information about OAuth 2.0, see About Configuring OAuth 2.0 for Authentication on page 17.
- SSO, which can be either:
  - Pre-existing SSO mechanisms used for Siebel Applications or EAI
  - SAML based SSO mechanisms

About Configuring OAuth 2.0 for Authentication

The Siebel REST API can use the OAuth 2.0 protocol for authentication to securely identify applications before connecting to the Siebel Server.

In general, the Siebel REST API layer contacts the OAuth server over a secure channel (for example, HTTPS) to validate the access token received or obtain additional token information. The Siebel Server only requires a USERID to establish a Siebel Server session since authentication takes place outside of Siebel in either SSO or OAuth, and does not require password.

The following prerequisites are required on the Siebel side before configuring OAuth for authentication. You must install and set up the components, including OAuth components, to suit your own business needs. Consult the supporting documentation of your chosen components (for example, Oracle Access Manager and Oracle API Gateway) for more information.

- The Siebel Object Manager must be configured for SSO when OAuth is enabled for authentication. The related security adapter is also required. In SSO mode, when used with a custom security adapter, the specified value is passed as the password parameter to a custom security adapter if the value corresponds to the value of the TrustToken parameter defined for the custom security adapter. For more information about configuring SSO, see *Siebel Security Guide*.

- The Siebel REST API layer contacts the OAuth server over a secure channel to validate or get token information. To enable HTTPS, the required certificates from the OAuth server must be installed in the environment where the Siebel REST API is hosted. To bypass HTTPS (the secured channel), you can set the ValidateCertificate parameter value to FALSE in the siebsrvr.properties file. Oracle does not recommend disabling the secured channel in production. For more information about the siebsrvr.properties file, see "Configuring the Siebel Java Web Container" on page 24.
- The following parameters must be set in the siebsrvr.properties file:
  - **SingleSignOn.** The SingleSignOn property must be set to TRUE to implement SSO.
  - **AuthenticationType.** The AuthenticationType property must be set to OAuth to implement OAuth authentication
  - **TrustToken.** The TrustToken value must be the same as the security adapter TrustToken parameter value.
  - **OAuthEndPoint.** The OAuthEndPoint value is the URL of the OAuth Service Provider end point for Access Token validation.
  - **ValidateCertificate.** The ValidateCertificate parameter is required to bypass the HTTPS secured channel. Oracle does not recommend setting the value for this parameter to False during production. Rather, you should set the value at the point when it is required to bypass the HTTPS secured channel.

For more information about the siebsrvr.properties file properties, see "Configuring the Siebel Java Web Container" on page 24.

While the customer authentication flows vary depending on your business needs, Oracle supports all OAuth 2.0 authentication flows. This topic contains a few sample authentication flows. In all authentication flows, the Siebel REST API layer extracts and validates the Access Token when the authentication type value is OAuth. Customers must generate the authorization and access code. The Siebel Server handles only the resource server initiated flow and any remaining flows must be implemented by the customer.

Client Credentials Grant Authentication Flow

The client credentials grant flow represents an application that calls another application or service, without end user intervention. In this example, the client server application makes a call to the Siebel resource server to request business information. Since there is no end user intervention, the client is pre-authorized to have access to the resource.

Figure 2 is an example of the Client Credentials Grant Authentication Flow

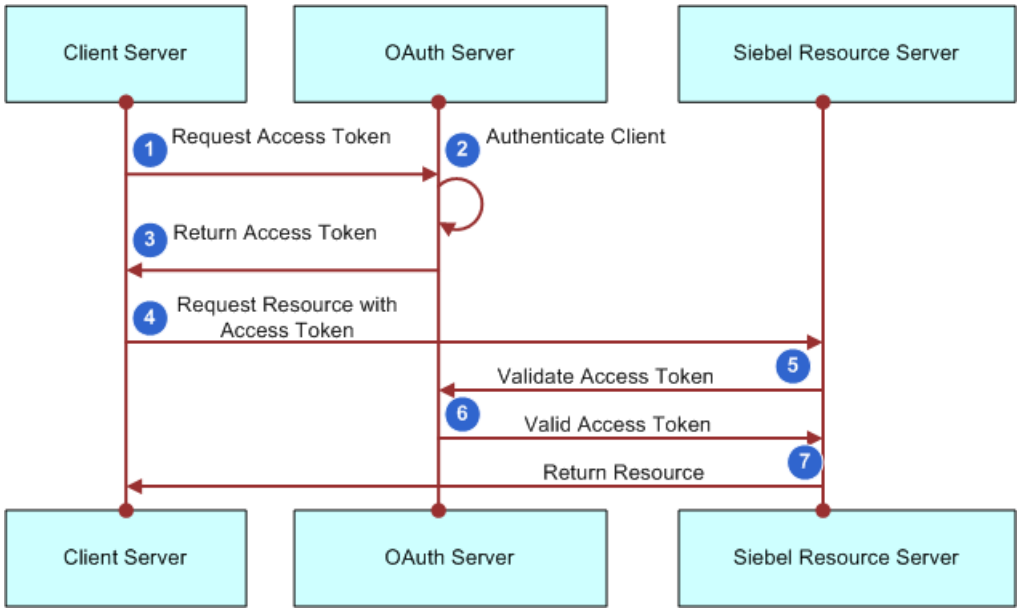

Figure 2. Client Credentials Grant Authentication Flow

The steps in client credentials grant authentication flow process shown in Figure 2 are:

1. The business client application makes a call to the Siebel Server to request some business information by passing an access token. Since there is no end user intervention, the client is pre-authorized to have access to the resource.
2. The request is redirected to the OAuth server for authentication.
3. The OAuth server returns an access token.
4. The client server sends a request to the resource server. The request includes the access token in the HTTP header. Siebel looks for USERID from the token to establish a Siebel Server session.
5. The Siebel server validates the access token with the OAuth server.
6. If the access token is authorized by the OAuth server, then access is granted to the Siebel resource.
7. The Siebel Server returns the requested resource.

Web Server Authentication Flow

In the Web server authentication flow, the client application is running on the client server. In this case, an end user accesses an application, which needs to fetch data from a resource server on behalf of the end user, but without the end user providing his credentials to the application. The user has to provide his consent for the client application to access resources in the resource server. In this use case, all the code resides in the client server and it is not visible to the end user. For the OAuth server, you can use Oracle Access Manager and Oracle Webgate, or any other Web application and gateway depending on your business needs.

Figure 3 is an example of the Web server Authentication Flow.

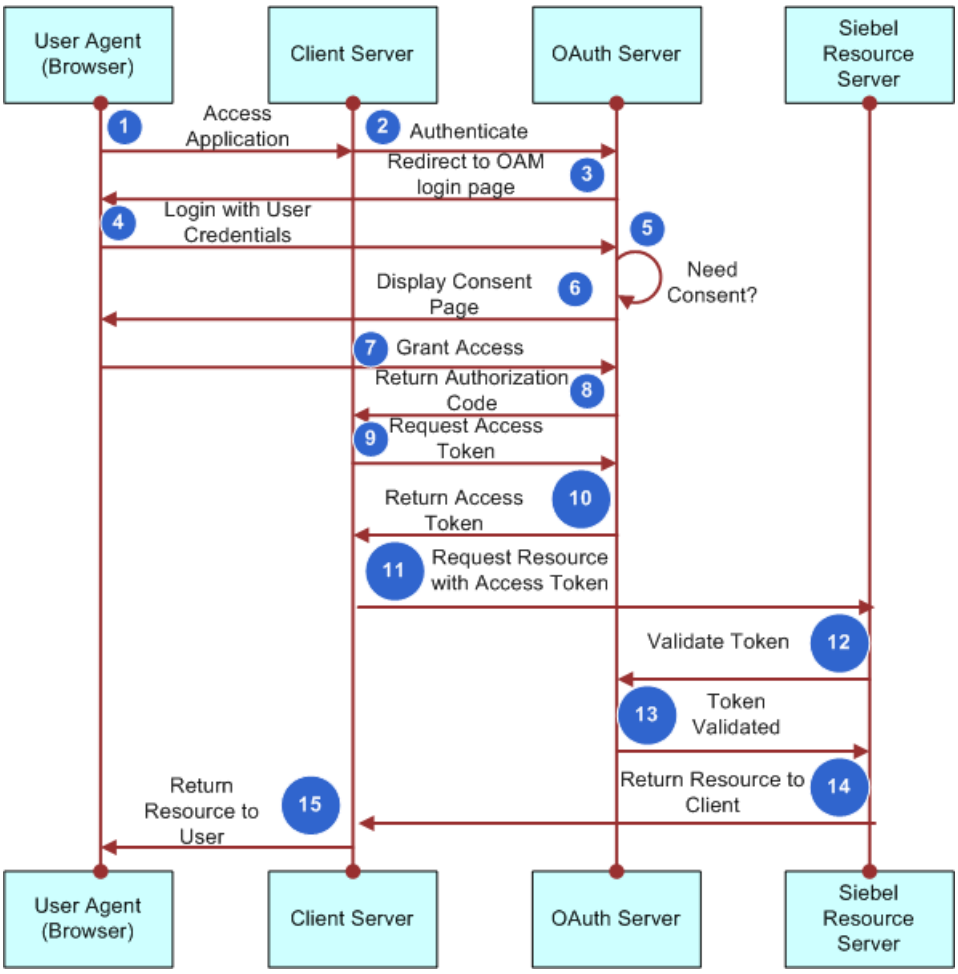

Figure 3. Web Server Authentication Flow

The steps in Web Server Authentication Flow process shown in Figure 3 are:

1 A user initiates a request to the client application.

2 The request is redirected to the OAUTH server (for example, Oracle Access Manager) for authentication.

3 The OAuth server sends a login form to the user to grant access.

4 The user enters their credentials and the OAuth server determines whether to grant access.

5 The OAuth server determines if user consent is required.

6 If user consent is the required, the OAuth server sends a consent form to the user.

7 The client server grants access.

8 If access is granted, the OAuth server sends an authorization code.

9 The client application requests an access token from the OAuth server.

10 The OAuth server sends the access token.

11 The client application sends a request to the resource server and includes the access token in the request header.

12 The resource server sends a request to the OAuth server to validate the access token.

13 The OAuth server validates the access token.

14 The resource server sends a response with the requested information to the client application.

15 The client application sends the requested information to the user.

3 Getting Started with the Siebel REST API

This chapter provides an overview of how to get started with using the Siebel CRM REST API. It contains the following topics:

About Setting Up the Siebel CRM REST API

The Siebel REST API is implemented as part of the existing EAI component group, so enabling the EAI component group will enable the REST related components on the Siebel server. For more information about enabling component groups, see *System Administration Guide*.

Before you begin using the Siebel CRM REST API, you must perform the tasks described in this topic. Many of these tasks are described in the *Siebel Deployment Planning Guide*.

1. Review all documented hardware and software requirements.

   For more information, see the Certifications tab on My Oracle Support.

2. The latest version of Siebel Enterprise Server software must be installed.

   For more information about performing a new Siebel Enterprise Server software installation, see the *Siebel Installation Guide* for the operating system you are using.

3. Use the Siebel Server Configuration Wizard to set the Anonymous User Login Name and Anonymous User Password Siebel RESTful Server Settings. These settings are for passwords that will be used to login to the Siebel Server when no credentials are passed through the REST API request. For more information about the Anonymous User Login Name and Anonymous User Password Siebel RESTful Server Settings, see "Configuring the Siebel Java Web Container" on page 24.

4 Determine your sizing requirements.

   If more than one Java Web Container is required for your Siebel REST deployment, the DfltTasks parameter must be configured on the Siebel Rest Proxy Object Manager Component. By default, the DfltTasks parameter value of 1. For more information about configuring load balancing, see "Configuring the worker.properties File for Load Balancing" on page 27.

   Oracle recommends load balancing to distribute complex tasks among multiple Java Web Containers. For more information about load balancing, see *Siebel Deployment Planning Guide*.

5 Configure a reverse proxy.

   If other Siebel access channels such as User Interface Applications and SOAP Enterprise Application Integration need to be accessed through the same domain as the Siebel REST API, a reverse proxy must be configured to for the Siebel Web Server and the Java Web Container and present a unified domain for external clients to request.

   For more information about configuring a reverse proxy, see "Configuring the Reverse Proxy" on page 28.

6 Configure business service methods for RESTful access. For more information about configuring business service methods, see "Configuring Business Service Methods for RESTful Access" on page 28.

7 Configure integration objects for REST API data access. For more information about configuring integration objects, see "Configuring Integration Objects for REST API Data Access" on page 30.

8 Configure a Java subsystem for REST API repository access and enable the EAI component group. For more information about configuring a Java subsystem and enabling the EAI component group, see "Enabling REST Services" on page 30.

Configuring the Siebel Java Web Container

The Siebel REST Proxy Object Manager is a background Siebel Server Component that manages the Java Web Container node tasks, such as starting, stopping, or restarting the Java Web Container nodes.

Java Web Container Node configuration information is stored in the following file:

$SIEBEL_ROOT/javacontainer/javacontainer<n>/siebsrvr.properties file

Each Java Web Container instance has a separate siebsrvr.properties file.

Table 7 describe the configurable fields in the siebsrv.properties file.

Table 7. Java Web Container siebsrv.properties file

Property Name	Description	Default Value
GatewayName	The name of the Gateway Server configured on the Siebel Server.	The SiebelRestCompIntMgr reads the values set on the Siebel Server and writes the value into the siebsrvr.properties file.
SCBPort	The Siebel Connection Broker Port configured on the Siebel Server.	The SiebelRestCompIntMgr reads the values set on the Siebel Server and writes the value into the siebsrvr.properties file.
Enterprise	The Enterprise name configured on the Siebel Server.	The SiebelRestCompIntMgr reads the values set on the Siebel Server and writes the value into the siebsrvr.properties file.
LogLevel	The Log level setting for log messages in the Java Web Container. Values include: ERROR, WARN, INFO, DETAIL, or DEBUG.	The default value is ERROR.
ObjMgr	The Object Manager that the Java Web Container should connect to.	The default value is eaiObjMgr_enu. The value can be changed to any Interactive Object Manager that can serve Siebel Server EAI requests.
Lang	The language of Object Manager specified in the ObjMgr property.	The default value is Enu. This value can be changed to any language, however, the value must match the value entered for the ObjMgr property.
AnonUser	The Siebel user that the Java Web Container simulates for Anonymous sign-in Requests.	None. This value is based on Siebel Installer inputs.
AnonPwd	The Encrypted Password for the Siebel user configured in the AnonUser property.	None. This value is based on Installer inputs.
BaseURL	The URL added to the REST response links.	The default value is: http://<hostname>/siebel-rest/v1.0/.

Table 7. Java Web Container siebsrv.properties file

Property Name	Description	Default Value
MaxConnections	The maximum percent of Object Manager Tasks to be used by a Java Web Container Node. This property determines the Maximum Number of Requests the Java Web Container Node can serve. Maximum Requests = (MaxConnections * MaxTasks/ MaxMTServers) / 100	The default value is 20. This value can be changed to a value range between 1-100. Invalid values will default to 20. The Maximum Requests value should arrive as an integer.
MinConnections	The minimum percent of Object Manager Tasks that are kept open in the Pool of Connections maintained by the Tomcat Node. Minimum Connection Pool Size = (MinConnections * MaxTasks/ MaxMTServers) / 100	The default Value is 25. This value can be changed to value range between 1-100. Invalid values will default to 25. The Minimum Connection Pool Size value should arrive as an integer. The Minimum Connection Pool Size value cannot be more than Maximum Requests.
MaxTasks	The maximum number of Object Manager tasks configured on the Java Web Container Node.	The Siebel REST Proxy Object Manager reads the values set on the Siebel Server and writes it to the siebsrvr.properties file.
MaxMTServers	The maximum number of Object Manager multithreaded processes configured on the Java Web Container Node.	The Siebel REST Proxy Object Manager reads the values set on the Siebel Server and writes it to the siebsrvr.properties file.
SessKeepAlive	The value of the SessKeepAlive property is the value of the Object Manager configured on the Java Web Container Node.	The Siebel REST Proxy Object Manager reads the values set on the Siebel Server and writes it to the siebsrvr.properties file.
AuthenticationType	The value of this property indicates the Authentication type. Values include: ■ Basic ■ SSO ■ OAuth	The default value is Basic.
TrustToken	If the Authentication Type value is SSO, then this is the TrustToken expected by Siebel Server for successful authentication.	None.

Table 7. Java Web Container siebsrv.properties file

Property Name	Description	Default Value
OAuthEndPoint	If the Authentication Type value is OAuth, then this is the URL of the OAuth Service Provider. For example: https://<OAuthServer>/api/oauth/tokeninfo	None.
SingleSignOn	The value of this property is either TRUE or FALSE. A TRUE value means AuthenticationType is SSO.	The default value is False.
UserSpec	The name of the variable on the HTTP Header that should be referenced by the Java Web Container to extract the User Name, in case the AuthenticationType value is SSO.	None.
UserSpecSource	This property informs the Java Web Container where to look for the User Name value if AuthenticationType value is SSO.	The default value is Header.
ValidateCertificate	The value is either True or False. This parameter is required to bypass the HTTPS secured channel. Oracle does not recommend setting the value for this parameter during production. Rather, you should set the value at the point when it is required to bypass the HTTPS secured channel.	The default value is True.

Configuring the worker.properties File for Load Balancing

This topic describes the initial configuration of the workers.properties file for load balancing. For more information about planning and managing load balancing for your deployment, see *Siebel Installation Guide* for the operating system you are using, *Siebel Deployment Planning Guide*, and *Siebel System Administration Guide*.

For more information about third-party load balancing options, see the Certifications tab on My Oracle Support.

The Apache HTTPD mod_jk Web server module is used to load balance Apache Tomcat servers. Oracle recommends for optimal usage of Siebel User Session Connections maintained on Tomcat, requests for the same user should be routed by the Load Balancer to the same Apache Tomcat Node every time by configuring Sticky Sessions on the Load Balancer.

To configure load balancing, you must update the workers.properties file. Oracle has provided a script along with documentation on how to use the script for generating the workers.properties file for Siebel REST API services. For more information, see the following link:

https://github.com/OracleSiebel/ConfiguringSiebel/tree/master/ExampleCode/REST/Load%20Balancing

You can generate the workers.properties file by executing the following command:

```
LoadBalancingConfig.pl <$SIEBEL_ROOT/javacontainer> <Apache HTTPd installation folder>
```

Configuring the Reverse Proxy

If the Siebel REST API, existing Siebel User Interface, and EAI SOAP services are to be exposed on the same domain as the Siebel REST API, then Oracle recommends that you configure a Reverse Proxy Web Server to route requests to the correct backend server.

The Reverse Proxy should route REST API traffic to the Java Web Container (for a single Web Container installation) or to the Apache HTTPD (or any other Load Balancer) configured for load balancing multiple Java Web Containers.

For more information about reverse proxy, see *Siebel Installation Guide* for the operating system you are using, *Siebel Deployment Planning Guide*, and *Siebel System Administration Guide*.

Configuring Business Service Methods for RESTful Access

You can use the Siebel REST API to access Siebel Business Services. Before you access the Siebel Business Services, you have to associate the business with a responsibility to control access to the business service and its methods. For more information about associating a business service with a responsibility, see the *Siebel Security Guide*.

Oracle recommends that you do not add responsibilities to the EAI Siebel Adapter Business Service. as it may lead to denial of general REST (repository or data) and SOAP Web Services access to other users.

The REST Service API has more restrictive access for Siebel Business Services than other Siebel Access Channels. Each Business Service and Business Service Method that is accessed through the REST API needs to be explicitly configured for access.

NOTE: For the Siebel REST API, if a Business Service Method is not configured for access by a particular responsibility, then it will not be accessible. This is different from SOAP or the User Interface channels where an unconfigured Business Service is accessible to all.

To configure Business Service Methods for RESTful Access

1. Log in as an administrator.
2. Navigate to the Administration - Application screen, then the Business Service Access view, and then the Access By Responsibility view.
3. In the Business Service list, click New to select a business service.

   A new record appears in the Business Service list.
4. Click the Select button in the Name field.

   The Business Service dialog box appears.
5. Select the business service to which you want to control access, then click OK.

   The selected business service appears in the Business Service list view.
6. In the Access By Responsibility list view, click New.

   The Add Responsibilities dialog box appears.
7. Select a responsibility to associate with the business service and then click OK.

   The selected responsibility appears in the Access By Responsibility list view.
8. In the Business Service Method list, click New to specify the business service methods to which the responsibility gains access.

   The Business Service Method dialog box appears. This dialog box displays the list of business service methods to which access is currently controlled.
9. If the business service method to which you want to allow the responsibility access appears in the Business Service Method dialog box, select it, then click OK.
10. Click the Select button in the Name field.

    The Business Service Method dialog box appears.
11. Select a business service method to associate with the responsibility and then click OK.

    The selected business service method appears in the Business Service Method list view.
12. From the Broadest Visibility list, select the view mode to associate with the responsibility.
13. Step off the record to save changes.
14. Click Clear Cache.
15. Start the Siebel Rest Proxy Object Manager with the following commands:

    ```
    stop comp SiebelRestCompIntMgr

    Start comp SiebelRestCompIntMgr
    ```

Configuring Integration Objects for REST API Data Access

An additional Business Object may be exposed through the Siebel REST API by creating an Integration Object for the Business Object, Base <Business Object Name>. Each Integration Component in the Integration Object should have an Integration Component Key, REST ROWID User Key:1, with one Integration Component Key Field Id defined. Without this Integration Component Key, the REST API upsert requests will not work on the Integration Component. For example:

IO Name = Base* & IC Key "REST ROWID User Key:1" with IC Key Field "Id"

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

Enabling REST Services

Enabling the EAI component group also enables the REST API components on the Siebel CRM Server.

For the Siebel business object and business service APIs, the REST API is ready to use as soon as the EAI component group is enabled. If you need more than one Java Web Container configured or a reverse proxy configured, you will need to configure them before enabling the EAI component group. For more information about configuring load balancing, see "Configuring the worker.properties File for Load Balancing" on page 27. For more information about configuring a reverse proxy, see "Configuring the Reverse Proxy" on page 28.

For more information about enabling EAI Component Groups, see the *Siebel System Administration Guide*.

If you plan to use the Siebel repository API, you will also need to create a JVM subsystem to implement a Java Business Service. For more information, see *Transports and Interfaces: Siebel Enterprise Application Integration*.

To enable the REST API

1. Open Siebel Sever Manager command-line interface and enable the EAI Component group with the following command:

   enable compgrp EAI

2. If you plan to use the Siebel repository API, open Siebel Server Manager command-line interface and create a named JVM subsystem, JAVA, with the following command:

   create named subsystem JAVA for subsystem jvmsubsys

3. Use the Siebel Sever Manager command-line interface to update the Profile Parameters.

   a. Set the Value of the JVM DLL Name parameter to the path where you have the jvm.dll file installed.

      Change param DLL=<*Full Path of jvm dll*>

b. Set the Value of the JVM Classpath parameter to the location of the .jar file with the following:

   CLASSPATH=<*full path to Siebel.jar*>;<*full path to SiebelJI_enu.jar*> for named subsystem JAVA

c. Set the Value of the JVM Options record to any JVM-specific options that you would like to enable:

   VMOPTIONS="-Xdebug -Xrunjdwp:transport=dt_socket,address=8000,server=y,suspend=n" for named subsystem JAVA

4. Restart the Siebel Server to make the new subsystem available.

Configuring Siebel Clinical Users

You can use Siebel Tools to configure Siebel Clinical users to pass default position and responsibility information in the Siebel REST response. For more information about Siebel Clinical, see *Siebel Clinical Trial Management System Guide*.

To configure Siebel Clinical Users

1. Log into Siebel Tools as an administrator.
2. Navigate to the Administration - Application screen and then the Business Service view.
3. In the Business Service list, select the Clinical User Provisioning Service business service.
4. In the Business Service Method list, select the CreateUser method.

   The selected business service method appears in the Business Service Method list view.
5. In the Business Service Method Arguments list, select Default Position.

   NOTE: The Default Position must belong to the Default Organization.
6. In the Business Service Method Arguments list, select Default Responsibility.
7. Step off the record to save changes.
8. Compile the Siebel repository file (SRF) after the changes and deploy the SRF file to the Siebel server(s).

4 Using the Siebel REST API

This chapter describes the Siebel REST API requests and responses for REST API calls to access Siebel CRM resources. It includes the following topics.

About Using the Siebel REST API

Each topic in this chapter provides examples demonstrating how to use the Siebel REST API calls to interact with Siebel Server resources.

Each REST API call in this chapter uses the following format:

- An example request, which includes the following information:
  - **URI.** The location of the Siebel REST API resource on the Siebel Server. For more information about Siebel REST API URL format, see "About Siebel CRM REST API URI Formats" on page 11.
  - **HTTP Method.** The HTTP method used to call the Siebel REST API to interact with the Siebel Server. For more information about supported HTTP Methods, "About Supported HTTP Methods" on page 14.
  - **Content-Type.** The part of the HTTP header that indicates the media type of the data that is sent by the Siebel REST API HTTP methods. For more information about supported HTTP headers, see "About Supported HTTP Header Fields" on page 14.
  - **Request Body.** The code example for the Siebel REST API request.
- An example response, which includes the following information:
  - **HTTP Code.** The HTTP status code returned to indicate whether the request was successful or if there was an error. For more information about supported HTTP codes, "About Standard HTTP Status Codes and Error Messages" on page 15.
  - **Content-Type.** The part of the HTTP header that indicates the media type of the data that is returned by the Siebel REST API HTTP methods. For more information about supported HTTP headers, see "About Supported HTTP Header Fields" on page 14.
  - **Response Body.** The code example for the Siebel REST API response.

Using the Siebel REST API to Access Siebel Repository Resources

You can use the Siebel REST API to access Siebel CRM repository resources. Users can perform Query, Insert, Update and Delete operations on the Siebel CRM repository resources (such as account or contacts) using REST API requests over HTTP as described in this section.

Querying for a Siebel CRM Parent Repository Resource

You can query for a Siebel CRM parent repository resource by sending an HTTP GET request to the repository resource's URI.

The following details are for a request query for a parent repository resource on the Siebel CRM Server:

- URI: http://*ServerName:port*/siebel-rest/v1.0/workspace/MyWorkspace/SIS Account List Applet
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic
- Request body: None

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type: application/json
- Response body:

```
{
  "Link": [
    {
      "rel": "self",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
       SIS Account List Applet",
      "name": "Applet"
    },
    {
      "rel": "canonical",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
      SIS Account List Applet",
      "name": "Applet"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
      SIS Account List Applet/Applet Message",
      "name": "Applet Message"
    },
    {
```

```
  "rel": "child",
  "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
  SIS Account List Applet/List",
  "name": "List"
},
{
  "rel": "child",
  "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
   SIS Account List Applet/Chart",
  "name": "Chart"
},
{
  "rel": "child",
  "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
  SIS Account List Applet/Tree",
  "name": "Tree"
},
{
  "rel": "child",
  "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
  SIS Account List Applet/Applet Web Template",
  "name": "Applet Web Template"
},
{
  "rel": "child",
  "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
  SIS Account List Applet/Applet Browser Script",
  "name": "Applet Browser Script"
},
{
  "rel": "child",
  "href": "http://slc05krw/siebel-rest/v1.0/workspace/MyWorkspace/Applet/SIS
   Account List Applet/Applet User Prop",
  "name": "Applet User Prop"
},
{
  "rel": "child",
  "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
  SIS Account List Applet/Applet Locale",
  "name": "Applet Locale"
},
{
  "rel": "child",
  "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
  SIS Account List Applet/Applet Toggle",
  "name": "Applet Toggle"
},
{
  "rel": "child",
  "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
  SIS Account List Applet/Applet Server Script",
  "name": "Applet Server Script"
},
{
```

```
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
      SIS Account List Applet/Applet Script",
      "name": "Applet Script"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
      SIS Account List Applet/Drilldown Object",
      "name": "Drilldown Object"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
      SIS Account List Applet/Control",
      "name": "Control"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
      SIS Account List Applet/Applet Method Menu Item",
      "name": "Applet Method Menu Item"
    }
  ],
    "ObjectLockedById": "",
    "AutoQueryMode": "",
    "PopupDimension": "",
    "Class": "CSSFrameListBase",
    "NoInsert": "N",
    "HelpIdentifier": "ID_APPLET_ACCOUNT_LIST",
    "ProjectLocked": "Y",
    "MailAddressField": "",
    "Name": "SIS Account List Applet",
    "BrowserClass": "",
    "ICLUpgradePath": "",
    "ProjectUIFreezeById": "",
    "HTMLPopupDimension": "",
    "workspace/MyWorkspaceName": "Siebel workspace/MyWorkspace",
    "MailTemplate": "",
    "BusinessComponent": "Account",
    "HTMLNumberOfRows": "",
    "ProjectLockedLanguage": "ENU",
    "Title-StringReference": "SBL_ACCOUNTS-1009090115-23Y",
    "Title-BaseRow": "",
    "ApplicationCode": "",
    "SearchSpecification": "",
    "Scripted": "N",
    "ResponsiveFlag": "Y",
    "ServiceApplicationUse": "",
    "TextStyle": "",
    "BackgroundBitmap": "",
    "NoUpdate": "N",
    "workspace/MyWorkspaceId": "88-4NW-1",
    "ObjectLocked": "N",
```

```
    "Title-RefDefault": "Accounts",
    "DefaultAppletMethod": "",
    "ToolsApplicationUse": "",
    "DefaultDoubleClickMethod": "",
    "Height": "4",
    "NoDelete": "N",
    "Module": "",
    "ObjectLockedDate": "",
    "ObjectLanguageLocked": "ENU",
    "Comments": "7.5.2.200 UI ENHANCEMENT",
    "Task": "",
    "Title-RefExact": "Accounts",
    "UpgradeAncestor": "",
    "ProjectLockedById": "0-1",
    "ProjectId": "88-4NW-4Y3",
    "InsertApplet": "",
    "BackgroundBitmapStyle": "",
    "DisableDatalossWarning": "N",
    "ProjectUIFreeze": "N",
    "InsertPosition": "",
    "SalesApplicationUse": "",
    "Width": "2",
    "ObjectLockedByName": "",
    "Inactive": "N",
    "UpgradeBehavior": "Preserve",
    "Type": "Standard",
    "LanguageCode": "",
    "AssociateApplet": "Agg Account Assoc Applet",
    "Title-StringOverride": "",
    "ProjectName": "VERT CUT Common",
    "ListName": "List",
    "NoMerge": "N"
}
```


Inserting a Siebel CRM Child Repository Resource

You can insert a Siebel CRM child repository resources by sending an HTTP PUT request to the repository resource's URI.

The following details are for a request query for a child repository resource on the Siebel CRM Server:

- URI: http://*ServerName:port*/siebel-rest/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet_1/Control/WriteRecord
- HTTP Method: PUT
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
  "Name":"WriteRecord"
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type: application/json
- Response body:

```
{
  "workspace/MyWorkspaceApplet": {
    "Link": {
      "rel": "self",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
      SIS Account List Applet_1",
      "name": "Applet"
    },
    "Name": "SIS Account List Applet_1"
  }
}
```

Upserting a Siebel CRM Parent Repository Resource

You can upsert a Siebel CRM parent repository resource by sending an HTTP PUT request to the resource's URI.

The following details are for a request to upsert a parent repository resource on the Siebel CRM Server:

- URL: http://*ServerName:port*/siebel-rest/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet_1
- HTTP Method: PUT
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
    "Name": "SIS Account List Applet_1",
    "UpgradeBehavior": "Preserve",
    "Comments": "New Record_Applet",
    "ProjectName": "VERT CUT Common"
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type:
- Response body:

```
{
  "workspace/MyWorkspaceApplet": {
    "Link": {
      "rel": "self",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
       SIS Account List Applet_1",
      "name": "Applet"
      },
      "Name": "SIS Account List Applet_1",
      "UpgradeBehavior": "Preserve",
      "ProjectName": "VERT CUT Common",
      "Comments": "New Record_Applet"
      }
}
```

Upserting a Siebel CRM Child Repository Resource

You can upsert a Siebel CRM child repository resource by sending an HTTP PUT request to the resource's URI.

The following details are for a request to upsert a child repository resource on the Siebel CRM Server:

- URI: http://*ServerName:port*/siebel-rest/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet_1/Control/WriteRecord
- HTTP Method: PUT
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
    "Name": "WriteRecord",
    "Comments": "New Record_Applet",
    "ProjectName": "VERT CUT Common"
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type:
- Response body:

```
{
  "workspace/MyWorkspaceApplet": {
    "Link": {
      "rel": "self",
      "href": "http://ServerName:port/siebel-rest/v1.0/workspace/MyWorkspace/Applet/
      SIS Account List Applet_1",
      "name": "Applet"
    },
```

```
    "Name": "SIS Account List Applet_1"
  }
}
```

Deleting a Siebel CRM Repository Resource

You can delete a Siebel CRM repository resource by sending an HTTP Delete request to the resource's URI.

The following details are for a request to delete a repository resource on the Siebel CRM Server:

- URI: http://*ServerName:port*/siebel-rest/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet_1
- HTTP Method: DELETE
- Content-Type: application/json
- Authorization: Basic
- Request body: None

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type:
- Response body: None

Using the Siebel REST API to Access Siebel Business Objects

You can use the Siebel REST API to access Siebel CRM Business Objects. Users can perform Query, Insert, update and Delete operations on the Siebel Business Objects using REST API requests over HTTP as described in this section.

Querying for a Siebel CRM Parent Business Object

You can query for a Siebel CRM parent Business Object by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a business object on the Siebel CRM Server:

- URL: http://*ServerName:port*/siebel-rest/v1.0/data/Account/Account/88-4XVPD
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic

- Request body: None

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type:
- Response body:


```
{
  "Link": [
    {
      "rel": "self",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD",
      "name": "Account"
    },
    {
      "rel": "canonical",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD",
      "name": "Account"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD/
      UT Account Partner",
      "name": "UT Account Partner"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD/
      TAF Product Catalog Product Category",
      "name": "TAF Product Catalog Product Category"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD/
      TAF Assortment Plan2",
      "name": "TAF Assortment Plan2"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD/
      Revenue Plan",
      "name": "Revenue Plan"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/ Account/88-4XVPD/
      Retail Product",
      "name": "Retail Product"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD/
```

```
      Retail Audit Product",
      "name": "Retail Audit Product"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD/
      RTD Retention Actions (B2B)",
      "name": "RTD Retention Actions (B2B)"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD/
      Pharma Account Call - CE",
      "name": "Pharma Account Call - CE"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD/
      My Account Offer",
      "name": "My Account Offer"
    },
    {
      "rel": "child",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD/
      FINCORP Subsidiary",
      "name": "FINCORP Subsidiary"
    },

...

],
    "SBA Review": "",
    "Friday End Time 2": "",
    "Saturday End Time 2": "",
    "Modified By Name": "SADMIN",
    "Friday End Time 1": "",
    "Saturday End Time 1": "",
    "Our Position": "",
    "Supply Characteristics": "",
    "DeDup Key Modification Date": "",
    "Prospect Flag": "N",
    "Market Share": "",
    "Statement Date": "",
    "Reference Stage": "",
    "Alt Email Address Type": "",
    "Business Description": "",
    "Fin Acct - Approx Annual Sales": "",
    "Start Date": "",
    "Contact Id": "",
    "Price List End Date": "",
    "Disable DataCleansing": "N",
    "Experian Id": "",
    "Preferred Shipping Method": "",
    "Project Fix Fee": "",
```

```
...
}
```

Upserting Siebel CRM Parent Business Object

You can upsert a Siebel CRM parent business object by sending an HTTP PUT request to the resource's URI.

The following details are for a request to upsert a parent business object on the Siebel CRM Server:

- URL: http://*ServerName:port*/siebel-rest/v1.0/data/Account/Account/88-4XVPD
- HTTP Method: PUT
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
    "Name": "AccountExample",
    "Primary Organization Id": "1SIA-7SY3",
    "Primary Organization": "Medical Products MD ENU",
    "Location": "Albany",
    "Description": "AccountData"
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type: application/json
- Response body:

```
{
  "items": {
    "Link": {
      "rel": "self",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD",
      "name": "Account"
    },
    "Name": "AccountExample",
    "Id": "88-4XVPD",
    "Location": "Albany",
    "Primary Organization Id": "1SIA-7SY3",
    "Primary Organization": "Medical Products MD ENU",
    "Description": "AccountData"
    }
}
```

Inserting a Siebel CRM Parent Business Object

You can insert a Siebel CRM parent business object by sending an HTTP POST request to the resource's URI.

The following details are for a request to insert a parent business object on the Siebel CRM Server:

- URL: http://*ServerName:port*/siebel-rest/v1.0/data/Account/Account
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
    "Name": "AccountExample",
    "Primary Organization": "Medical Products MD ENU",
    "Location": "Albany",
    "Primary Organization Id": "1SIA-7SY3",
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type: application/json
- Response body:

```
{
  "items": {
    "Link": {
      "rel": "self",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD",
      "name": "Account"
    },
    "Name": "AccountExample",
    "Id": "88-4XVPD",
    "Location": "Albany",
    "Primary Organization Id": "1SIA-7SY3",
    "Primary Organization": "Medical Products MD ENU"
  }
}
```

Inserting a Siebel CRM Child Business Object

You can insert a Siebel CRM child business object by sending an HTTP PUT request to the resource's URI.

The following details are for a request to insert a child business object on the Siebel CRM Server:

- URL: http://*ServerName:port*/siebel-rest/v1.0/data/Account/Account/88-4XVPD/Contact

- HTTP Method: PUT
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
    "Bill To First Name":"Siebel",
    "Bill To Last Name": "Customer",
    "Person UId":"0CR-1MF5Z6d11",
    "Primary Organization":"Default Organization",
    "Personal Contact":"N"
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type:
- Response body:

```
{
  "items": {
    "Link": {
      "rel": "self",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD",
      "name": "Account"
    },
    "Id": "88-4XVPD"
    }
}
```

Upserting a Siebel CRM Child Business Object

You can upsert a Siebel CRM child business object by sending an HTTP PUT request to the resource's URI.

The following details are for a request to upsert a child business object on the Siebel CRM Server:

- URL: http://*ServerName:port*/siebel-rest/v1.0/data/Account/Account/88-4XVPD/Contact/88-4W6YS
- HTTP Method: PUT
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
    "Primary Account Name": "DemoAccount1303",
    "Personal Contact": "N",
    "Account Integration Id": "",
```

```
    "Id": "88-4W6YS",
    "Bill To Last Name": "Customer",
    "Person UId": "0CR-1MF5Z6d11",
    "Bill To First Name": "Siebel",
    "Employee Number": "88-4W6YS",
    "Primary Organization": "Default Organization",
    "Bill To Job Title": "VP"
}
```

The following are the details for the response to a successful request:

- HTTP Code: 201
- Content-Type:
- Response body:

```
{
  "items": {
    "Link": {
      "rel": "self",
      "href": "http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-4XVPD",
      "name": "Account"
    },
    "Id": "88-4XVPD"
  }
}
```

Deleting a Siebel CRM Business Object

You can delete a Siebel CRM business object by sending an HTTP DELETE request to the resource's URI.

The following details are for a request to delete a business object on the Siebel CRM Server:

- URL: http://*ServerName:port*/siebel-rest/v1.0/data/Account/Account/88-4XVPD
- HTTP Method: DELETE
- Content-Type: application/json
- Authorization: Basic
- Request body:

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type:
- Response body:

Using the Siebel REST API to Access Siebel Business Services

You can use the Siebel REST API to access Siebel Business Services. Users can call Siebel business services using the Siebel REST API HTTP POST request to specify the Business Service Name, Method name in the URI and Method parameters.

Before you can access the Siebel Business Services, you must configure the access and responsibility values of the Siebel Business Service. For more information, see "Configuring Business Service Methods for RESTful Access" on page 28.

Querying for a Siebel CRM Business Service

You can query for a Siebel CRM business service by sending an HTTP POST request to the resource's URI. When you query for a Siebel CRM business service, you must include the business service argument values in URI and the query parameters in the request body.

The following details are for a request to query for a business service on the Siebel CRM Server:

- URI: http://*ServerName:port*/siebel-rest/v1.0/service/Siebel Account/QueryByExample?PageSize=2&ViewMode=All
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
  "body":{
    "StartRowNum": "0",
      "SiebelMessage":{
        "MessageId":"",
        "MessageType":"Integration Object",
        "IntObjectName":"Account Interface",
        "IntObjectFormat":"Siebel Hierarchical",
        "ListOfAccount Interface":{
        "Account":{"Name": "3Com" }
        }
     }
   }
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type: application/json
- Response body:

```
{
  "items": {
    "Business Address": {
      "Address Id": "1-6NMY+C",
      "Street Address": "2000 West Embarcadero Rd",
      "Address Integration Id": "",
      "County": "",
      "Street Address 2": "",
      "City": "Palo Alto",
      "State": "CA",
      "Country": "USA",
      "Postal Code": "94510",
      "Province": "",
      "IsPrimaryMVG": "Y"
      },
     "Related Sales Rep": [
     {
      "IsPrimaryMVG": "Y",
      "Division": "IF 20 Commercial Banking",
      "Position Id": "1-1JD",
      "Position": "IF 20 33 Relationship Manager",
      "Position Integration Id": "",
      "Login": "FINADMIN"
       },
    {
      "IsPrimaryMVG": "N",
      "Division": "Siebel Administration",
      "Position Id": "0-5220",
      "Position": "Siebel Administrator",
      "Position Integration Id": "",
      "Login": "SADMIN"
      }
    ],
    "Related Contact": [
    {
      "Contact Id": "88-23SUX",
      "First Name": "Selfservice",
      "Contact Integration Id": "",
      "Middle Name": "",
      "Last Name": "Merchant",
      "Primary Organization": "Comms-Media Default Organization (COM ENU)",
      "Person UId": "88-23SUX"
      },
    {
      "Contact Id": "1-6S9M",
      "First Name": "Avo",
      "Contact Integration Id": "",
      "Middle Name": "",
      "Last Name": "Sarkissian",
      "Primary Organization": "Default Organization",
      "Person UId": "1-6S9M"
      },
    {
      "Contact Id": "1-BAEL",
```

```
    "First Name": "Todd",
    "Contact Integration Id": "",
    "Middle Name": "",
    "Last Name": "What",
    "Primary Organization": "Default Organization",
    "Person UId": "1-BAEL"
  },
  {
    "Contact Id": "1-BAER",
    "First Name": "Walter",
    "Contact Integration Id": "",
    "Middle Name": "",
    "Last Name": "What",
    "Primary Organization": "Default Organization",
    "Person UId": "1-BAER"
    }
  ],
  "Related Organization": {
    "IsPrimaryMVG": "Y",
    "Organization": "Millennium Institutional Finance Services IF ENU",
    "Organization Id": "1-1DG",
    "Organization Integration Id": ""
    },
    "Account Id": "1-63Q9",
    "Account Status": "Active",
    "Parent Account Integration Id": "",
    "Number of Employees": "0",
    "PO Auto Approval Date": "",
    "Name": "3Com",
    "PO Approved Flag": "",
    "Integration Id": "",
    "VAT registration number": "",
    "Assignment Country Code": "",
    "Parent Account Id": "",
    "Skip Credit Check": "N",
    "Price List": "",
    "Price List Id": "",
    "Price List Integration Id": "",
    "PO Auto Approval Limit": "",
    "Main Fax Number": "6505551212",
    "Assignment Area Code": "",
    "Alias": "",
    "Competitor Flag": "N",
    "Global Ultimate DUNS": "",
    "PO Auto Approval Currency Code": "",
    "Parent HQ DUNS": "",
    "Parent Account Location": "",
    "DUNS Number": "",
    "Credit Auto Approval Limit": "0",
    "Main Phone Number": "6505551212",
    "Parent Account Name": "",
    "Location": "HQ-Distribution",
    "Expertise": "",
    "Primary Organization": "Millennium Institutional Finance Services IF ENU",
```

```
      "Type": "Customer",
      "Home Page": "3COM.com",
      "Domestic Ultimate DUNS": "",
      "Credit Status Date": "",
      "Currency Code": "USD",
      "Partner Flag": "N"
    }
}
```

Using the Siebel REST API with Siebel Clinical

You can use the Siebel REST API to create, synchronize, and delete Siebel Clinical users. For more information about Siebel Clinical, see *Siebel Clinical Trial Management System Guide*.

You can also use the Siebel REST API to invoke business services using Siebel server script techniques. For more information, see *Using Siebel Tools*.

When you create or synchronize Siebel Clinical users, you can pass default position and responsibility information in the Siebel REST API response by configuring the LS Clinical User Provisioning Service Business method. For more information, see "Configuring Siebel Clinical Users" on page 31.

Creating a Siebel Clinical User

The following details are for a request to create a Siebel clinical user on the Siebel CRM Server:

- URI:http://*ServerName:port*/siebel-rest/v1.0/service/LS Clinical User Provisioning Service/CreateUser
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
  "body":
    {
      "Employee":
        {
           "First Name": "Cathy",
           "Last Name": "Rogers",
           "Login Name":"Cathy.Rogers",
           "EMail Addr":"Cathy.Rogers@oracle.com"
         }
      "ListOfPosition":
        {
            "Position":
               {
                    "Name":"Cathy.Rogers",
```

```
                    "Division":"Default Organization"
                }
            },
          "ListOfEmployee_Responsibility":
            {
                "Employee_Responsibility":
                  {
                    "Responsibility":"Clinical Research Associate
                  }
              }
      }
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type: application/json
- Response body: None.

Synchronizing a Siebel Clinical Users

The following details are for a request to synchronize a Siebel clinical user on the Siebel CRM Server:

- URI: `http://`*`ServerName:port`*`/siebel-rest/v1.0/service/LS Clinical User Provisioning Service/SynchronizeUser`
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
  "body":
    {
      "Employee":
       {
         "Login Name": "Cathy.Rogers",
         "Field 1": "Value 1",
         "Field 2": "Value 2",
         ……..
       }
    }
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type: application/json

- Response body: None.

Deleting a Siebel Clinical User

When you use the Siebel REST API to delete a Siebel clinical user, the Siebel Clinical application performs a soft delete of the clinical user by stamping the Termination Date field with the Current Date value and removing the user's responsibilities.

The following details are for a request to delete a Siebel clinical user on the Siebel CRM Server:

- URI: `http://ServerName:port/siebel-rest/v1.0/service/LS Clinical User Provisioning Service/TerminateUser`
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
  "body":
    {
     "Employee":
       {
         "Login Name":"Cathy.Rogers",
        }
    }
}
```

The following are the details for the response to a successful request:

- HTTP Code: 200
- Content-Type: application/json
- Response body: None.

Index

W