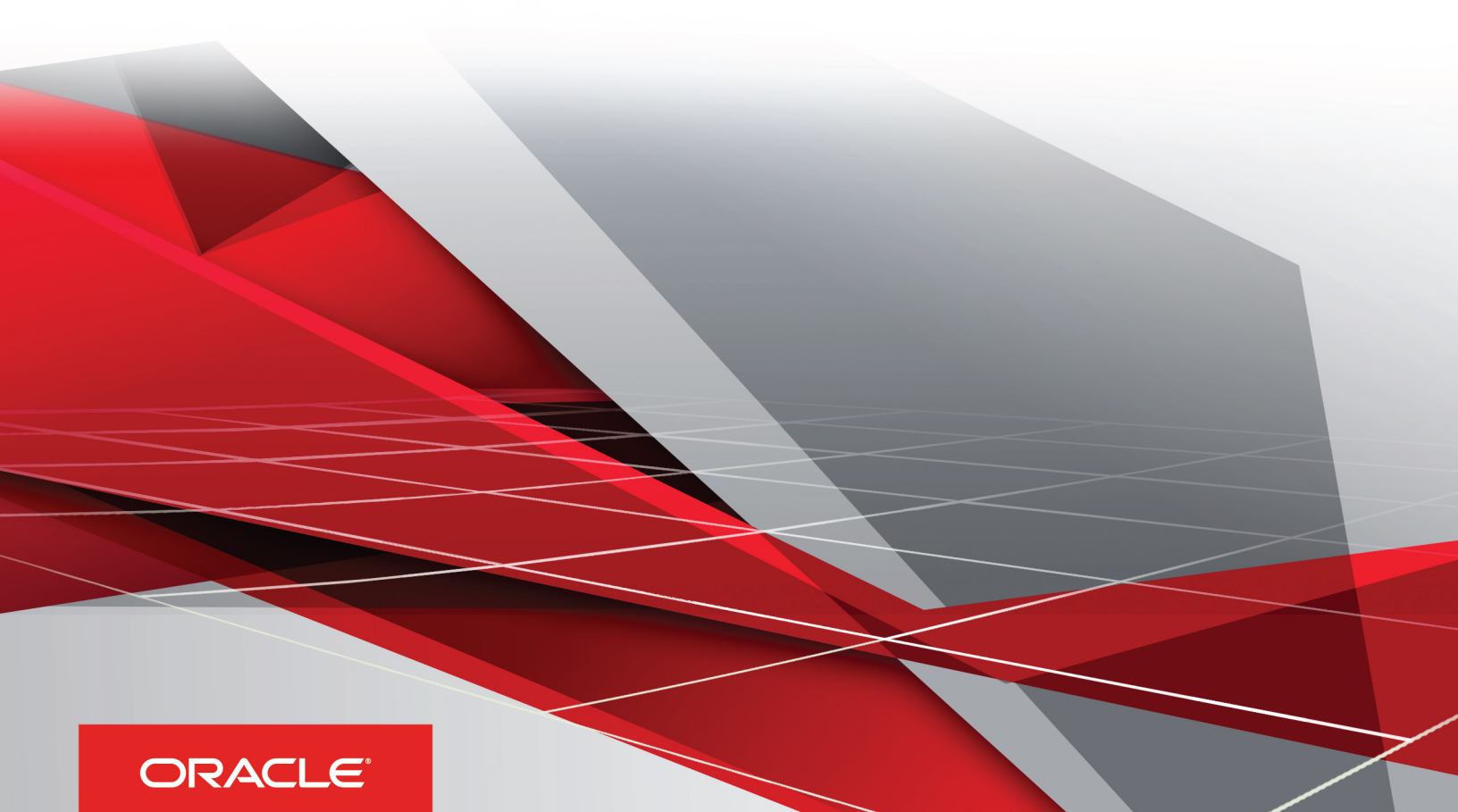
# Siebel
# REST API Guide

## Siebel 2018
## December 2018



Oracle Siebel
REST API Guide

December 2018

Part Number: F12234-01



Authors: Siebel Information Development Team

# Contents

# Preface

This preface introduces information sources that can help you use the application and this guide.

## Using Oracle Applications

To find guides for Oracle Applications, go to the Oracle Help Center at *http://docs.oracle.com/*.

## Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the *Oracle Accessibility Program website*.

## Contacting Oracle

### Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit *My Oracle Support* or visit *Accessible Oracle Support* if you are hearing impaired.

### Comments and Suggestions

Please give us feedback about Oracle Applications Help and guides! You can send an e-mail to: *oracle_fusion_applications_help_ww_grp@oracle.com*.

# 1 What's New in This Release

## What’s New in Siebel REST API Guide,Siebel CRM 18.12 Update

The following information lists the changes in this revision of the documentation to support this release of the software.

**Note:** Siebel 2018 is a continuation of the Siebel 8.1/8.2 release.

| Topic | Description |
|---|---|
| *About URI Parameters* | Modified topic. Added details about the **`uniformresponse`** flag. |
| *Using Siebel REST API to Access Siebel Business Objects JSON Examples* | Modified topic. Added five cases for querying with and without the **`uniformresponse`** flag. |

## What’s New in Siebel REST API Guide,Siebel CRM 18.7 Update

The following information lists the changes described in this version of the documentation to support this release of the software.

| Topic | Description |
|---|---|
| *About Getting the Siebel REST API Specification in the Open API 2.0 Standard Using Describe* | New topic. This topic provides information about the OpenAPI describe parameter. |
| *About Access Controls for Siebel Business Component REST Requests* | New topic. This topic provides information about access controls for Siebel Business Component REST requests. |
| *About URI Parameters* | Updated topic. Added the following new parameters to *About URI Parameters*:<br>• describe<br>• ViewMode<br>• childlinks |
| *Using the Siebel REST API* | Updated topic. Added new examples. |

# What's New in Siebel REST API Guide,Siebel Innovation Pack 2017, Rev. A

The following information lists the changes described in this version of the documentation to support this release of the software.

| Topic | Description |
|---|---|
| *About REST Outbound* | Updated topic. This topic provides information on how to configure Siebel REST API for outbound communications. |
| *Creating an Outbound Web Service Based on an Open APICompliant JSON File* | New topic. This topic provides information on how to create an outbound Web Service based on an OpenAPI compliant JSON file. |

# What's New in Siebel REST API Guide,Siebel Innovation Pack 2017

The following information lists the changes described in this version of the documentation to support this release of the software.

| Topic | Description |
|---|---|
| *About Siebel CRM REST API Architecture* | Updated topic. Added information about the new REST API architecture for Version #.#. |
| *About URI Parameters* | Updated topic. Added the fields and searchspec parameters to *About URI Parameters*. |
| *About REST Outbound* | New topic. This topic provides information on how to configure Siebel REST API for outbound communications. |
| *Using Siebel Management Console to Configure a Siebel Application Interface Profile* | New topic. This topic provides information on the REST parameters that can be configured when you configure the Siebel Application Interface Profile. |
| *Using Siebel REST API to Access Siebel Repository Data XML Examples* | New topic. This topics provides XML examples that demonstrate how to use Siebel REST API to access repository data. |
| *Using Siebel REST API to Access Siebel CRM Business Objects XML Examples* | New topic. This topics provides XML examples that demonstrate how to use Siebel REST API to access business object data. |

| Topic | Description |
| --- | --- |
| *Using Siebel REST API to Access Siebel Business Services XML Examples* | New topic. This topics provides XML examples that demonstrate how to use Siebel REST API to access business service data. |

# 2 Overview of Using the Siebel REST API

## 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 provided with the Siebel Application Interface installation. For more information about installing Siebel Application Interface, see *Siebel Installation Guide for Microsoft Windows* . The Siebel REST API is enabled by configuring and deploying a Siebel Application Interface profile. The Siebel Application Interface profile can be deployed to multiple Siebel Application Interface nodes.

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

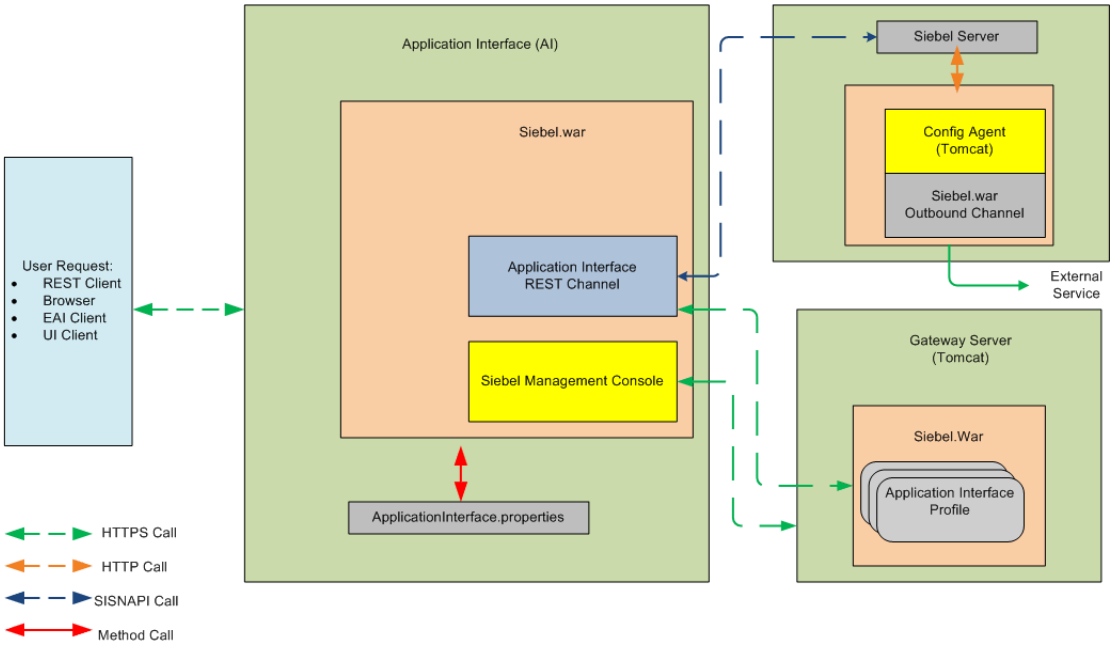
- A base URI to access Siebel Server resources, for example:
  **`http://Server Name:port/Siebel/v1.0/`**
- Support for JSON resource representations.
- Support for XML resource representations.
- Operations on Siebel CRM resources are mapped to semantically similar HTTP methods, such as GET, PUT, POST, and DELETE.
- Hypertext links to Siebel CRM Child Business Components in the case of the Data API resources in the REST API response.
- Support for Outbound to interact with other Cloud applications that communicate through REST.

# About Siebel CRM REST API Architecture

The Siebel CRM REST API services are deployed as a WAR file (siebel-rest.war) on the Siebel Application Interface. The Siebel Application Interface is a java-based Web component that runs on the Tomcat server. The Siebel Application Interface provides the Web entry point to Siebel Business Application services. The Siebel Application Interface interacts with the Siebel Server and Gateway Server to manage and fulfill Siebel REST requests.

Each Siebel Enterprise can have multiple Siebel Application Interface nodes and can have multiple Siebel Servers but can have only one Gateway Server. Installations of Siebel Enterprise Server and Siebel Application Interface modules include deployment of WAR files into the application container and configuration of application container ports.

The Siebel application configurations are managed by the Siebel Management Console and stored in the Siebel Gateway Server.



The preceding figure shows the high-level architecture of Siebel REST within the Siebel Application Interface.

- Requests: Users send an application request to the Siebel Application Interface. Siebel Application Interface requests include: UI, EAI, or REST. The appropriate application channel accepts the request. Application channels include: UI Channel, EAI Channel, or REST Channel.
- Siebel Application Interface: The Siebel Application Interface serves as the Web server for the Siebel Business Applications. The Siebel Application Interface identifies requests for Siebel application data coming from Web clients and flags these requests for routing to a Siebel Server. When information is sent from the Siebel Server back to the Web client, the Siebel Application Interface helps complete the composition of the Web page for forwarding to the client.
- Siebel Management Console: The Siebel Management Console is a Web-based application that allows users to configure various Siebel Application components and save the configurations in the Siebel Gateway Server.
- REST Channel: The REST channel is one of three application channels that provide an endpoint responsible for receiving and processing user requests. The REST Channel is responsible for serving the REST requests. Other application channels include the UI channel and EAI channel.
- Siebel Server: Siebel Server functions as an application server and is composed of server components.
- Gateway Server: Siebel Gateway provides the dynamic address registry for Siebel Servers and server components, and also for Siebel Application Interface. The Siebel Application Interface configurations are also stored and managed in the Gateway Server.
- Outbound Channel: The outbound channel forwards requests to external Web services.

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

# 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*.
  - 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*.
- 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*.
- 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*.

After the Siebel CRM Server processes the request, the server sends back a response in JSON or XML format based on the requested content type. 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*.

- 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*.
- 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*.

# 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:
  **`https://Server Name:port/Siebel/v1.0/data/`**
- Siebel CRM REST API basic URI for Siebel Business Services:
  **`https://Server Name:port/Siebel/v1.0/service/`**
- Siebel CRM REST API URI for a Siebel Repository resource:
  **`https://ServerName:port/Siebel/v1.0/workspace/<your workspace name>/`**

Where:

- Server Name:port: Indicates the name of the server and port hosting the Siebel REST API services.
- siebel: 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* .
- <your workspace name>: Indicates the name of your repository workspace.

# About URI Parameters

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

| Parameter | Description |
|---|---|
| PageSize | Used only for the GET operation. The PageSize parameter is the 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 |

| Parameter | Description |
|---|---|
| | restrict the number of records returned. You restrict the number of records returned by setting the PageSize parameter to a reasonable number. The default value is 10. If you do not set a value for this parameter, then query will return only 10 records.<br><br>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).<br><br>**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 only for the GET operation. The StartRowNum parameter is 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).<br><br>The default value and returns the records from beginning.<br><br>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. |
| fields | Used for only GET or Query operations to specify a comma-separated list of property names (fields) that are required in the REST API response. The response contains only the files given in this list irrespective of fields available in the source being queried. |
| searchspec | Used to include search specifications in the REST API response. |
| ViewMode | Used as an access control that controls users access to data and application functionality. Values include:<br><br>• Personal<br>• Sales Rep<br>• Organization<br>• Group<br>• Catalog<br>For more information about access controls, see *About Access Controls for Siebel Business Component REST Requests*. |
| childlinks | Used for only GET or Query operations to specify a comma-separated list of child business components that require links returned in the REST API response. The response returns only links to child objects specified in this parameter value.<br><br>For an example of childlinks, see *Querying for a Siebel CRM Repository Resource That Returns Child Links Only for Lists*. |
| uniformresponse | Used for only GET or Query operations to specify a consistent interface to the consumers and enable them to use a single parser for responses of requests of similar type. The response returns one or more records wrapped in an array. It is passed as a query parameter with a case insensitive value for the flag as y or Yes. For example, URI: **`data/Account/Account/?searchspec=([Location] LIKE 'HQ' AND [Account Status]='Active')&uniformresponse=y`** |

| Parameter | Description |
|---|---|
| | For examples of **uniformresponse**, see *Using Siebel REST API to Access Siebel Business Objects JSON Examples*. |

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 StartRowNum 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*. For more information about standard HTTP methods, see *About Supported HTTP Methods*.

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*.
- 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. Before you begin, a Workspace has to be created before performing any repository operations and the name of the workspace has be mentioned in the REST API requests. 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

The following table contains the HTTP methods supported by the Siebel REST API and the corresponding Siebel CRM operation.

| 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.

The following table contains the HTTP header fields supported by the Siebel REST API.

| HTTP Header Field Name | Description | Example |
|---|---|---|
| Authorization | The HTTP request header field that indicates the type of authorization. Options include:<br>• Basic, if the Authentication type configured in siebsrvr.properties is Basic or SSO<br>• 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. Content-Type decides the format of response for all requests.<br><br>The Siebel REST API supports JSON and XML encoding for the request body. The default value is application/JSON.<br><br>The Content-Type field is used with POST, PUT, and GET 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.<br><br>For GET requests, the content type is used to determine the format of response, either XML or JSON. | 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.

The following table contains the standard HTTP status codes used by the Siebel REST API.

| 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. |

| HTTP Code | Message | Description |
|---|---|---|
| 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 type. 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

Siebel CRM supports the following mechanisms that the client uses to authentication user credentials:

- Basic authentication over SSL. This is User ID and Password based authentication. The Base64 encoded value of the User ID and Password must be included in the Authorization header.
- OAuth user using OAuth 2.0. For more information about OAuth 2.0, see *About Configuring OAuth 2.0 for Authentication*.
- SSO, which can be either:
  - Pre-existing SSO mechanisms used for Siebel Applications or EAI.
  - SAML based SSO mechanisms. This is Identity Provider-Initiated Single Sign-On Authentication. For more information, see *Siebel Security Guide* .

Authentication parameters are configured in the Siebel Application Interface Profile. For information about REST Inbound Authentication parameters, see *Configuring REST Inbound Authentication Parameters*. For information about configuring the Siebel Application Interface Profile, see *Siebel Installation Guide for Microsoft Windows* .

# 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 Server in either SSO or OAuth, and does not require a 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.
- The following parameters must be set in the Siebel Application Interface profile:
  - SingleSignOn. The SingleSignOn property must be set to TRUE to implement SSO.
  - Authentication Type. The Authentication Type property must be set to OAuth to implement OAuth authentication

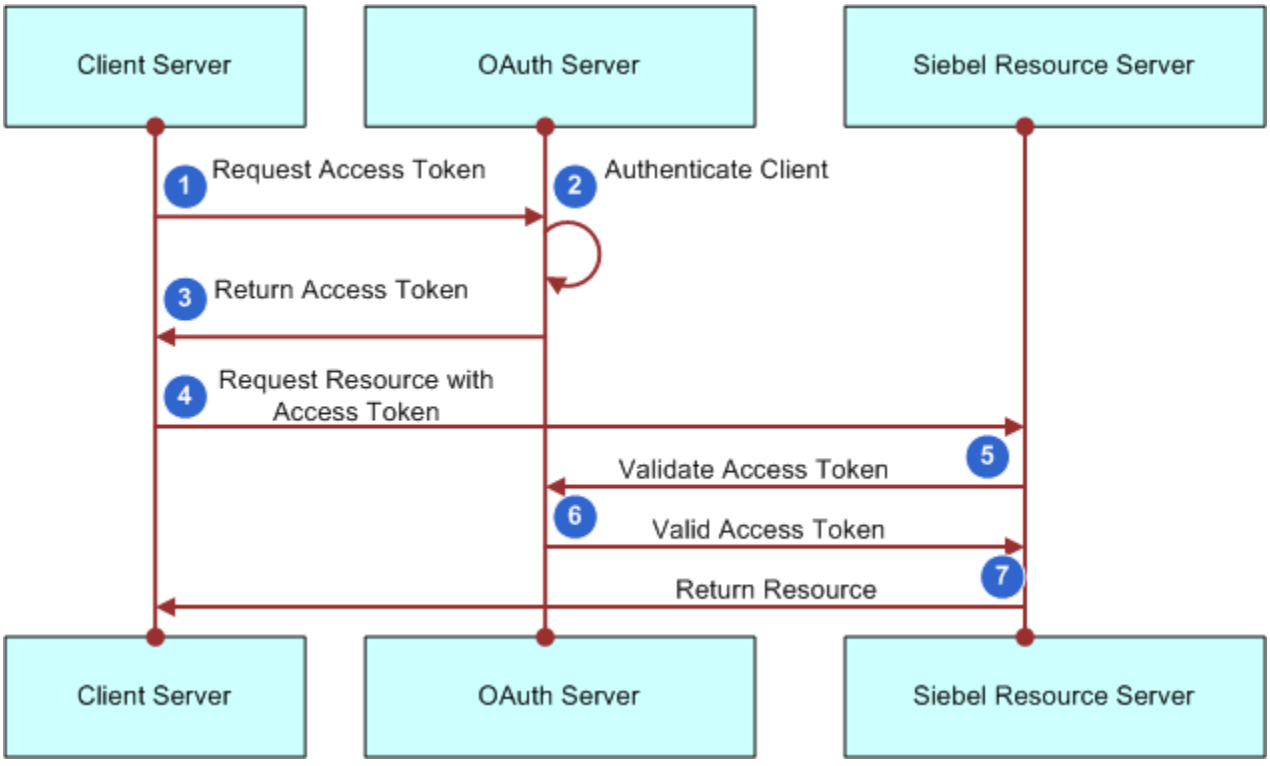
- Trust Token. The Trust Token value must be the same as the security adapter TrustToken parameter value.
- Authentication URL. The Authentication URL value is the URL of the OAuth Service Provider end point for Access Token validation.
- Secure Channel. Set this parameter only when you have already imported the Authentication URL's CA certificate into the Application Interface truststore. Deselect this check box when the Authentication URL's CA certificate is not available in the Application Interface truststore. In this case, the Application Interface trusts all certificates while calling the Authentication URL over HTTPS. Oracle does not recommend this.

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.

The following figure is an example of the Client Credentials Grant Authentication Flow.




The steps in client credentials grant authentication flow process shown in the preceding figure are:

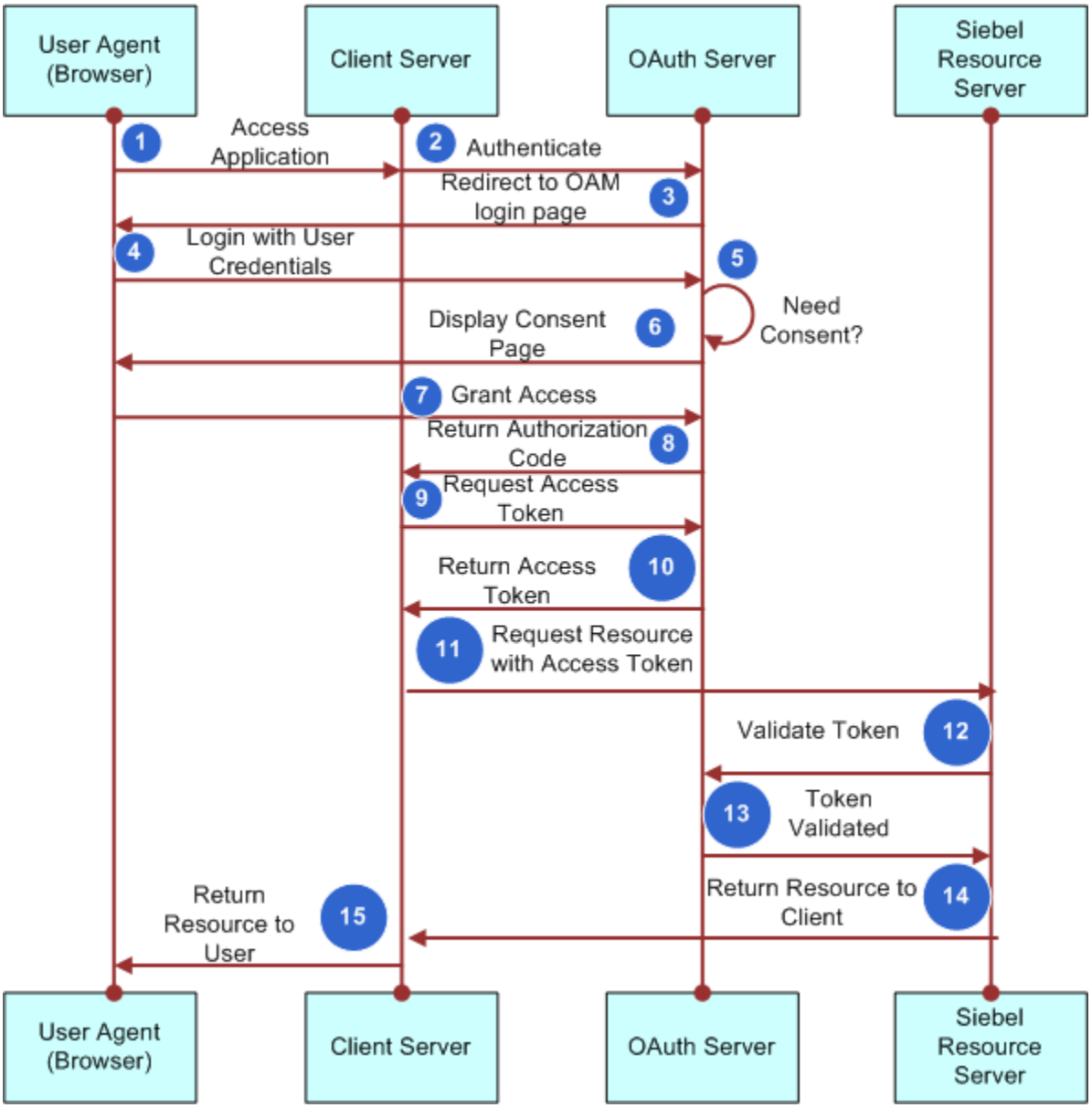
1. The business client application makes a call to the Siebel Server to request some business information by passing an access token. Since there is no end user intervention, the client is pre-authorized to have access to the resource.

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 Server looks for the 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.

The following figure is an example of the Web server Authentication Flow.




The steps in Web Server Authentication Flow process shown in the preceding figure 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.

# About REST Outbound

Siebel CRM supports outbound REST communications for on-premises or cloud applications, allowing the external applications to interact with other on-premises or cloud applications that communicate through REST APIs.

Siebel outbound REST functionality is built on SOAP Outbound functionality and allows Siebel CRM to call any external endpoint that provides a valid Open API or SWAGGER compliant schema.

Siebel outbound REST functionality leverages Siebel Tools to import the REST contract and create the required metadata, such as business services or integration objects. For more information about Siebel Tools, see *Using Siebel Tools* .

To use REST Outbound in Siebel CRM:

- Create a new outbound Web Service based on an OpenAPI compiant JSON file. For information about creating an Outbound Web Service based on an OpenAPI compliant JSON file, see *Creating an Outbound Web Service Based on an Open APICompliant JSON File*.
- Once you have successfully imported the schema in Siebel Tools, open the Siebel Application and navigate to the Administration - Web Services screen and then to the Outbound REST Services screen. Verify the newly imported artifacts are all present in the Outbound REST Services, Service Params, and Operations applets. Make sure you configure the exact endpoint URL in the Address column of the Service Params Applet.
- Create a 64-bit Java Subsystem. For more information about creating a 64-bit Java Subsystem, see *Transports and Interfaces: Siebel Enterprise Application Integration* .
- Configure any new Siebel Business Services for RESTful access. For more information, see *Configuring Business Service Methods for RESTful Access*.
- Configure new Siebel Business Services by associating a responsibility with a business service to control access to the business service and its methods. For more information, see *Siebel Security Guide* .
- Setting up the proxy for external endpoint access. For more information, see *Integration Platform Technologies: Siebel Enterprise Application Integration* .
- Configure external REST endpoints. For external REST endpoints, the following parameters must be added in the Input PropertySet:
    - isExternalURL: true

      Setting the isExternal parameter value to true removes the body keyword in the POST request body.
    - isEnclosedArray: true

      If the whole POST body has to be enclosed in any array, then set the isEnclosedArray parameter to true.
    - methodValue: <value>

      For path templating, add a value for the methodValue parameter. Path templating is the use of curly braces ({}) to mark a section of a URL path as replaceable.

# About Getting the Siebel REST API Specification in the Open API 2.0 Standard Using Describe

Siebel REST APIs use the OpenAPI Specification (formerly the Swagger Specification) to define formats for REST requests and responses to the Siebel REST API servers.

You can use the OpenAPI Specification Describe URI parameter to provide addition metadata information in the REST API response and request. The describe parameter returns a JSON object that contains the attributes, actions, and links defined in the REST resource definition.

The describe parameter allows you to discover additional metadata on the following:

- Siebel Business Objects. The Siebel Business Objects catalog contains a list of all Business Objects exposed as Base Integration Objects. The following is an example of a Siebel Business Objects URL request with a describe parameter: **host-name/siebel/v1.0/data/describe**
- Siebel Business Services. The Siebel Business Services catalog contains list of all Siebel Business Service names, methods defined for business services, and links to each business service. The following is an example of a Siebel Business Services URL request with a describe parameter: **host-name/siebel/v1.0/service/describe**
- Siebel Repository Objects. The Siebel Repository Objects catalog contains lists of all repository types and catalog links to their children. The following is an example of a Siebel Repository Objects URL request with a describe parameter: **host-name/siebel/v1.0/workspace/main/describe**

In the response, pagination is implemented for a list of Business Objects, a list of Repository Objects, and a list of Business Services but not for the associated children.

The following table contains the OpenAPI Objects supported by the Siebel REST API.

| Field Name | Description |
|---|---|
| swagger | Required. Specifies the Swagger Specification version being used. |
| info | Required. Provides metadata about the API. |
| schemes object | The transfer protocol of the API. Values MUST be from the list: "http", "https", "ws", "wss". If the schemes is not included, the default scheme to be used is the one used to access the Swagger definition itself. |
| securityDefinitions | Security scheme definitions that can be used across the specification. |
| externalDocs | Additional external documentation. |
| host | The host (name or ip) serving the API. This must be the host only and does not include the scheme nor sub-paths. It MAY include a port. If the host is not included, the host serving the documentation is to be used (including the port). The host does not support path templating. |
| basePath | The base path on which the API is served, which is relative to the host. If it is not included, the API is served directly under the host. The value MUST start with a leading slash (/). The basePath does not support path templating. |

| Field Name | Description |
|---|---|
| definitions | An object to hold data types produced and consumed by operations. |
| tags | A list of tags used by the specification with additional metadata. The order of the tags can be used to reflect on their order by the parsing tools. Not all tags that are used by the Operation Object must be declared. The tags that are not declared may be organized randomly or based on the tools' logic. Each tag name in the list must be unique. |
| paths | Required. The available paths and operations for the REST API. |
| security | A declaration of which security schemes are applied for the API as a whole. |
| parameters | A list of the parameters for the endpoint. |
| responses | A lists of the responses from the REST API request. |

# About Access Controls for Siebel Business Component REST Requests

Access Controls refer to the set of Siebel Business Applications mechanisms that control users access to data and application functionality. For more information about access controls, see *Siebel Security Guide* .

Siebel implements access controls for business component data by using the ViewMode query parameter. The business component ViewMode query parameter defines the access control for a business component in a view and decides which users can access what portion of the data. The following fields in the BusComp View Modes list in Siebel Tools determine allowable visibility for a business component. For more information about View Mode, see *Siebel Security Guide* .

- Owner Type. This field specifies the party type, with one exception (described in the following list), that is used to determine whether a user is associated with a record. This field value specifies the owner of the records in the current view mode. The allowable owner types are:
  - Person. The access control can be based on the user's Person record.
  - Position. The access control can be based on the position of the user.
  - Organization. The access control can be based on the organization of the user, as determined by the organization to which the user's current position belongs.
  - Group. The access control can be based on membership in access groups that have access to particular catalogs and categories.
  - Catalog Category. Catalog Category is not a party type. Access can be restricted to all of the data in all of the categories across catalogs to which the user has access. This data includes data in public categories and data in private categories to which the user's access groups have access. The user sees a flat (uncategorized) list of data.
- Name. The name typically suggests the view mode.
  - Personal. This name is typically used when Owner type is Person.

- Sales Rep. This name is typically used when Owner type is Position.
- Organization. This name is typically used when Owner type is Organization.
- Group. This name is typically used when Owner type is Group.
- Catalog. This name is typically used when Owner type is Catalog.

You can only use access controls for Siebel REST API GET requests. You can use an access control by adding ViewMode= to your REST API GET request URL. The URL format is same for both JSON and XML REST requests. The ViewMode query parameter used in the URL is case sensitive. If the ViewMode query parameter is not specified in the URL, by default ViewMode=Sales Rep is used. ViewMode=All is not supported for REST requests.

You can use the ViewMode query parameter on the following:

- Siebel Business Objects. The Siebel Business Objects catalog contains a list of all Business Objects exposed as Base Integration Objects. The following is an example of a Siebel Business Objects URL request with a ViewMode query parameter:

  **host-name/siebel/v1.0/data/Account/Account/?ViewMode="Sales Rep"**

- Siebel Repository Objects. The Siebel Repository Objects catalog contains lists of all repository types and catalog links to their children. The following is an example of a Siebel Repository Objects URL request with a ViewMode query parameter:

  **host-name/siebel/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet?ViewMode="Personal"**

# 3 Getting Started with the Siebel REST API

## 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 Siebel Application Interface installation. You must create and deploy a Siebel Application Interface profile before using REST related components on the Siebel server. For more information about Siebel Application Interface, see *Siebel Installation Guide for Microsoft Windows* .

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. Install Siebel Application Interface.

   For more information about installing Siebel Application Interface, see *Siebel Installation Guide for Microsoft Windows* .
4. Configure a Siebel Application Interface profile.

   For more information about configuring a Siebel Application Interface profile, see *Siebel Installation Guide for Microsoft Windows* .

   For information about Siebel RESTful configuration parameters that must be configured when you create the Siebel Application Interface Profile, see *Using Siebel Management Console to Configure a Siebel Application Interface Profile*.
5. Deploy your Siebel Application Interface profile.

   For more information about deploying a Siebel Application Interface profile, see *Siebel Installation Guide for Microsoft Windows* .
6. Determine your sizing requirements.

Based on your sizing requirements, install and configure your application interface nodes. For more information about configuring load balancing, see *Configuring the worker.properties File for Load Balancing*.

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

7. Configure business service methods for RESTful access. For more information about configuring business service methods, see *Configuring Business Service Methods for RESTful Access*.
8. Configure integration objects for REST API data access. For more information about configuring integration objects, see *Configuring Integration Objects for REST API Data Access*.

# Using Siebel Management Console to Configure a Siebel Application Interface Profile

After the Siebel Application Interface is installed, you must use the Siebel Management Console to create a Siebel Application Interface profile before using Siebel RESTful services. For more information about creating a Siebel Application Interface Profile, see *Siebel Installation Guide for Microsoft Windows* .

This topic includes the Siebel RESTful configuration parameters that must be configured when you create the Siebel Application Interface Profile. This topic includes the following:

- *Configuring REST Inbound Authentication Parameters*
- *Configuring REST Inbound Default Parameters*
- *Configuring REST Resource Parameters*

## Configuring REST Inbound Authentication Parameters

You can configure resource parameters by giving the parameters alternative query names.

The following table contains the REST inbound authentication parameters that you configure when you create a Siebel Application Interface Profile. For more information about configuring a Siebel Application Interface profile, see *Siebel Installation Guide for Microsoft Windows* .

| Siebel Management Console Parameter | Section | Description |
|---|---|---|
| Anonymous User Name | Authentication > REST Inbound Authentication | Specify the anonymous user to use for anonymous REST inbound requests. |
| Anonymous User Password | Authentication > REST Inbound Authentication | Specify the password for the anonymous user for REST inbound requests. |
| Authentication Type | Authentication > REST Inbound Authentication | Specify the authentication type that the Siebel Application Interface nodes accept for REST inbound authentication. You can select one of the following options:<br>• Basic Authentication |

| Siebel Management Console Parameter | Section | Description |
|---|---|---|
| | | • Single Sign-On<br>• OAuth |
| Trust Token | Authentication > REST Inbound Authentication | This option appears if you select the Single Sign-On or OAuth option.<br><br>Specify the trust token to use for REST inbound authentication |
| Authentication URL | Authentication > REST Inbound Authentication | This option appears if you select the OAuth option.<br><br>Specify the URL to use for REST inbound authentication. |
| User Specification | Authentication > REST Inbound Authentication | This option appears if you select the Single Sign-On option. Specify the user specification to use for REST inbound authentication. |
| Session Timeout (seconds) | Authentication > REST Inbound Authentication | Specify the session timeout, in seconds, to use for REST inbound authentication. This is the timeout which a connection will be kept open for further requests from same user. |
| Secure Channel | Authentication > REST Inbound Authentication. | This option applies only for the OAuth authentication type as follows:<br><br>• Select this check box only when you have already imported the Authentication URLfs CA certificate into the Application Interface truststore.<br>• Deselect this check box when the Authentication URLfs CA certificate is not available in the Application Interface truststore.<br><br>In this case, the Application Interface trusts all certificates while calling the Authentication URL over HTTPS. |

# Configuring REST Inbound Default Parameters

The following table contains the REST inbound default parameters that you configure when you create a Siebel Application Interface Profile. For more information about configuring a Siebel Application Interface profile, see *Siebel Installation Guide for Microsoft Windows* .

| Siebel Management Console Parameter | Section | Description |
|---|---|---|
| Object Manager | REST Inbound Defaults | Select the Object Manager component to use for REST inbound communications, such as EAI Object Manager. |
| REST Response Base URL | REST Inbound Defaults | Specify the base URL used to generate URLs for resources in REST responses. |
| Maximum Possible Connections | REST Inbound Defaults | Specify the maximum number of possible connections available to serve REST requests. Default value: 20. |
| Minimum Connections in Pool Size (%) | REST Inbound Defaults | Specify the minimum number of connections in the pool available to serve REST requests, as a percentage of the maximum. Default value: 25. |

# Configuring REST Resource Parameters

The table below contains the REST resource parameters that you configure when you create a Siebel Application Interface Profile. For more information about configuring a Siebel Application Interface profile, see *Siebel Installation Guide for Microsoft Windows* .

| Siebel Management Console Parameter | Section | Description |
|---|---|---|
| Method Name | REST Inbound Defaults / REST Resource Parameter List > Query. | Specify the method name to use for queries. |
| Name | REST Inbound Defaults / REST Resource Parameter List > Query > Parameter List. | Specify the name for each query parameter. |
| Alias | REST Inbound Defaults / REST Resource Parameter List > Query > Parameter List | Specify the alias for each query parameter in the REST resource URIs. Query parameters include: Limit, Pagesize, and Offset. |

# Configuring the worker.properties File for Load Balancing

A load balancer acts as a reverse proxy and distributes network or application traffic across a list of available of servers. A load balancer works as a middleman between the client and the servers by accepting requests from clients and distributing the requests to the backend servers based on the configured parameters.

The Apache **HTTPD mod_jk** Web server module is used to load balance Apache Tomcat servers. The **load balancer.mod_jk** is the connector used to connect Apache Tomcat with Web servers using an AJP connector.

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.

Oracle recommends for optimal usage of Siebel User Session Connections maintained on Apache Tomcat.

## To configure load balancing

1. Download and install the version of Apache HTTPd Server you want to install from the following location:
   **http://tomcat.apache.org/**
   For detailed information about installing the HTTPd Server, see Apache Tomcat documentation.
2. To configure the listen port, go to **<HTTPDRootDir\>** folder and do the following:
   a. Using any text editor, open the **httpd.conf** file on the Web server.
   b. Locate the Listen section and add the HTTP port number:
      Listen <port>
   c. Save the **httpd.conf** file.
   d. Start the Web server by executing the **httpd.exe** file.
   e. To check if the Web server is running, open a Web browser and enter the following URL:
      **http://hostip:<port>**
3. Download the Apache **mod_jk** module from the following location:
   **http://httpd.apache.org/download.cgi**
4. Save the **mod_jk.so** file then place it in the **<HTTPDRootDir\>** modules directory folder.
5. Create the following properties and log files:
   a. Create the **workers.properties** file then place it in the **<HTTPDRootDir\>** directory folder.
   b. Create the **mod_jk.log** file then place it in the **<HTTPDRootDir\>** directory folder.
6. Modify the apache **httpd.conf** file as follows:
   a. Add a Load statement to load the **mod_jk.so** file as follows:

```
# Load the mod_jk module

LoadModule jk_module modules/mod_jk.so
```

   b. Add an if statement to define the commands to be executed once the module is loaded as follows:

```
#Declare the module for use with the <IfModule directive> element.
<IfModule jk_module>
 JkWorkersFile conf/workers.properties
 JkLogFile logs/mod_jk.log
 JkLogStampFormat "[%b %d %Y - %H:%M:%S] "
 JkRequestLogFormat "%w %V %T"
 JkLogLevel trace
 JkMount /* loadbalancer
 JkMount /Jkmanager status
<IfModule>
```

c. Save and close the **httpd.conf** file.

7. Define the list of Tomcat workers that can accept requests by adding the following configuration to the **workers.properties** file:

```
# Define workers

worker.list=loadbalancer,status

# Set properties for worker1
worker.javacontainer1.type=ajp13
worker.javacontainer1.host=<hostip>
worker.javacontainer1.port=<ajportnumber>
worker.javacontainer1.lbfactor=1
worker.javacontainer 1.socket_keepalive=1
worker.javacontainer1.socket_timeout=300

# Set properties for worker2
worker.javacontainer2.type=ajp13
worker.javacontainer2.host=<hostip>
worker.javacontainer2.port=<ajportnumber>
worker.javacontainer2.lbfactor=1
worker.javacontainer2.socket_keepalive=1
worker.javacontainer2.socket_timeout=300

# Set properties for loadbalancer
worker.loadbalancer.type=lb
worker.loadbalancer.balance_workers= javacontainer1, javacontainer2

# Get statistics
worker.status.type=status
```

**Note:** For the **worker.<workername>** property, both Apache Tomcat workers must have a unique name. For each node, the workername must be same as the JVMRoutename defined in the Apache Tomcat **server.xml** file. For example, if tomcat 1 has javacontainer1 set as the value for the JVMRoute name in the Apache Tomcat **server.xml** file for tomcat1, then the worker.properties parameter must have a worker.javacontainer1 value.

8. Test the server by submitting the following REST API request:

**http://hostip:<port>/siebel/v1.0/data/Account/Account/88-431RF**

# 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* .

**CAUTION:** 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. Restart the Siebel Application Interface Apache Tomcat server by executing the following batch scripts:

    For Microsoft Windows:

    ```
    \applicationcontainer\bin\shutdown.bat
    \applicationcontainer\bin\startup.bat
    ```

    For UNIX:

    ```
    \applicationcontainer\bin\shutdown.sh
    \applicationcontainer\bin\startup.sh
    ```

# 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:

If BO = "Service Request" then the IO name must be "Base Service Request"

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

# Creating an Outbound Web Service Based on an Open APICompliant JSON File

Use the procedures in this topic to create an outbound Web Service based on an OpenAPI compliant JSON file.

## To create an outbound Web service based on an OpenAPI compliant JSON file

1. In Siebel Tools, create a new workspace.
2. From the File menu, choose New Object to display the New Object Wizards dialog box.
3. Click the EAI tab, and then double-click Web Service.

   The JSON Import Wizard appears.

   a. Select the project where you want the objects to be held after they are created from the JSON document.
   b. Specify the external JSON schema document that contains the Web service or Web services definition that you want to import.

c. Specify the log file where you want errors, warnings, and other information related to the import process to be logged or accept the default.

4. Click Next.

   A summary of your import information, as well as any errors, appears.
5. Oracle recommends that you do not select the Deploy the Integration Object(s) and the Proxy Business Service(s).
6. Click Finish to complete the process of importing the external object definitions into the Siebel repository.

This procedure generates two objects in the Siebel repository:

- A REST outbound proxy business service. This service acts as a client-side implementation of the Web service and includes the operations and the arguments to the operations defined in the JSON document.
- Integration objects, representing input and output parameters of the service methods, if any of the operations require a complex argument to be passed. If the service does not use complex arguments, then no integration object definitions will be created.

Business Services, methods, arguments, and port information are shown in the Administration - Web Services screen, REST Outbound Web Services view in the Siebel client.

Before importing the external schema, ensure that there are no objects in the Siebel Database with the same name as that of the new Web Service, Port, Port Type and Operation or Methods.

For additional information about outbound web services, see *Integration Platform Technologies: Siebel Enterprise Application Integration* .

# 4 Using the Siebel REST API

## 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 both JSON and XML examples that demonstrate 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*.
  - 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*.
  - 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*.
  - 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*.
  - 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*.
  - Response Body. The code example for the Siebel REST API response.

**Note:** Because of the length of REST responses, some REST responses have been omitted.

# Using Siebel REST API to Access Siebel Repository Resources JSON Examples

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.

This topic includes the following information:



## Querying for a Siebel CRM Repository Resource

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

The following details are for a request query that returns control properties of the WriteRecord Method in the SIS Account List Applet applet on the Siebel CRM Server:

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

## Querying for a Siebel CRM Repository Resource with a Search Specification

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

The following details are for a request query that returns properties of the SIS Account List Applet applet based on search specification from the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet?searchspec=[Business Component] LIKE 'B***
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Repository Resource with a Returned Field List

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

The following details are for a request query that returns field values only for the Name, ProjectName, and Comments fields of the SIS Account List Applet applet from the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet?fields=Name,ProjectName,Comments**
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Repository Resource That Returns Child Links Only for Lists

You can query for a Siebel CRM Repository Resource that returns child links for lists by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Repository Resource that returns child links only for lists on the Siebel CRM Server:

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

# Querying for a Siebel CRM Repository Resource That Returns Child Links Only for Lists and Charts

You can query for a Siebel CRM Repository Resource that returns child links for lists by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Repository Resource that returns child links only for lists and charts on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/dev_sadmin_arul/Applet/SIS Account List Applet?ChildLinks=List,Chart**
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic

# Querying for a Siebel CRM Repository Resource That Does Not Return Any Child Links

You can query for a Siebel CRM Repository Resource that does not return child links by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Repository Resource that does not return child links on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/dev_sadmin_arul/Applet/SIS Account List Applet?ChildLinks=None**
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic

# Querying for a Siebel CRM Repository Resource with Access Controls

You can query for a Siebel CRM Repository Resource with access controls by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Repository Resource with a ViewMode="Personal" access control on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet?ViewMode="Personal"**
- HTTP Method: GET
- Content-Type: application/json

- Authorization: Basic
- Request body: None

# Inserting a Siebel CRM Repository Resource

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

The following details are for a request to insert a new applet on the Siebel CRM Server:

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

```
{
 "Name":"SIS Account List Applet_1",
 "ProjectName":"Siebel Rest",
 "UpgradeBehavior":"Preserve",
 "Comments":"SIS Account List Applet: Added by Rest"
}
```

# Upserting a Siebel CRM Repository Resource

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

The following details are for a request to upsert a repository resource by updating the comments of the applet and adding a BtnAutoSchedule control to it on the Siebel CRM Server:

- URL: **http://ServerName:port/siebel/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",
}
```

# 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/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet_1**
- HTTP Method: DELETE

- Content-Type: application/json
- Authorization: Basic
- Request body: None

# Using the Describe Parameter to Return Top-Level Repository Objects

You can use the OpenAPI Describe parameter by appending Describe to an HTTP GET request to the resource's URI.

The following details are for a Describe request to return top-level repository objects from the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/main/describe**
- 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:


```
{
 "swagger": "2.0",
 "info":
{
},
 "schemes":
 [
 "http",
 "https"
 ],
 "securityDefinitions":
 {
 "Basic Auth":
 {
 "type": "basic"
 }
 },
 "externalDocs":
 {
 "description": "OpenAPI",
 "url": "https://openapis.org"
 },
 "host": "host:port number",
 "basePath": "/siebel/v1.0",
 "tags": [
 {
 "name": "workspace/main/Applet/describe",
 "description": "Catalogging of Applet",
 "externalDocs":
 {
 "description": "Find Out More",
 "url": ""
 }
```

```
},
{
"name": "workspace/main/Application/describe",
"description": "Catalogging of Application",
"externalDocs":
{
"description": "Find Out More",
"url": ""
}
}
],
"paths":
{
"/workspace/main/Applet/describe":
{
"get":
{
"tags": [
"workspace/main/Applet/describe"
],
"summary": "",
"description": "",
"operationId": "workspace/main/Applet/describe",
"produces": [
"application/xml",
"application/json"
],
"responses": {
"200": {"description": "Successful Operation"},
"204": {"description": "No Resource Found"},
"404": {"description": "There is no data for the requested resource"},
"500": {"description": "Internal Server Error"}
},
"parameters": [],
"security": [
{
"Basic Auth": [],
"OAuth 2.0": []
}
]
}
}
}
```


# Using Describe to Return Metadata for a Repository Object and Children Catalog URLs

You can use the OpenAPI Describe parameter by appending Describe to an HTTP GET request to the resource's URI.

The following details are for a Describe request to return metadata for a repository object and children catalog URLs from the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/main/Applet/describe**
- HTTP Method: GET
- 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: application/json
- Response body:


```
{
 "swagger": "2.0",
 "info": {},
 "schemes": [],
 "securityDefinitions": {},
 "externalDocs": {},
 "host": "host:port number",
 "basePath": "/siebel/v1.0",
 "definitions": {},
 "tags": [],
 "paths": {
 "/workspace/main/Applet/{key}/Applet Browser Script/describe": {
 "get": {
 "tags": [
 "workspace/main/Applet/{key}/Applet Browser Script/describe"
 ],
 "summary": "",
 "description": "",
 "operationId": "workspace/main/Applet/{key}/Applet Browser Script/describe",
 "produces": [
 "application/xml",
 "application/json"
 ],
 "responses": {
 "200": {"description": "Successful Operation"},
 "204": {"description": "No Resource Found"},
 "404": {"description": "There is no data for the requested resource"},
 "500": {"description": "Internal Server Error"}
 },
 "parameters": [
 {
 "name": "key",
 "in": "path",
 "description": "",
 "required": true,
 "default": "key",
 "type": "string"
 }
 ],
 "security": [
 {
 "Basic Auth": [],
 "OAuth 2.0": []
 }
 ]
 }
 }
 }
}
```

# Using Describe to Return Metadata for Child Repository Objects and Catalog URLs for Grand Child Repository Objects

You can use the OpenAPI Describe parameter by appending Describe to an HTTP GET request to the resource's URI.

The following details are for a Describe request to return metadata for child repository objects and grand child catalog URLs from the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/main/Applet/{key}/Control/describe**
- 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:


```
{
 "swagger": "2.0",
 "info": {},
 "schemes": [],
 "securityDefinitions": {},
 "externalDocs": {},
 "host": "host:port number",
 "basePath": "/siebel/v1.0",
 "definitions": {
 "workspace_main_Applet_{key}_Control_": {
 "type": "object",
 "required": [
 "Name",
 "Type",
 "Show Popup",
 "HTML Row Sensitive",
 "Parent Id",
 "HTML Default Control"
 ],
 "properties": {
 "ActiveX Bind Property": {
 "maxLength": 75,
 "x-siebel-precision": "0",
 "type": "string",
 "x-siebel-datatype": "DTYPE_TEXT",
 "x-siebel-scale": "0",
 "title": "ActiveX Bind Property"
 },
 "tags": [],
 "paths":{}
}
```

# Using Siebel REST API to Access Siebel Business Objects JSON Examples

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.

This topic includes the following information:



## Querying for a Siebel CRM Business Object

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

The following details are for a request to query for an Account business object with an ID of 1LS-9XKU on the Siebel CRM Server:

- URL: **http://ServerName:port/siebel/v1.0/data/Account/Account/1LS-9XKU**

- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Business Object with a Search Specification

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

The following details are for a request to query for a Contact business object with a search specification parameter on the Siebel CRM Server:

- URL: **http://ServerName:port/siebel/v1.0/data/Account/Account/Account/1-32HG/Contact/?searchspec=([First Name] LIKE 'J*' AND [Last Name] LIKE 'A*')**
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic

# Querying for a Siebel CRM Business Object with a Returned Fields List

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

The following details are for a request to query for an Account business object that return values only for the Name, Location, and Account Status fields of an Account with an 1-32HG ID number on the Siebel CRM Server:

- URL: **http://ServerName:port/siebel/v1.0/data/Account/Account/Account/1-32HG?fields=Name, Location, Account Status**
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic

# Inserting a Siebel CRM Business Object

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

The following details are for a request to create a new account on the Siebel CRM Server:

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

- Request body:

```
{
 "Name": "AccountExample",
 "Primary Organization": "Millennium Institutional Finance Services IF ENU",
 "Location": "HQ-Distribution",
 "Description": "AccountData",
 "Primary Organization Id": "1-1DG",
}
```

# 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 Contact business object into an existing Account business object on the Siebel CRM Server:

- URL: **http://ServerName:port/siebel/v1.0/data/Account/Account/88-431RF/Contact**
- HTTP Method: PUT
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
 "Employee Number":"1231",
 "Employer Name":"BXM",
 "Bill To First Name":"MAYANew",
 "Bill To Last Name": "ABRAHAMNew",
 "Primary Organization Id":"0-R9NH",
 "Account Integration Id":"",
 "Job Title":"",
 "Person UId":"0CR-1MF5Z611",
 "Primary Organization":"Default Organization",
 "Personal Contact":"N"
}
```

# Inserting Multiple Siebel CRM Child Business Objects

You can insert multiple 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 multiple Contacts into an existing Account business object on the Siebel CRM Server:

- URL: **http://ServerName:port/siebel/v1.0/data/Account/Account/Account/1-1/Contact**
- HTTP Method: PUT
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
 "Contact":[
 {
 "Primary Organization Id":"1-1DG",
```

```
 "Primary Organization":"Default Organization",
 "First Name":"Ken",
 "Last Name": "Bass",
 "Id":"12345"
 },
 {
 "Primary Organization Id":"1-1DG",
 "Primary Organization":"Default Organization",
 "First Name":"test",
 "Last Name": "test1",
 "Id":"123456"
 }
 ]
}
```

# Upserting a Siebel CRM Business Object

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

The following details are for a request to update the fields of an existing Account business object on the Siebel CRM Server:

- URL: **http://ServerName:port/siebel/v1.0/data/Account/Account/88-431RF**
- HTTP Method: PUT
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
 "Name": "AccountExample",
 "Primary Organization":
 "Millennium Institutional Finance Services IF ENU",
 "Location": "HQ-Distribution",
 "Description": "AccountDataUpdate",
 "Primary Organization Id": "1-1DG"
}
```

# 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 update the fields of an Opportunity of a given Account child business object on the Siebel CRM Server:

- URL: **http://ServerName:port/siebel/v1.0/data/Account/Account/88-431RF/Opportunity**
- HTTP Method: PUT
- Content-Type: application/json
- Authorization:  Basic
- Request body:

```
{
 "Id":"123456",
 "Name":"NewOpp",
 "Currency Code": "AUD",
```

```
 "Primary Organization":"Default Organization"
}
```

# 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 an Account business object on the Siebel CRM Server:

- URL: **http://ServerName:port/siebel-rest/v1.0/data/Account/Account/88-43CGR**
- HTTP Method: DELETE
- Content-Type: application/json
- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Business Object That Returns Specified Child Links

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

The following details are for a request to query for a Siebel CRM Business Object that returns child links for the UT Account Partner on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account?ChildLinks=UT Account Partner**
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Business Object That Returns Child Links to CUT

You can query for a Siebel CRM Business Object that returns child links to CUT by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Business Object that returns child links only for the CUT Address for Account/Contact, FINS Security - Account External Holdings and FINS cBanking Facility on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account?ChildLinks=CUT Address for Account/Contact,FINS Security - Account External Holdings,FINS cBanking Facility**
- HTTP Method: GET
- Content-Type: application/json

- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Business Object That Returns Child Links Only for Lists

You can query for a Siebel CRM Business Object that returns child links for lists by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Business Object that returns child links only for List Mgmt Lists on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/88-3CFLJ/Contact?ChildLinks=List Mgmt Lists**
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Business Object with Access Controls

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

The following details are for a request to query for a Siebel CRM Business Object with a ViewMode="Sales Rep" access control on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/?ViewMode="Sales Rep"**
- HTTP Method: GET
- Content-Type: application/json
- Authorization: Basic

# Querying for a Siebel CRM Business Object That Does Not Return Child Links

You can query for a Siebel CRM Business Object that does not return child links by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Business Object that does not return child links on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/88-3CFLJ/Contact?ChildLinks=None**
- HTTP Method: GET
- Content-Type: application/json

- Authorization: Basic
- Request body: None

# Using Describe to Return the Siebel Base Business Object Catalog

You can use the OpenAPI Describe parameter by appending Describe to an HTTP GET request to the resource's URI.

The following details are for a Describe request to return the Siebel Base Business Object from the the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/describe**
- 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:


```
{
 "swagger":"2.0",
 "info":{
 "description":"Siebel REST API",
 "version":"1.0",
 "title":"Siebel REST API",
 "contact":{
 "email":"sample@sample.com"
 }
 },
 "schemes":[
 "http",
 "https"
 ],
 "securityDefinitions":{
 "Basic Auth":{
 "type":"basic"
 },
 "OAuth 2.0":{
 "type":"oauth2",
 "authorizationUrl":"http://openAPI.io/",
 "flow":"implicit",
 "scopes":{
 "write:":"modify",
 "read:":"read only"
 }
 }
 },
 "externalDocs":{
 "description":"OpenAPI",
 "url":"https://openapis.org"
 },
 "host": "host:port number",
 "basePath":"/siebel/v1.0",
 "tags":[
 {
 "name":"data/Account/describe",
```

```
"description":"Catalogging of Account",
"externalDocs":{
"description":"Find Out More",
"url":""
}
},
{
"name":"data/Service Request/describe",
"description":"Catalogging of Service Request",
"externalDocs":{
"description":"Find Out More",
"url":""
}
},
"paths":{
"/data/Account/describe":{
"get":{
"tags":[
"data/Account/describe"
],
"summary":"",
"description":"",
"operationId":"data/Account/describe",
"produces":[
"application/xml",
"application/json"
],
"responses":{
"200":{"description":"Successful Operation"},
"204":{ "description":"No Resource Found"},
"404":{"description":"There is no data for the requested resource"},
"500":{"description":"Internal Server Error"}
},
"parameters":[

],
"security":[
{
"Basic Auth":[],
"OAuth 2.0":[]
}
]
}
},
"/data/Service Request/describe":{
"get":{
"tags":[
"data/Service Request/describe"
],
"summary":"",
"description":"",
"operationId":"data/Service Request/describe",
"produces":[
"application/xml",
"application/json"
],
"responses":{
"200":{"description":"Successful Operation"},
"204":{"description":"No Resource Found"},
"404":{"description":"There is no data for the requested resource"},
"500":{"description":"Internal Server Error"}
},
"parameters":[],
"security":[
{
"Basic Auth":[],
```

```
"OAuth 2.0":[]
}
]
},
"post":{
"tags":[
"data_Account_Account_"
],
"summary":"",
"description":"",
"operationId":"data_Account_Account_/post",
"produces":[
"application/xml",
"application/json"
],
"consumes":[
"application/xml",
"application/json"
],
"responses":{
"200":{"description":"Successful Operation"},
"204":{"description":"No Resource Found"},
"404":{ "description":"There is no data for the requested resource"},
"500":{ "description":"Internal Server Error"}
},
"parameters":[
{
"in":"body",
"name":"body",
"description":"",
"required":true,
"schema":{
"$ref":"#/definitions/data_Account_Account_"
}
}
],
"security":[
{
"Basic Auth":[],
"OAuth 2.0":[]
}
]
}
}
}

}
```


# Using the Describe Parameter to Return the Business Object Catalog of a Parent Business Component

You can use the OpenAPI Describe parameter by appending Describe to an HTTP GET request to the resource's URI.

The following details are for a Describe request to return the Business Object Catalog of a parent Business Component from the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/describe**
- HTTP Method: GET

- 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:


```
{
"definitions":{
"data_Account_Account_":{
"type":"object",
"required":
"properties":
}
},
"tags":[
{
"name":"data/Account/Account/",
"description":"Operations available on data/Account/Account/",
"externalDocs":{
"description":"Find Out More",
"url":""
}
},
"paths":{
"/data/Account/Account/":{
"get":{
"tags":[
"data_Account_Account_"
],
"summary":"",
"description":"",
"operationId":"data_Account_Account_/get",
"produces":[
"application/xml",
"application/json"
],
"responses":{
"200":{"description":"Successful Operation"},
"204":{"description":"No Resource Found"},
"404":{"description":"There is no data for the requested resource"},
"500":{"description":"Internal Server Error"}
},
"parameters":[
{
"name":"key",
"in":"path",
"description":"",
"required":true,
"type":"string"
}
],
"security":[
{
"Basic Auth":[],
"OAuth 2.0":[]
}
]
}
}
}
```

```
}
```

# Using the Describe Parameter to Return Metadata for a Parent Business Component and Catalog of Child Business Components

You can use the OpenAPI Describe parameter by appending Describe to an HTTP GET request to the resource's URI.

The following details are for a Describe request to return Metadata for a parent Business Component and catalog of Child Business Components from the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/1-6/Contact/describe**
- 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:

```
{
"definitions":{
 "data_Account_Account_{key}_Contact_":{
 "type":"object",
 "required":[],
 "properties":[],
 "tags":[
 {
 "name":"data/Account/Account/{key}/Contact/",
 "description":"Operations available on data/Account/Account/{key}/Contact/",
 "externalDocs":{
 "description":"Find Out More",
 "url":""
 }
 },
 "paths":{
 "/data/Account/Account/{key}/Contact/":{
 "get":{
 "tags":[
 "data_Account_Account_{key}_Contact_"
 ],
 "summary":"",
 "description":"",
 "operationId":"data_Account_Account_{key}_Contact_/get",
 "produces":[
 "application/xml",
 "application/json"
 ],
 "responses":{
 "200":{"description":"Successful Operation"},
 "204":{"description":"No Resource Found"},
 "404":{ "description":"There is no data for the requested resource"},
 "500":{"description":"Internal Server Error"}
 },
 "parameters":[
```

```
{
"name":"key",
"in":"path",
"description":"",
"required":true,
"type":"string"
}
],
"security":[
{
"Basic Auth":[],
"OAuth 2.0":[]
}
]
},
"post":{
"tags":[
"data_Account_Account_{key}_Contact_"
],
"summary":"",
"description":"",
"operationId":"data_Account_Account_{key}_Contact_/post",
"produces":[
"application/xml",
"application/json"
],
"consumes":[
"application/xml",
"application/json"
],
"responses":{
"200":{"description":"Successful Operation"},
"204":{"description":"No Resource Found"},
"404":{"description":"There is no data for the requested resource"},
"500":{"description":"Internal Server Error"}
},
"parameters":[
{
"in":"body",
"name":"body",
"description":"",
"required":true,
"schema":{
"$ref":"#/definitions/data_Account_Account_{key}_Contact_"
}
}
],
"security":[
{
"Basic Auth":[],
"OAuth 2.0":[]
}
]
}
}
}

}
```

# Querying for all Contacts in an Account Without Using the Uniformresponse Flag Where Response Returns a Single Record

You can query for a Siebel CRM object by sending an HTTPS GET request to the resource's URI.

The following details are to request for all Contacts in a given Account from the Siebel CRM Server, without using the **uniformresponse** flag, and receive a single record in the response:

- URI: **GET https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/Contact?fields=First Name, Last Name&childlinks=Account,Position,Organization**
- 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:

```
{
 "Id": "0CR-1MF5Z6",
 "First Name": "JOHN",
 "Last Name": "SMITH",
 "Link": [
 {
 "rel": "self",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6",
 "name": "Contact"
 },
 {
 "rel": "canonical",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6",
 "name": "Contact"
 },
 {
 "rel": "parent",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND",
 "name": "Account"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Position",
 "name": "Position"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Organization",
 "name": "Organization"
 },
 {
```

```
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Account",
 "name": "Account"
 }
 ]
}
```

# Querying for all Contacts in an Account by Using the Uniformresponse Flag

You can query for a Siebel CRM object by sending an HTTPS GET request to the resource's URI.

The following details are to request for all Contacts in a given Account from the Siebel CRM Server by using the **uniformresponse** flag:

- URI: **GET https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/Contact?fields=First Name, Last Name&childlinks=Account,Position,Organization&uniformresponse=yes**
- 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:

```
{
 "items": [
 {
 "Id": "0CR-1MF5Z6",
 "First Name": "JOHN",
 "Last Name": "SMITH",
 "Link": [
 {
 "rel": "self",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6",
 "name": "Contact"
 },
 {
 "rel": "canonical",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6",
 "name": "Contact"
 },
 {
 "rel": "parent",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND",
 "name": "Account"
 },
 {
 "rel": "child",
```

```
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Position",
 "name": "Position"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Organization",
 "name": "Organization"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Account",
 "name": "Account"
 }
 ]
 }
 ],
 "Link": [
 {
 "rel": "self",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/Contact?
&uniformresponse=y&childlinks=Account,Position,Organization&fields=First Name, Last Name",
 "name": "Contact"
 }
 ]
}
```

# Querying for all Contacts in an Account Without Using the Uniformresponse Flag Where Response Returns Multiple Records

You can query for a Siebel CRM object by sending an HTTPS GET request to the resource's URI.

The following details are to request for all Contacts in a given Account from the Siebel CRM Server, without using the **uniformresponse** flag, and receive multiple records in the response:

- URI: **GET https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/Contact?fields=First Name, Last Name&childlinks=Account,Position,Organization**
- 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:

```
{
 "items": [
 {
 "Id": "0CR-1MF5Z6",
 "First Name": "JOHN",
```

```
 "Last Name": "SMITH",
 "Link": [
 {
 "rel": "self",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6",
 "name": "Contact"
 },
 {
 "rel": "canonical",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6",
 "name": "Contact"
 },
 {
 "rel": "parent",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND",
 "name": "Account"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Position",
 "name": "Position"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Organization",
 "name": "Organization"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Account",
 "name": "Account"
 }
 ]
 }
 {
 "Id": "0V-18PLXQ",
 "First Name": "WILLIAM",
 "Last Name": "BROWN",
 "Link": [
 {
 "rel": "self",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6",
 "name": "Contact"
 },
 {
 "rel": "canonical",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6",
 "name": "Contact"
 },
 {
 "rel": "parent",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND",
 "name": "Account"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Position",
 "name": "Position"
```

```
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Organization",
 "name": "Organization"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/
Contact/0CR-1MF5Z6/Account",
 "name": "Account"
 }
 ]
 }
 ],
 "Link": {
 "rel": "self",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/Contact?
&childlinks=Account,Position,Organization&fields=First Name, Last Name",
 "name": "Contact"
 }
 }
```

# Querying for a Single Account Record Without Using the Uniformresponse Flag

You can query for a single Siebel CRM object by sending an HTTPS GET request to the resource's URI.

The following details are to request for a single account from the Siebel CRM Server without using the **uniformresponse** flag:

- URI: **GET https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND?fields=Name,Location,Id&childlinks=Contact,Position**
- 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:

```
{
 "Name": "3Com",
 "Id": "88-26CND",
 "Location": "Headquarters",
 "Link": [
 {
 "rel": "self",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND",
 "name": "Account"
 },
 {
```

```
"rel": "canonical",
"href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND",
"name": "Account"
},
{
"rel": "child",
"href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/Contact",
"name": "Contact"
},
{
"rel": "child",
"href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/Position",
"name": "Position"
}
]
}
```

# Querying for a Single Account Record by Using the Uniformresponse Flag

You can query for a single Siebel CRM object by sending an HTTPS GET request to the resource's URI.

The following details are to request for a single account from the Siebel CRM Server by using the **uniformresponse** flag:

- URI: **GET https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND?fields=Name,Location,Id&childlinks=Contact,Position,Organization&uniformresponse=y**
- 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:

```
{
 "items": [
 {
 "Name": "3Com",
 "Id": "88-26CND",
 "Location": "Headquarters",
 "Link": [
 {
 "rel": "self",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND",
 "name": "Account"
 },
 {
 "rel": "canonical",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND",
 "name": "Account"
 },
 {
```

```
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/Contact",
 "name": "Contact"
 },
 {
 "rel": "child",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account/88-26CND/Position",
 "name": "Position"
 },
 ]
 }
]
 "Link": [
 {
 "rel": "self",
 "href": "https://<host_name>:<port_number>/siebel/v1.0/data/Account/Account?
uniformresponse=y&childlinks=Contact,Position,Organization&fields=Name,Location,Id",
 "name": "Account"
 }
 ]
}
```

# Using Siebel REST API to Access Siebel Business Services JSON Examples

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, and method parameters in the URI.

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*.

This topic includes the following information:

- *Accessing a Siebel Business Service with Arguments in the Request Body*
- *Acessing a Siebel Business Service with Arguments in the Request URI*
- *Accessing the Query By Example Method of the Siebel Account Business Service*
- *Querying for an Account Using the Siebel Business Service QueryId Method*
- *Inserting an Account Using the Siebel Account Business Service*
- *Updating an Account Using a Siebel Account Business Service*
- *Upserting an Account Using a Siebel Account Business Service*
- *Using Describe to Return Methods for a Business Service*
- *Using Describe to Return a Catalog of URLs for All Available Business Services*

## Accessing a Siebel Business Service with Arguments in the Request Body

You can access a Siebel CRM business service by sending an HTTP POST request to the resource's URI.

The following details are for a request to call the CreateAccount Method on the Account Business Service with business service arguments included in the request body. This request creates a REST Test Business Service3 account on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/service/Account/CreateAccount**
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
 "body":
 {
 "Account IO":
 {
 "IntObjectName":"Account IO",
 "IntObjectFormat":"Siebel Hierarchical",
 "ListOfAccount IO":
 {
 "Account":
 {
 "Name":"REST Test Business Service3"
 }
 }
 }
 }
}
```

# Acessing a Siebel Business Service with Arguments in the Request URI

You can access a Siebel CRM business service by sending an HTTP POST request to the resource's URI.

The following details are for a request to call the QueryPage method of the AccountWS Business Service with PageSize=10, StartRowNum=0 and ViewMode=All parameters included in the request URI. This request returns the REST Test Business Service3 account in the format of the integration Account_EMR object on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/service/AccountWS/QueryPage?PageSize=10&StartRowNum=0&ViewMode=All**
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
 "body":
 {
 "SiebelMessage":
 {
 "MessageId":"",
 "MessageType":"Integration Object",
 "IntObjectName":"Account_EMR",
 "IntObjectFormat":"Siebel Hierarchical",
 "ListOfAccount_EMR":
 {
```

```
 "Account":
 {
 "Name":"REST Test Business Service3"
 }
 }
 }
 }
}
```

# Accessing the Query By Example Method of the Siebel Account Business Service

You can access a Siebel CRM business service by sending an HTTP POST request to the resource's URI.

The following details are for a request to call the QueryByExample method of the Siebel Account business service. This request will query and return an account with the name 3Com.

- URI: **http://ServerName:port/siebel/v1.0/service/Siebel Account/QueryByExample**
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

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

# Querying for an Account Using the Siebel Business Service QueryId Method

You can query for an Account using the Siebel CRM business service QueryId method by sending an HTTP POST request to the resource's URI.

The following details are for a request to call the QueryId method of the Siebel Account business service and the PrimaryRowId parameter is accepted as the query parameter. The following example query returns an account with an 88-459YQ PrimaryRowId value.

- URI: **http://ServerName:port/siebel/v1.0/service/Siebel Account/QueryById?PrimaryRowId=88-459YQ**
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
}
```

# Inserting an Account Using the Siebel Account Business Service

You can access a Siebel CRM business service by sending an HTTP POST request to the resource's URI.

The following details are for a request to call the Insert method of Siebel Account Business service to insert a new account with details given in the request.

- URI: **https://ServerName:port/siebel/v1.0/service/Siebel Account/Insert**
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
 "body":
 {
 "SiebelMessage":
 {
 "MessageId":"",
 "MessageType":"Integration Object",
 "IntObjectName":"Account Interface",
 "IntObjectFormat":"Siebel Hierarchical",
 "ListOfAccount Interface":
 {
 "Account":
 {
 "Account Id":"2345",
 "Name":"REST Test Business Service"
 }
 }
 }
}
}
```

# Updating an Account Using a Siebel Account Business Service

You can access a Siebel CRM business service by sending an HTTP POST request to the resource's URI.

The following details are for a request to call the Update method of Siebel Account Business Service.

- URI: **https://ServerName:port/siebel/siebel-rest/v1.0/service/Siebel Account/Update**
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
 "body":
 {
 "SiebelMessage":
 {
 "MessageId":"",
 "MessageType":"Integration Object",
 "IntObjectName":"Account Interface",
 "IntObjectFormat":"Siebel Hierarchical",
 "ListOfAccount Interface":
 {
 "Account":
 {
 "Account Id":"1-6",
 "Name":"REST Test Business Service32",
 "Main Phone Number": "2018742315"
 }
 }
 }
 }
}
```

# Upserting an Account Using a Siebel Account Business Service

You can access a Siebel CRM business service by sending an HTTP POST request to the resource's URI.

The following details are for a request to call the InsertOrUpdate method of Siebel Account Business service to insert an account with details given in the request.

- URI: **https://ServerName:port/siebel/v1.0/service/Siebel Account/InsertOrUpdate**
- HTTP Method: POST
- Content-Type: application/json
- Authorization: Basic
- Request body:

```
{
 "body":
 {
 "SiebelMessage":
 {
 "MessageId":"",
 "MessageType":"Integration Object",
 "IntObjectName":"Account Interface",
 "IntObjectFormat":"Siebel Hierarchical",
 "ListOfAccount Interface":
 {
 "Account":
 {
 "Account Id":"34567",
```

```
"Name":"Rest",
"Main Phone Number": "20187423154"
}
}
}
}
}
```

# Using Describe to Return Methods for a Business Service

You can use the OpenAPI Describe parameter by appending Describe to an HTTP GET request to the resource's URI.

The following details are for a Describe request to return methods for business services from the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/service/Siebel Account/describe**
- 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:

```
"swagger": "2.0",
"info": {},
"schemes": [],
"securityDefinitions": {},
"externalDocs": {},
"host": "host:port number",
"basePath": "/siebel/v1.0",

"definitions": {},

"tags": [],

"paths": {

"/service/Siebel Account/Delete/describe": {

"get": {

"tags": [

"service/Siebel Account/Delete/describe"

],

"summary": "",

"description": "",

"operationId": "service/Siebel Account/Delete/describe",

"produces": [
```

```
    ],
    "responses": {
    },
    "parameters": [],
    "security": [
    ]
    }
    },
    "/service/Siebel Account/Insert/describe": {
    },
    "/service/Siebel Account/InsertOrUpdate/describe": {
    },
    "/service/Siebel Account/QueryByExample/describe": {
    },
    "/service/Siebel Account/QueryById/describe": {
    },
    "/service/Siebel Account/Synchronize/describe": {
    },
    "/service/Siebel Account/Update/describe": {
    }
    }
  }
```


# Using Describe to Return a Catalog of URLs for All Available Business Services

You can use the OpenAPI Describe parameter by appending Describe to an HTTP GET request to the resource's URI.

The following details are for a Describe request to return a catalog of URLs for all available business services from the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/main/Applet/{key}/Control/describe**
- HTTP Method: GET
- 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: application/json
- Response body:


```
{
 "swagger": "2.0", //Existing Swagger specification version
 "info": { //Metadata about API
 },
 "schemes": [
 "http",
 "https"
 ],
 "securityDefinitions": { //information about the authorization schemes
 "Basic Auth": {
 "type": "basic"
 }
 },
 "externalDocs": {
 "description": "OpenAPI",
 "url": "https://openapis.org"
 },
 "host": "host:port number",
 "basePath": "/siebel/v1.0",
 "tags": [
 {
 "name": "service/ABO Bulk Request Explode Service/describe",
 "description": "Catalogging of ABO Bulk Request Explode Service",
 "externalDocs": {
 "description": "Find Out More",
 "url": ""
 }
 },

 {

 "name": "service/nextSet/describe",

 "description": "Catalogging of nextSet",

 "externalDocs": {

 "description": "Find Out More",

 "url": ""

 }

 }

 ],
 "paths": {
 "/service/ABO Bulk Request Explode Service/describe": { //links
 "get": {
 "tags": [
 "service/ABO Bulk Request Explode Service/describe"
 ],
 "summary": "",
 "description": "",
 "operationId": "service/ABO Bulk Request Explode Service/describe",
 "produces": [
 "application/xml",
 "application/json"
 ],
```

```
"responses": {
"200": {"description": "Successful Operation"},
"204": {"description": "No Resource Found"},
"404": {"description": "There is no data for the requested resource"},
"500": {"description": "Internal Server Error"}
},
"parameters": [],
"security": [
{
"Basic Auth": [],
"OAuth 2.0": []
}
]
}
},

"/service/describe?PageSize=10&StartRowNum=10": { //links
"get": {
"tags": [
"service/nextSet/describe"
],
"summary": "",
"description": "",
"operationId": "service/nextSet/describe",
"produces": [
"application/xml",
"application/json"
],
"responses": {
"200": {"description": "Successful Operation"},
"204": {"description": "No Resource Found"},
"404": {"description": "There is no data for the requested resource"},
"500": {"description": "Internal Server Error"}
},
"parameters": [],
"security": [
{
"Basic Auth": [],
"OAuth 2.0": []
}
]
}
}
}
}
```


# Using Siebel REST API to Access Siebel Repository Data XML Examples

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. Requests and responses include XML data.

This topic includes the following information:

- *Querying for a Siebel CRM Repository Resource*
- *Querying for a Siebel CRM Repository Resource with a Search Specification*

# Querying for a Siebel CRM Repository Resource

You can query for a Siebel CRM 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 that returns WriteRecord control properties in the SIS Account List Applet:

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

# Querying for a Siebel CRM Repository Resource with a Search Specification

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

The following details are for a request query for a repository resource on the Siebel CRM Server that returns SIS Account List Applet properties based on a given search specification:

- URI: **http://ServerName:port/siebel/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet?searchspec=[Business Component] LIKE 'B*'**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Repository Resource that Returns Fields List

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 repository resource on the Siebel CRM Server that returns values for the Name, ProjectName, and Comments fields of the SIS Account List Applet applet:

- URI: **http://ServerName:port/siebel/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet?fields=Name,ProjectName,Comments**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic
- Request body: None

# Inserting a Siebel CRM Repository Resource

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

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

- URI: **http://ServerName:port/siebel/v1.0/workspace/dev_sadmin_temp/Applet/SIS Account List Applet_1**
- HTTP Method: POST
- Content-Type: application/xml
- Authorization: Basic
- Request body:

```
<?xml version="1.0" encoding="UTF-8" ?>
 <request>
 <Name>SIS Account List Applet_1</Name>
 <ProjectName>Siebel Rest</ProjectName>
 <UpgradeBehavior>Preserve</UpgradeBehavior>
 <Comments>SIS Account List Applet: Added by Rest</Comments>
 </request>
```

# Upserting a Siebel CRM Repository Resource

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

The following details are for a request to upsert comments and add a control to a repository resource on the Siebel CRM Server:

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

```
<?xml version="1.0" encoding="UTF-8" ?>
 <request>
 <Name>WriteRecord</Name>
 </request>
```

# Deleting a Siebel CRM Repository Resource

You can delete a Siebel CRM repository resource by sending an HTTP DELETE request to the repository 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/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet_1**
- HTTP Method: DELETE
- Content-Type: application/xml
- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Repository Resource That Returns Child Links Only for Lists

You can query for a Siebel CRM Repository Resource that returns child links for lists by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Repository that returns child links only for SIS Account Lists on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/dev_sadmin_arul/Applet/SIS Account List Applet?ChildLinks=List**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic
- Request body: None

# Querying for a Siebel CRM Repository Resource That Returns Only List and Chart Child Links

You can query for a Siebel CRM Repository Resource that returns child links for lists and charts by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Repository that returns child links only for SIS Account Lists and charts on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/dev_sadmin_arul/Applet/SIS Account List Applet?ChildLinks=List,Chart**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic

- Request body: None

## Querying for a Siebel CRM Repository Resource That Does Not Return Child Links

You can query for a Siebel CRM Repository Resource that does not return child links by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Repository Resource that does not return child links on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/dev_sadmin_arul/Applet/SIS Account List Applet?ChildLinks=None**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic

## Querying for a Siebel CRM Repository Resource with Access Controls

You can query for a Siebel CRM Repository Resource with access controls by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Repository Resource with a ViewMode="Personal" access control on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/workspace/MyWorkspace/Applet/SIS Account List Applet?ViewMode="Personal"**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic
- Request body: None

# Using Siebel REST API to Access Siebel CRM Business Objects XML Examples

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.

This topic includes the following information:

# Querying for a Siebel CRM Business Object

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

The following details are for a request query for an Account Business Object resource on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/1LS-9XKU**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic

# Querying for a Siebel CRM Business Object with a Search Specification

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

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

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/1-32HG/Contact/?searchspec=([First Name] LIKE 'J*' AND [Last Name] LIKE 'A*')**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic

# Querying for a Siebel CRM Business Object Resource that Returns Fields List

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

The following details are for a request query for a business object resource on the Siebel CRM Server that returns field values for an Account Business Object:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/1-32HG?fields=Name, Location, Account Status**

- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic

# Inserting a Siebel CRM Parent Business Object

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

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

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

```
<?xml version="1.0" encoding="UTF-8" ?>
 <request >
 <Name>AccountData</Name>
 <Primary_spcOrganization>Millennium Institutional Finance Services IF ENU</
 Primary_spcOrganization>
 <Location>HQ-Distribution</Location>
 <Description>AccountData</Description>
 <Primary_spcOrganization_spcId>1-1DG</Primary_spcOrganization_spcId>
 </request>
```

# Upserting a Siebel CRM Parent Business Object

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

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

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account**
- HTTP Method: PUT
- Content-Type: application/xml
- Authorization: Basic
- Request body:

```
<?xml version="1.0" encoding="UTF-8" ?>
 <request>
 <Name>AccountData</Name>
 <Primary_spcOrganization>Millennium Institutional Finance Services IF ENU</
 Primary_spcOrganization>
 <Location>HQ-Distribution</Location>
 <Description>AccountDataUpdate</Description>
 <Primary_spcOrganization_spcId>1-1DG</Primary_spcOrganization_spcId>
 </request>
```

# Deleting a Siebel CRM Parent Business Object

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

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

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/88-43CGR**
- HTTP Method: DELETE
- Content-Type: application/xml
- Authorization: Basic
- Request body:

```
<?xml version="1.0" encoding="UTF-8" ?>
<request>
 <Id>88-43CGR</Id>
</request>
```

# Querying for a Siebel CRM Business Object Resource that Returns UT Account Partner Child Links

You can query for a Siebel CRM business object resource that returns UT Account Partner child links by sending an HTTP PUT request to the repository resource's URI.

The following details are for a request to query for a Siebel CRM business object resource that returns UT Account Partner child links on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account?ChildLinks=UT Account Partner**
- HTTP Method: PUT
- Content-Type: application/xml
- Authorization: Basic

# Querying for a Siebel CRM Business Object That Returns Child Links to CUT

You can query for a Siebel CRM business object resource that returns child links to CUT by sending an HTTP PUT request to the repository resource's URI.

The following details are for a request to query for a Siebel CRM business object resource that returns child links to CUT on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account?ChildLinks=CUT Address for Account/Contact,FINS Security - Account External Holdings,FINS cBanking Facility**
- HTTP Method: PUT

- Content-Type: application/xml
- Authorization: Basic

# Querying for a Siebel CRM Business Object Resource That Returns List Management List Child Links

You can query for a Siebel CRM Business Object Resource that returns list management list child links by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Business Object Resource that returns list management list child links on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/88-3CFLJ/Contact?ChildLinks=List Mgmt Lists**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic

# Querying for a Siebel CRM Business Object Resource That Does Not Return Child Links

You can query for a Siebel CRM Business Object Resource that does not return child links by sending an HTTP GET request to the resource's URI.

The following details are for a request to query for a Siebel CRM Business Object Resource that does not return child links on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/88-3CFLJ/Contact?ChildLinks=None**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic

# Querying for a Siebel CRM Business Object Resource with Access Controls

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

The following details are for a request to query for a Siebel CRM Business Object Resource with a ViewMode="Sales Rep" access control on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/data/Account/Account/?ViewMode="Sales Rep"**
- HTTP Method: GET

- Content-Type: application/xml
- Authorization: Basic

## Using Siebel REST API to Access Siebel Business Services XML Examples

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, and method parameters in the URI.

This topic includes the following information:



# Using a Siebel CRM Business Service to Insert an Account

You can use a Siebel CRM Business Service resource to insert an Account by sending an HTTP POST request to the repository resource's URI.

The following details are for a request to use a business service resource to insert an Account on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/service/Siebel Account/Insert**
- HTTP Method: POST
- Content-Type: application/xml
- Authorization: Basic
- Request body:

```
<?xml version="1.0" encoding="UTF-8" ?>
<body>
 <SiebelMessage>
 <MessageId></MessageId>
 <MessageType>Integration Object</MessageType>
 <IntObjectName>EAI Account</IntObjectName>
 <IntObjectFormat>Siebel Hierarchical</IntObjectFormat>
 <ListOfEAI_spcAccount>
 <Account>
 <Name>Test Account</Name>
 </Account>
 </ListOfEAI_spcAccount>
 </SiebelMessage>
</body>
```

# Using a Siebel CRM Business Service to Update an Account

You can use a Siebel CRM Business Service resource to update an Account by sending an HTTP PUT request to the repository resource's URI.

The following details are for a request to update a Main Phone field for an Account Business Service on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/service/Siebel Account/Update**
- HTTP Method: PUT
- Content-Type: application/xml
- Authorization: Basic
- Request body:

```
<?xml version="1.0" encoding="UTF-8" ?>
<body>
 <SiebelMessage>
 <MessageId></MessageId>
 <MessageType>Integration Object</MessageType>
 <IntObjectName>Account Interface</IntObjectName>
 <IntObjectFormat>Siebel Hierarchical</IntObjectFormat>
 <ListOfAccount_spcInterface>
 <Account><Account_spcId>1-34Z</Account_spcId>
 <Main_spcPhone_spcNumber>2018742315</Main_spcPhone_spcNumber>
 </Account>
 </ListOfAccount_spcInterface>
 </SiebelMessage>
</body>
```

# Using a Siebel CRM Business Service to Delete an Account

You can use a Siebel CRM Business Service to delete an Account by sending an HTTP DELETE request to the resource's URI.

The following details are for a request to delete an Account on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/service/Siebel Account/Delete**
- HTTP Method: DELETE
- Content-Type: application/xml
- Authorization: Basic
- Request body:

```
<?xml version="1.0" encoding="UTF-8" ?>
<body>
 <SiebelMessage>
 <MessageId></MessageId>
 <MessageType>Integration Object</MessageType>
 <IntObjectName>Account Interface</IntObjectName>
 <IntObjectFormat>Siebel Hierarchical</IntObjectFormat>
 <ListOfAccount_spcInterface>
 <Account>
 <Account Id>1-34Z</Account Id>
 </Account>
 </ListOfAccount_spcInterface>
```

```
 </SiebelMessage>
</body>
```

# Using a Siebel CRM Business Service to Query for an Account

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

The following details are for a request to query for an Account with an ID=1-6 value on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/v1.0/service/Siebel Account/QueryById?PrimaryRowId=1-6**
- HTTP Method: GET
- Content-Type: application/xml
- Authorization: Basic
- Request body:

  <xml>

  </xml>

# Accessing the Query By Example Method of the Siebel Account Business Service

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

The following details are for a request to query for an Account with a 3Com Name value on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/service/Siebel Account/QueryByExample**
- HTTP Method: GET
- Content-Type:  application/xml
- Authorization:  Basic
- Request body:

```
<?xml version="1.0" encoding="UTF-8" ?>
<body>
 <SiebelMessage>
 <MessageId></MessageId>
 <MessageType>Integration Object</MessageType>
 <IntObjectName>Account Interface</IntObjectName>
 <IntObjectFormat>Siebel Hierarchical</IntObjectFormat>
 <ListOfAccount_spcInterface>
 <Account>
 <Name>3Com</Name>
 </Account>
 </ListOfAccount_spcInterface>
 </SiebelMessage>
</body>
```

# Using a Siebel CRM Business Service to Insert an Account

You can use a Siebel CRM Business Service to insert an Account by sending an HTTP POST request to the resource's URI.

The following details are for a request to insert an Account with a REST Test Business Service Name value on the Siebel CRM Server:

- URI: **http://ServerName:port/siebel/v1.0/service/Siebel Account/Insert**
- HTTP Method: POST
- Content-Type: application/xml
- Authorization: Basic
- Request body:

```
<?xml version="1.0" encoding="UTF-8" ?>
<body>
 <SiebelMessage>
 <MessageId></MessageId>
 <MessageType>Integration Object</MessageType>
 <IntObjectName>Account Interface</IntObjectName>
 <IntObjectFormat>Siebel Hierarchical</IntObjectFormat>
 <ListOfAccount_spcInterface>
 <Account>
 <Account_spcId>2345</Account_spcId>
 <Name>REST Test Business Service</Name>
 </Account>
 </ListOfAccount_spcInterface>
 </SiebelMessage>
</body>
```

# 5 Using Siebel REST API For Siebel Clinical

## Using Siebel REST API For Siebel Clinical

This chapter describes Siebel Representational State Transfer (RESTful) Services for Siebel Clinical users and how to use them. It includes the following topics:



## 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.

## 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*.

# 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 User

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":
 {
 "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.