Oracle® Calendar

Application Developer's Guide

10*g* Release 1 (10.1.2)

B25489-01

June 2006

ORACLE®

Oracle Calendar Application Developer's Guide, 10*g* Release 1 (10.1.2)

B25489-01

Primary Author: Raymond Gallardo

Contributor: Bernard Desruisseaux, Daniel Fiore, Graham Gilmore, Jean-Philippe Guguy, Frederic Leblanc

Contents

4 Oracle Calendar SDK API Reference

11 Oracle Calendar Web Services Client-Side Java Implementation

12 Oracle Calendar Web Services Supported Data Components and Properties

Preface

This Preface contains these topics:

Audience

Oracle Calendar Application Developer's Guide is intended for any programmers and developers who want to use the Oracle Calendar SDK or the Oracle Calendar Web Services Toolkit to create custom applications that access Oracle Calendar.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, see the following manuals in the Oracle Collaboration Suite documentation set:

- *Oracle Calendar SDK Java API Reference*
- *Oracle Calendar Web Services Java API Reference*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Part I

Oracle Calendar SDK

This part of the Oracle Calendar Application Developer's Guide describes the Oracle Calendar SDK.

This part contains the following chapters:

1

Overview of Oracle Calendar SDK

The Oracle Calendar SDK is a set of functions, written in C/C++ with corresponding Java functions, that a developer can use to create applications that interface with Oracle Calendar. Using a native C interface, the SDK allows for implementation using any language that can call C functions natively.

Using standard iCalendar objects to represent meetings and events, a developer can use Oracle Calendar SDK functions to create programs that read/write calendar data, storing the information on the Oracle Calendar server.

In addition, a developer can use vCard objects with the Oracle Calendar SDK to create programs with address book functionality.

Examples of programs that can be created include custom interfaces to the Oracle Calendar server and migration utilities that allow for data extraction from any other system capable of producing iCalendar output.

SDK Contents

The Oracle Calendar SDK includes the following:

- Shared library implementing the APIs
- C header file
- Java .jar file
- Javadoc HTML documentation for the SDK, *Oracle Calendar SDK Java API Reference*
- Oracle Calendar client libraries
- Oracle Calendar ACE (authentication, compression, encryption) modules
- Sample/Demo programs.

2

Oracle Calendar SDK Implementation Considerations

This chapter discusses the following topics to be taken into consideration in your Oracle Calendar SDK implementations:

Best Practices

The Oracle Calendar SDK provides standards-based wrappers around a subset of Oracle Calendar core functions. It should be used for rapid development of utilities and applications that extend the existing Oracle Calendar applications, and not as a tool to replace existing interfaces or logic.

The SDK has been used to integrate with portals, FCGI-based applications, and even simple command-line scripts which might display a day's events. The SDK is best used to achieve a specific goal.

Character Sets

All SDK functions operate only on UTF-8 encoded text. All strings given to the SDK functions must be in UTF-8 and all strings returned by the SDK will be in UTF-8. For more information on UTF-8, refer to RFC 3629.

Security Model

There are two parts to the security model: storing and fetching events. These are handled by different security paradigms.

Security Model of Storing Events

The owner of an event can add or delete properties of that event. When an "ATTENDEE" property is created for a calendar user, the property is created with default values for its parameters. The owner of the event cannot modify the parameters of that property, only the user to whom it corresponds can do that.

When a user is updating their "ATTENDEE" property no error will be returned if there is an attempt to modify other event data, but the modifications will not occur.

It is possible for a user to refuse invitations from another user. In that case an "ATTENDEE" property will not be created for that user and the status for that user's handle will indicate that the invitation was refused. This may also occur when attempting to double book resources.

Security Model of Fetching Events

When fetching events the security model is based on the iCalendar classification of the event. Users grant other users different access levels to different classes of events. The three access levels are: no read access, read the start and end times of the event only, and read all details of the event. When fetching events with the SDK this results in some events for which only the "UID," "DTSTART," "DURATION" and "DTEND" properties will be returned. All other events will be invisible or all of their properties will be returned.

The Oracle Calendar SDK does not allow users to modify the security records which govern this behavior.

Other Security Issues

Unlike many Oracle products, the Oracle Calendar SDK (nor Oracle Calendar Web services) does not filter data entered by users for malicious HTML that could be used for breaching security layers, such as cross-site scripting attacks. As such, it is the responsibility of any application that retrieves data from the Oracle Calendar server to ensure that user data is properly filtered before it is displayed in a Web browser.

User Identification

The Oracle Calendar SDK identifies users with a user identification string. The format of this string flexible and allows you to specify a number of optional parameters. Depending on the server configuration, some of these options (such as the Node ID) may be required in the user identification string. The same user identification string format is used both at logon and when obtaining a handle; however not all options will be applicable in both cases.

Logging into the server as a resource is not supported, but it is possible to work as a designate for a resource.

Format of User Identification Strings

A user identification string consists of a user ID string followed by a question mark (?) and a series of key-value pairs. This series of pairs is called the extended data. (In the above example, the user ID has not been defined.)

The Oracle SDK can identify calendar users with the user ID string or a search string that looks for specific information in the extended data, such as the user's email address.

With a standalone Oracle Calendar server, the Oracle Calendar administrator sets the user ID string individually per user. As a result, the administrator may choose to leave the user ID string blank, as in the above example.

This user ID string is mapped to an LDAP attribute, typically `uid`. You may determine which attribute this is by looking at the file `ocal/misc/unison.ini`. In the LDAP section, look for the `attr_uid` key.

The key-value pairs of the extended data are separated by a delimiter. This delimiter is the character following the question mark (?) of the logon string. The delimiter may be any ASCII character except a digit, a letter, NUL, asterisk (*) or equal sign (=). Key-value pairs consist of a field name, followed by an equal sign (=), which is followed by the value. The value is a string that does not contain the delimiter character, the NUL character, and, for user identification strings, the slash (/) character. The logon string is terminated by a delimiter followed by a NUL character.

The following is a legal string for identifying a user:

```
?/S=Bunny/G=Bugs/
```

The field name G denotes the given name, and S denotes the surname. No user ID string is specified, so the extended data will be used to search for the user. (Note that if a search results in multiple matches, the SDK will return an error to the caller; a user ID is the best method of specifying a user, if it is available.) Even with no user ID, the question mark (?) still separates the user ID string from the extended data. The character immediately following the question mark, in this case a slash (/), is used as the delimiter. Note that the string ends with the delimiter character.

Any field used for identifying a user may be terminated with an asterisk (*), which is used as a wildcard. This is not available for specifying nodes. For example, the following will match the preceding user:

```
?/S=Bu*/G=Bugs/
```

Remember that if multiple users match a given search string, the SDK call will return an error.

Identification Strings of Oracle Calendar Resources

The following is an example of an identification string of a calendar resource:

```
?/RS=CA:MTRL:ConferenceRoomName/
```

Calendar resources use the same string format as those for calendar users, except that calendar resource identification strings must define at least the field "RS" that indicates the resource's name.

Logging into the Oracle Calendar server as a resource is not supported, but it is possible to work as a designate for a resource.

Syntax of Identification Strings

The following grammar (in ABNF form, as described in RFC 2234) describes legal identification strings. The description diverges from ABNF in that values in double quotes are case-sensitive; for example, field names must be in uppercase. Also, the delimiter character must be the same in all cases in a single string.

Example 2–1 Oracle Calendar SDK Identification Strings Grammar

```
logon-string = ( userid [ "?" DELIMITER node DELIMITER] %x00
```

```
                / "?" DELIMITER 1*(x400-field) [node DELIMITER] %x00
                / "?" DELIMITER resource-name DELIMITER [node DELIMITER] %x400
                / "?" DELIMITER email DELIMITER [node DELIMITER] %x00
                / "?" DELIMITER guid DELIMITER [node DELIMITER] %x00 )

userid = *( ALPHA / DIGIT / "-")
x400-field = ( surname / given-name / initials / generation / org-unit
             / organization / country / admin / private ) DELIMITER
email = "EMAIL=" 1*VALUE-CHAR
resource-name = "RS=" 1*VALUE-CHAR ["*"] DELIMITER
guid = "GUID=" 1*VALUE-CHAR

node = "ND=" node-number
node-number = 1*DIGIT
surname = "S=" 1*VALUE-CHAR ["*"]
given-name = "G=" 1*VALUE-CHAR ["*"]
initials = "I=" 1*VALUE-CHAR ["*"]
generation = "X=" 1*VALUE-CHAR ["*"]
org-unit = ( "OU1" / "OU2" / "OU3" / "OU4" ) "=" 1*VALUE-CHAR ["*"]
organization = "O=" 1*VALUE-CHAR ["*"]
country = "C=" 1*VALUE-CHAR ["*"] DELIMITER
admin = "A=" 1*VALUE-CHAR ["*"] DELIMITER
private = "P=" 1*VALUE-CHAR ["*"] DELIMITER

DELIMITER = %x01-%x29 / %x2B-%x2F / %x3A-%x3C / %x3E-%x40 / %x5B-%x60
            / %x7B-%x7F
VALUE-CHAR = %x01-29 / %x2B-2E / %x30-7F
```

Specifying a particular field more than once is, while redundant, still legal, although only the last field will be used.

Note also that the DELIMITER cannot be used as a VALUE-CHAR.

Also note that the SDK identification strings parser does not validate guid and email strings for correctness; it accepts these strings as long as it passes these strings on as-is, as long as they don't break the user string format.

The following are examples of identification strings:

Table 2–1 Oracle Calendar SDK Identification Strings

Type (as specified by logon-string)	Example
userid	`tjefferson`
x400-field	`?/G=GERMAINE/S=LAUZON/I=/X=/OU1=/OU2=/OU3=/OU4=/ O=Vision Corp./C=CA/A=/P=/`
resource-name	`?/RS=CA:MTRL:ConferenceRoomName/`
email	`?/EMAIL=fred.flintstone@bedrockquarry.com/`
guid	`?/GUID=F934D040EC9D4ABCE030578C82100D80/`

Date and Time Values

The Oracle Calendar SDK uses the iCalendar data types DATE, DATE-TIME and DURATION.

DATE

Identifies values that contain a calendar date. For example, September 28, 2002 would be represented by the following:

```
20020928
```

DATE-TIME

Identifies values that specify a precise calendar date and time of day. It may be in either floating time or UTC time. Floating time uses the user's timezone (the user's timezone preference stored on the Oracle Calendar server).

For example, the following floating time value represents September 29, 2002, at noon:

```
20020929T120000
```

The following UTC time value represents September 29, 2002 at 1700 UTC:

```
20020929T170000Z
```

The Oracle Calendar SDK disregards (and does not store in the Oracle Calendar store) any seconds (the last two digits) in DATE-TIME values. However, the seconds are still mandatory for this format.

DURATION

Identify properties that contain a duration of time. For example, the following represents five days and three hours in the future:

```
+P5DT3H
```

The following represents one week in the past:

```
-P1W
```

Alarms

Alarms are considered private to each user, so users cannot read or write alarms for each other. Since users cannot read each other's alarms it is not possible for users to do fetches by alarm range on each other's calendars. Any user may set an alarm for an event which they are attending, so the same events can have a different alarm when fetched by a different user.

Data Streams

By default, the SDK deals with MIME (see RFC 2045) encapsulated iCalendar and vCard objects for both input and output. A single request may fetch data from a list of calendars. A reply to such a request will consist of a separate iCalendar object for each calendar in the list, inside separate MIME parts. That is, a request for events from calendar A and calendar B results in a MIME stream of this form:

```
MIME envelope
--MIMEBOUNDARYasdfasdf
Content-type: text/calendar
Content-Transfer-Encoding: quoted-printable

BEGIN:VCALENDAR
events from calendarA
END:VCALENDAR
--MIMEBOUNDARYasdfasdf
Content-type: text/calendar
```

```
Content-Transfer-Encoding: quoted-printable

BEGIN:VCALENDAR
events from calendarB
END:VCALENDAR
--MIMEBOUNDARYasdfasdf--
```

A blank line separates the MIME header from the body (which in this case would be an iCalendar object).

The order of the iCalendar objects corresponds to the order of the calendars in the request list. If a request results in an empty solution set, the return stream will be an empty iCalendar object. If there is any sort of error with a calendar the iCalendar reply object corresponding to that calendar will be empty.

On a successful fetch the "VCALENDAR" may contain many "VEVENT" components, each containing the requested properties, if available. iCalendar allows these different components to contain information about different instances of the same event. The returned data may use any of the following methods to give instance specific information:

Data for each instance can be placed in a different "VEVENT" component, with a different "DTSTART".

Data for multiple instances can be placed in a single "VEVENT" by identifying instances with the properties "RRULE", "RDATE", "EXRULE" and "EXDATE"

Hybrids of the preceding two methods allow grouping of multiple instances which share all properties except their start time in a single "VEVENT" component, and returning many such components.

Note that the "DTSTART" property returned indicates the start time of the first instance identified in the "VEVENT" component in which it resides and not the start time of the first instance of the event in the Oracle Calendar store. Furthermore the number of "VEVENT" components returned in the calendar has no relation to the number of instances of the event. Consequently, when fetching events, if the recurrence identifying properties are not requested, there will be no way to determine how many instances exist, and to which instances each returned property applies.

When storing, data supplied to the Oracle Calendar SDK must consist of a single "VCALENDAR" component inside a single MIME part. The calendar may contain many "VEVENT" objects, but these must all be information about a single event. For example, this is a valid input:

```
Content-type: text/calendar
Content-Transfer-Encoding: 7bit

BEGIN:VCALENDAR
VERSION:2.0
BEGIN:VEVENT
event properties
END:VEVENT
BEGIN:VEVENT
event properties
END:VEVENT
END:VCALENDAR
```

Access Control

Access to data through the SDK is controlled by the Oracle Calendar server. It is based on the requester's identity and the data or operation being requested. The SDK provides an interface to request reading any combination of properties. Properties that the requesting user is not authorized to read will not be returned.

Users will only have privileges to modify the events to which they are invited, or which they own. If the user is the owner of the event they will have full privileges to modify the event (except for modifying other users' attendance information), otherwise if they are invited to the event they will have restricted privileges to modify information relating to their own attendance, such as acceptance and alarms.

Errors may occur for specific agendas when attempting to modify events or when creating events. These errors will be returned using a supplied array of status values, allowing the rest of the operation to proceed.

Multi-Threaded Applications

Session and handle usage has to be considered when designing multi-threaded applications with the C and Java Oracle Calendar SDK.

No two threads should concurrently use a session or handle object, even if the threads are performing operations on the same user's agenda or handle: The SDK does not support the concurrent use of sessions and handles.

If a possibility exists that two threads could use the same session object, it is the application's responsibility, using its own synchronization functionality, to guarantee that the same session will not be used concurrently by both threads.

The following scenario describes what might happen if a thread uses a session and a handle at the same time:

1. Both Thread1 and Thread2 use Session1.
2. Thread1 authenticates as User1.
3. Thread2 then deauthenticates and reauthenticates as User2.
4. Thread1 still assumes it is authenticated as User1 and inadvertently performs operations on User2.

3

Oracle Calendar SDK Supported Data Components and Properties

The Oracle Calendar SDK uses the iCalendar format for dealing with Calendar data. However, not all iCalendar data is actively supported by this version of the SDK. In particular, VFREEBUSY and VJOURNAL components are not supported.

Data for these properties will not always be preserved because events are shared entities in Oracle Calendar. Some properties are stored only per event rather than per instance so only one value is preserved. In particular, a calendar event with several attendees will appear in each of the attendee's agendas. However, that calendar event appears only once in the Oracle Calendar store.

For more information regarding data types, syntax, and other characteristics of iCalendar components and properties, see *RFC 2445 - Internet Calendaring and Scheduling Core Object Specification (iCalendar)*.

The Oracle Calendar SDK also uses the vCard format for dealing with contact information. For more information regarding characteristics of vCard components and properties, see *RFC 2426 - vCard MIME Directory Profile*.

The following tables and sections describe the iCalendar and vCard components and properties that the Oracle Calendar SDK supports. The Oracle Calendar SDK also provides Oracle-specific components and properties whose names begin with "X-ORACLE."

Components of iCalendar

The Oracle Calendar SDK supports the VEVENT and VTODO iCalendar components.

The following tables describe the component properties of VEVENT and VTODO. The following are clarifications of some of these table's headings and abbreviations:

- Columns labeled "Minimum Occurrences" indicate the minimum number of each component property (or property parameter) that the client must create or generate in order to add the specified component to the Calendar store.
- Columns labeled "Maximum Occurrence" indicate the maximum number of each component property (or property parameter) the specified component may contain.
- The symbol "n" indicates no limit of the maximum number of the specified property or parameter

VEVENT

Describes appointments, daily notes, day events, and holidays.

Table 3–1 Component Properties of VEVENT

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
ATTACH	0	1	file location of the attachment
ATTENDEE	0	n	CAL-ADDRESS
CATEGORIES	0	n	TEXT
CLASS	0	1	TEXT
COMMENT	0	n	TEXT
CONTACT	0	n	TEXT
CREATED	0	1	DATE-TIME
DESCRIPTION	0	1	TEXT
DTEND	1 (However, must not appear with DURATION)	1	DATE-TIME (default), DATE
DTSTAMP	0	1	DATE-TIME
DTSTART	1	1	DATE-TIME (default), DATE
DURATION	1 (However, must not appear with DTEND)	1	DURATION
EXDATE	0	1	DATE-TIME (default), DATE
EXRULE	0	1	RECUR
GEO	0	1	two semicolon-separated FLOAT values.
LAST-MODIFIED	0	1	DATE-TIME
LOCATION	0	1	TEXT
ORGANIZER	0	1	CAL-ADDRESS
PRIORITY	0	1	INTEGER
RDATE	0	n	DATE-TIME (default), DATE, PERIOD
RECURRENCE-ID	0	1	DATE-TIME
RELATED-TO	0	n	DATE-TIME (default), DATE
RESOURCES	0	n	TEXT
RRULE	0	1	RECUR
SEQUENCE	0	1	INTEGER
STATUS	0	1	TEXT
SUMMARY	0	1	TEXT
TRANSP	0	1	TEXT
UID	0	1	TEXT
URL	0	1	URI
X-ORACLE-CLASS	0	1	TEXT
X-ORACLE-EVENT-GUID	0	1	TEXT
X-ORACLE-EVENTINSTANCE-GUID	0	1	TEXT

Table 3–1 (Cont.) Component Properties of VEVENT

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
X-ORACLE-EVENTTYPE	0	1	TEXT
X-ORACLE-ISRTCENABLED	0	1	BOOLEAN
X-ORACLE-RTC-ATTENDEE-URL	0	1	URI
X-ORACLE-RTC-DIALININFO	0		TEXT
X-ORACLE-RTC-ENABLEDIRECTJOIN	0	1	BOOLEAN
X-ORACLE-RTC-HOST-URL	0	1	TEXT
X-ORACLE-RTC-MEETINGID	0	1	TEXT
X-ORACLE-RTC-PASSWORD	0	1	TEXT
X-ORACLE-RTC-PUBLISHATTENDEES	0	1	BOOLEAN
X-ORACLE-RTC-SECURITYTYPE	0	1	TEXT
X-ORACLE-RTC-SENDEMAILNOTIFICATION	0	1	BOOLEAN
X-ORACLE-RTC-SITEID	0	1	INTEGER
X-ORACLE-RTC-VERSION	0	1	TEXT

VALARM

Describes reminders for Calendar entries. Properties of VALARM include the type of reminder, such as a popup or an email, and the time before which the VALARM should notify the user of the Calendar event.

Table 3–2 Component Properties of VEVENT

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
ACTION	1	1	TEXT
ATTENDEE	0 (May appear only if ACTION is EMAIL)	n	CAL-ADDRESS
DESCRIPTION	Required only if ACTION is DESCRIPTION or EMAIL	1	TEXT
DURATION	0 (REPEAT must appear if DURATION appears)	1	DURATION
REPEAT	0 (DURATION must appear if REPEAT appears)	1	INTEGER
SUMMARY	Required only if ACTION is EMAIL	1	TEXT
TRIGGER	1	1	TRIGGER

VTODO

The VTODO component describes tasks stored in the Oracle Calendar server.

Table 3–3 Component Properties of VTODO

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
ATTENDEE	0	n	CAL-ADDRESS
CATEGORIES	0	n	TEXT
CLASS	0	1	TEXT
COMMENT	0	n	TEXT
COMPLETED	0	1	DATE-TYPE
CONTACT	0	1	TEXT
CREATED	0	1	DATE-TIME
DESCRIPTION	0	1	TEXT
DTSTAMP	0	1	DATE-TIME
DTSTART	0	1	DATE-TIME (default), DATE
DUE	1 (However, must not appear of DURATION appears)	1	DATE-TIME (default), DATE
DURATION	1 (However, must not appear if DUE appears)	1	DURATION
GEO	0	1	two semicolon separated FLOAT values
LAST-MODIFIED	0	1	DATE-TIME
LOCATION	0	1	TEXT
ORGANIZER	0	1	CAL-ADDRESS
PERCENT-COMPLETE	0	1	INTEGER
PRIORITY	0	1	INTEGER
RELATED-TO	0	n	TEXT
RESOURCES	0	n	TEXT
REQUEST-STATUS	0	1	TEXT
SEQUENCE	0	1	INTEGER
STATUS	0	1	TEXT
SUMMARY	0	1	TEXT
UID	0	1	TEXT
URL	0	1	URI
X-ORACLE-CLASS	0	1	TEXT

VEVENT, VTODO, VALARM Component Properties

The following describes the iCalendar and Oracle-specific component properties of the VEVENT, VTODO, and VALARM calendar components.

ACTION

This property defines the action to be invoked when an alarm is triggered. It can have one of the following values:

- AUDIO:
- DISPLAY:

- EMAIL:
- PROCEDURE:
- X-ORACLE-ALARM-DEFAULT: Specifies that a VEVENT or VTODO calendar component be created or modified to have an alarm based on the user's default alarm settings.
- X-ORACLE-ALARM-NONE: Specifies that a VEVENT or VTODO calendar component be created or modified without an alarm.
- X-ORACLE-SMS: Indicates that the VALARM should have the server send an out-of-band reminder (such as an SMS or email) according to the user's preferences

ATTACH

The current version of the Calendar SDK doesn't fully support the ATTACH property as stated in RFC 2045. Only file URIs are supported. The SDK must be configured to support the ATTACH property. This involves setting `tmpDirectoryPath` in the `SDK` section of the Oracle Calendar SDK configuration file. See Chapter 7, "Oracle Calendar SDK Configuration Settings" for more information about this setting. Temporary file attachments will be saved in the directory specified in tmpDirectoryPath when an event is exported.

When an event is imported or stored, the ATTACH property value is file location of the attachment. The following is an example of the ATTACH property of an event to be stored:

```
ATTACH:file:/tmp/my_attachment.txt
```

When an event is exported or fetched, if the configuration setting tmpDirectoryPath points to a valid directory, any attachments will be downloaded in subdirectories of this directory. These subdirectories will be in the following format:

```
./<UID>-<RECURRENCE_ID>
```

If tmpDirectoryPath does not point to a valid directory, attachments will not be downloaded.

The following is an example of the ATTACH property of a fetched event:

```
ATTACH:file:///c:/20050427T160642Z-45c-a90b-27b4a6d7-Oracle-20050427T040000Z/my_
attachment.txt
```

ATTENDEE

Defines an attendee within a calendar component.

When storing ATTENDEE properties, an attempt will be made to correlate attendee properties with Calendar users. CSDK_StoreEvents performs a look-up on the Oracle Calendar server to find the corresponding calendar user. Non-calendar users will still be invited (as "external attendees") when using CSDK_StoreEvents.

Look-ups are performed first using the X-ORACLE-GUID parameter, if present. If this fails, the user is treated as an external attendee. If the X-ORACLE-GUID parameter is not present, the value of the ATTENDEE property is used to do a look-up by email address of the calendar user.

To add a resource as an attendee, use either the X-ORACLE-GUID parameter, or set the CUTYPE parameter to "RESOURCE" and the CN parameter to the resource's name.

When retrieving data with the Oracle Calendar SDK, a property is generated for each ATTENDEE. The parameters PARTSTAT, ROLE, CUTYPE, and CN are obtained from the attendee and user information.

The following table describes the property parameters of ATTENDEE:

Table 3–4 Property Parameters of ATTENDEE

Property Parameter	Minimum Occurrences	Maximum Occurrences	Data Type
CN	0	1	TEXT
CUTYPE	0	1	TEXT
DELEGATED-FROM	0	1	TEXT
DELEGATED-TO	0	1	TEXT
DIR	0	1	URI
MEMBER	0	1	TEXT
PARTSTAT	0	1	TEXT
ROLE	0	1	TEXT
RSVP	0	1	TEXT
SENT-BY	0	1	TEXT
X-ORACLE-GUID	0	1	TEXT
X-ORACLE-PERSONAL-COMMENT	0	1	A BASE64 encoded character string, as defined by [RFC 2045]
X-ORACLE-PERSONAL-COMMENT-ISDIRTY	0	1	TEXT
X-ORACLE-PERSONAL-COMMENT-RTF	0	1	TEXT
X-ORACLE-SHOWASFREE	0	1	TRUE/FALSE

CATEGORIES

Defines the categories for a calendar component.

When using CSDK_StoreEvents, the CATEGORIES value is stored on the server and will be returned by the various CSDK_FetchEvents functions. (The property X-ORACLE-EVENTTYPE is used with CSDK_StoreEvents to specify the event type and the same values are recognized).

Calling one of the CSDK_FetchEventsBy functions will return a user-specified value (which may have been stored using the Oracle Calendar SDK or another client).

CLASS

Defines the access classification for a calendar component.

This property is mapped to an Oracle Calendar server access level. The mapping between the CLASS value and the Oracle Calendar server access level is as follows:

Table 3–5 Mapping Between CLASS Values and Oracle Calendar Server Access Levels

iCalendar CLASS Value	Oracle Calendar Server Access Level
CLASS:PUBLIC	PUBLIC
CLASS:PRIVATE	PERSONAL
CLASS:CONFIDENTIAL	CONFIDENTIAL

Table 3–5 (Cont.) Mapping Between CLASS Values and Oracle Calendar Server Access

iCalendar CLASS Value	Oracle Calendar Server Access Level
CLASS:PRIVATE X-ORACLE-CLASS:NORMAL	NORMAL

If the CLASS property is not specified, the access level for the calendar component is PUBLIC. This is the behavior as defined in RFC 2445.

This property is stored per event, which implies that all recurrence instances of an event must have the same CLASS value.

COMMENT

Specifies non-processing information intended to provide a comment to the calendar user.

COMPLETED

Defines the date and time that a task was actually completed.

CONTACT

Represents contact information or alternately a reference to contact information associated with the calendar component.

CREATED

Specifies the date and time that the calendar information was created by the calendar user agent in the calendar store. The date and time is a UTC value.

DESCRIPTION

Provides a more complete description of the calendar component than that provided by the SUMMARY property.

This is set to the Event's details. It will be truncated if it is longer than 32 Kb. This property is stored per instance when calling CSDK_StoreEvents.

DTEND

Specifies the date and time that a calendar component ends.

DTSTAMP

Indicates the date/time that the instance of the iCalendar object was created. The value must be specified in the UTC time format. This property is different than the CREATED and LAST-MODIFIED properties. These two properties are used to specify when the particular Calendar data in the calendar store was created and last modified. This is different than when the iCalendar object representation of the Calendar service information was created or last modified.

DTSTART

Specifies when the Calendar component begins.

If DTEND is present, it will be used to calculate the event duration; the actual end time is not stored. As event times are measured in minutes, the start time and duration will have their 'seconds' component set to zero.

DUE

Defines the date and time that a task is expected to be completed.

DURATION

Specifies a positive duration of time.

For example, a duration of 15 days, 5 hours and 20 seconds would be represented as P15DT5H0M20S. A duration of 7 weeks would be represented as P7W

EXDATE

Defines the list of date/time exceptions for a recurring calendar component. The following is an example of this property:

```
EXDATE:19960402T010000Z,19960403T010000Z,19960404T010000Z
```

EXRULE

Defines a rule or repeating pattern for an exception to a recurrence set. For example, the following excludes, for a recurrence set, dates that occur every other week on Tuesday and Thursday for 4 occurrences:

```
EXRULE:FREQ=WEEKLY;COUNT=4;INTERVAL=2;BYDAY=TU,TH
```

The following excludes dates that occur every day for 10 occurrences:

```
EXRULE:FREQ=DAILY;COUNT=10
```

The following excludes dates that occur every year in June and July for 8 occurrences

```
EXRULE:FREQ=YEARLY;COUNT=8;BYMONTH=6,7
```

GEO

Specifies information related to the global position for the activity specified by a calendar component. The property value specifies latitude and longitude, in that order.

LAST-MODIFIED

Specifies the date and time that the information associated with the calendar component was last revised in the calendar store. This is analogous to the modification date and time for a file in the file system. The property value must be specified in the UTC time format.

LOCATION

Defines the intended venue for the activity defined by a calendar component.

ORGANIZER

Defines the organizer for a calendar component. Uses the same property parameters as ATTENDEE.

PERCENT-COMPLETE

Used by an assignee or delegatee of a VTODO to convey the percent completion of a task to the Organizer. The property value is a positive integer between zero and one hundred.

PRIORITY

Defines the relative priority for a calendar component.

This property is mapped to one of the Oracle Calendar server's five priority values. This property is stored per event.

RDATE

Defines the list of date/times for a recurrence set. The following are examples of this property:

```
RDATE:19970714T123000Z

RDATE;TZID=US-EASTERN:19970714T083000

RDATE;VALUE=DATE:19970101,19970120,19970217,19970421,19970526,19970704,
  19970901,19971014,19971128,19971129,19971225
```

RDATEs of the type VALUE=PERIOD are not supported.

RECURRENCE-ID

Used in conjunction with the UID property to identify a specific instance of a recurring VEVENT or VTODO calendar component. The property value is the effective value of the DTSTART property of the recurrence instance.

RELATED-TO

Represent a relationship or reference between one calendar component and another. It consists of the persistent, globally unique identifier of another calendar component. This value would be represented in a calendar component by the UID property.

The Oracle Calendar SDK stores this value as-is, but does not use it; there is no business logic associated with it.

REQUEST-STATUS

Ignored by the Oracle Calendar SDK.

RESOURCES

Defines the equipment or resources anticipated for an activity specified by a calendar entity.

RRULE

Defines a rule or repeating pattern for recurring events, to-dos, or time zone definitions.

REPEAT

Defines the number of time the alarm should be repeated, after the initial trigger.

If the alarm triggers more than once, then this property must be specified along with the "DURATION" property.

SEQUENCE

Defines the revision sequence number of the calendar component within a sequence of revisions.

STATUS

Defines the overall status or confirmation for the calendar component.

A tentative event will have a TENTATIVE status. Non-tentative events will be marked as CONFIRMED. No other STATUS values are generated.

SUMMARY

Defines the title of the event or instance.

TRANSP

Defines whether an event is transparent or not to busy time searches. Time transparency is the characteristic of an event that determines whether it appears to consume time on a calendar.

Events that consume actual time for the individual or resource associated with the calendar are recorded as OPAQUE, allowing them to be detected by free-busy time searches. Other events, which do not take up the individual's (or resource's) time are recorded as TRANSPARENT, making them invisible to free-busy time searches.

The Oracle Calendar SDK keeps track of transparency separately per attendee with the X-ORACLE-SHOWASFREE parameter on the ATTENDEE property, which can have the values FREE, BUSY, OUT, or TENTATIVE.

However, the Oracle Calendar SDK uses the TRANSP property only on output. It is set to OPAQUE for regular events and TRANSPARENT for day events, daily notes, and holidays.

TRIGGER

Specifies when an alarm will trigger.

UID

Defines the persistent, globally unique identifier for the calendar component.

If a UID is not specified in stored data the Oracle Calendar server will assign a UID. When using CSDK_StoreEvents, the generated UIDs are returned as part of the results in CSDKRequestResult.

URL

Defines a Uniform Resource Locator (URL) associated with the iCalendar object.

X-ORACLE-CLASS

This property defines an Oracle-specific access classification for an iCalendar component.

This property describes the access classification specific to Oracle for the iCalendar component. Currently, this property is only being used for iCalendar components with the "NORMAL" access level. For interoperability and security considerations, "X-ORACLE-CLASS:NORMAL" is always returned with "CLASS:PRIVATE".

X-ORACLE-EVENT-GUID

Uniquely identifies VEVENT components.

X-ORACLE-EVENTINSTANCE-GUID

Uniquely identifies VEVENT instances.

X-ORACLE-EVENTTYPE

Identifies the type of event that the VEVENT represents. The property can be specified once in the VEVENT component. Possible values are "daily note", "holiday", "day event", and "appointment".

The following is an example of this property:

```
X-ORACLE-EVENTTYPE:DAY EVENT
```

X-ORACLE-ISRTCENABLED

Indicates that an instance is Web conference-enabled. This property is generated by the Oracle Calendar server and used by Calendar clients. This property can be specified in the VEVENT calendar component.

X-ORACLE-ORGANIZATION

Specifies the organization associated with the task described by the VTODO. The property may be specified multiple times in a VTODO calendar component.

The following is an example of this property:

```
X-ORACLE-ORGANIZATION:ACME, Inc.
```

X-ORACLE-RTC-ATTENDEE-URL

This property may be specified once in the VEVENT calendar component.

X-ORACLE-RTC-DIALININFO

Specifies dial-in information required by attendees to join a teleconference, such as the phone number and conference ID. This property may be specified once in the VEVENT calendar component.

The following is an example of this property:

```
X-ORACLE-RTC-DIALININFO:1-999-999-9999 Conference Id: 999999
```

X-ORACLE-RTC-ENABLEDIRECTJOIN

A boolean value that specifies whether Calendar Web services should include Web conference details (including the Join URL) to allow attendees to join a conference through an email invitation. This property may be specified once in the VEVENT calendar component.

The following is an example of this property:

```
X-ORACLE-RTC-ENABLEDIRECTJOIN:TRUE
```

X-ORACLE-RTC-HOST-URL

Specifies the URL of the Web page hosting the Web conference associated with this calendar component instance. This value is set by Web conference. This property may be specified once in the VEVENT calendar component.

The following is an example of this property:

```
X-ORACLE-RTC-HOST-URL:http://www.example.com
```

X-ORACLE-RTC-MEETINGID

Specifies the Web conference ID associated with the VEVENT calendar component instance. The value is strictly generated by the Web conference server. This property may be specified once in the VEVENT calendar component.

X-ORACLE-RTC-PASSWORD

The optional key (the password) of a Web conference. The property can be specified in the VEVENT calendar component.

X-ORACLE-RTC-PUBLISHATTENDEES

Currently not supported by the Oracle Calendar server and its clients. The Oracle Calendar server will always set this value to TRUE. Oracle Calendar server clients should not expose this property to users. This property may be specified once in the VEVENT calendar component.

X-ORACLE-RTC-SECURITYTYPE

Indicates the security type of an Web conference enabled instance. Possible types are "restricted", "regular", and "public". The property may be specified once in the VEVENT calendar component. This property may be specified once in the VEVENT calendar component.

X-ORACLE-RTC-SENDEMAILNOTIFICATION

If set to TRUE, specifies that the Calendar client should create the notification email addressed to the attendees of the conference. This property may be specified once in the VEVENT calendar component.

X-ORACLE-RTC-SITEID

Specifies the site ID associated with the Web conference represented by the VEVENT. The value for this property is generated by the Oracle Calendar server. Clients will not access this value. An administrator will be able to change the designated value in

order to track sites integrated with Web conferencing. This property may be specified once in the VEVENT calendar component.

X-ORACLE-RTC-VERSION

Used by the Oracle Calendar server to ensure that it supports this type of Web conference. The property may be specified once in the "VEVENT" calendar component.

Property Parameters

The following describes iCalendar and Oracle-specific property parameters of CAL-ADDRESS (the data type of ATTENDEE and ORGANIZER).

CUTYPE

Indicates the type of calendar user.

CN

Common or displayable name associated with the component property (in this case, ATTENDEE or ORGANIZER).

DELEGATED-FROM

Indicates whom the request was delegated from.

DELEGATED-TO

Indicates the calendar users that the original request was delegated to.

DIR

Indicates the URI that points to the directory information corresponding to the ATTENDEE or ORGANIZER.

MEMBER

Indicates the groups that the ATTENDEE or ORGANIZER belongs to.

PARTSTAT

Indicates the participation status of the ATTENDEE or ORGANIZER.

This parameter may have the following values, which have the following responses and replies from Oracle Connector for Outlook and the Oracle Calendar native client:

Table 3–6 PARTSTAT Possible Values

ATTENDEE Property Parameters	Native Client Reply	OCFO Response
PARTSTAT=NEEDS-ACTION	Decide later	None
PARTSTAT=ACCEPTED	Accepted	Accepted
PARTSTAT=DECLINED	Declined	Declined

Table 3–6 (Cont.) PARTSTAT Possible Values

ATTENDEE Property Parameters	Native Client Reply	OCFO Response
PARTSTAT=DECLINED X-ORACLE-UNAVAILABLE=TRUE (This is only applicable when connecting to Release 2 (9.0.4) of the Oracle Calendar server)	Declined; would prefer another time	Declined
PARTSTAT=TENTATIVE	Accepted; would prefer another time	Tentative

Note that X-ORACLE-UNAVAILABLE is only meaningful when set to TRUE and PARTSTAT is set to DECLINED.

ROLE

The intended role that the ATTENDEE or ORGANIZER will have in the calendar component.

RSVP

Indicates whether the ATTENDEE should reply or contact the organizer of the calendar event.

SENT-BY

Indicates who is acting on behalf of the ATTENDEE or ORGANIZER.

X-ORACLE-GUID

Uniquely identifies Oracle Collaboration Suite users.

The following is an example of this property:

```
ATTENDEE;X-ORACLE-GUID=D99DBDBAAF8D6D3DE0340003BA18F2E9:mailto:
  john.doe@example.com
```

X-ORACLE-PERSONAL-COMMENT

Specifies the personal comment of a Calendar user. This parameter can be specified in the ATTENDEE property.

The value type of this property is a BASE64 encoded character string as defined by RFC 2045.

The following is an example of an attendee component with the X-ORACLE-PERSONAL-COMMENT property defined:

```
ATTENDEE;X-ORACLE-PERSONAL-COMMENT=
  RoaXMgaXMgYSBtdWx0aS1saW5lXG4NCiAgcGVyc29uYWwgYW5ub3RhdGlvbi4gV2l0aC
  BhIGJ1bmNoIG9mIHNwZWNpYWxcbg0KICBjaGFyIHN1Y2ggYXMgY29tbWEgKCwpIHNlbWkt
  Y29sb24gKDspLCBhbmQgZG91YmxlLQ0KICBxdW90ZSAoXCIpLiI=:
  MAILTO:john.doe@example.com
```

X-ORACLE-PERSONAL-COMMENT-ISDIRTY

If set to TRUE, specifies that the DESCRIPTION property of the VEVENT component was modified after the value of the X-ORACLE-PERSONAL-COMMENT and

X-ORACLE-PERSONAL-COMMENT-RTF parameters were last modified. This parameter can be specified on the ATTENDEE property.

X-ORACLE-PERSONAL-COMMENT-RTF

Specifies the personal comment in Rich Text Format (RTF) of the Calendar user. This parameter can be specified in the ATTENDEE property.

X-ORACLE-SHOWASFREE

Specifies whether an event is transparent or not to searches for busy times for a specific attendee. it can have one of the following values:

- FREE
- BUSY
- OUT
- TENTATIVE

Components of vCard

Describes business and personal contacts in a Calendar user's address book. The following is a sample of a vCard:

```
BEGIN:VCARD
UID:ORACLE:CALSERV:CONTACT/AAAAAAAABBBBBBBBCCCCCCCCDDDDDDDD
REV:20040802T195051Z
EMAIL;TYPE=PREF;TYPE=INTERNET:calendar.user@my-company.com
ADR;TYPE=X-ORACLE-OTHER:;;;;;;;
ADR;TYPE=HOME:;;;;;;;
ADR;TYPE=WORK:;;;;;;;
N:User;Calendar;;;
FN:User\, Calendar
VERSION:3.0
END:VCARD
```

Table 3–7 Component Properties of vCard

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
ADR	1	n	TEXT values separated by semicolons
AGENT	0	1	vCard (default), TEXT, URI
BDAY	0	1	DATE (default), DATE-TIME
CATEGORIES	0	n	TEXT values separated by commas
CLASS	0	1	TEXT
EMAIL	0	1	TEXT
FN	1	1	TEXT
GEO	0	1	two FLOAT values separated by semicolons
KEY	0	1	BINARY (default), TEXT
LABEL	0	1	TEXT
LOGO	0	1	BINARY (default), URI
MAILER	0	1	TEXT

Table 3–7 (Cont.) Component Properties of vCard

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
N	1	1	TEXT (can have multiple values)
NICKNAME	0	1	TEXT values separated by a commas
NOTE	0	1	TEXT
ORG	0	1	TEXT values separated by semicolons
PHOTO	0	1	BINARY (default), URI
PRODID	0	1	TEXT
REV	0	1	DATE-TIME (default), DATE
ROLE	0	1	TEXT
SORT-STRING	0	1	TEXT
TEL	0	n	phone-number value
TITLE	0	1	TEXT
TZ	0	1	UTC-offset value (default), TEXT
UID	0	1	TEXT
URL	0	1	URI
VERSION	1	1	TEXT

vCard Component Properties

The following describes the iCalendar and Oracle-specific component properties of vCard.

ADR

Specifies the components of the delivery address for the vCard object.

AGENT

Specifies information about another person who will act on behalf of the individual or resource associated with the vCard.

BDAY

Specifies the birth date of the object the vCard represents. The default is a single date value. It can also be reset to a single date-time value.

CATEGORIES

Specifies application category information about the vCard.

CLASS

Specifies the access classification for a vCard object.

EMAIL

Specifies email address for communication with the object the vCard represents.

FN

Specifies the formatted text corresponding to the name of the object the vCard represents.

GEO

Specifies information related to the global positioning of the object the vCard represents.

KEY

Specifies a public key or authentication certificate associated with the object that the vCard represents.

LABEL

Specifies the formatted text corresponding to the delivery address of the object the vCard represents. It can include the type parameter TYPE to specify the delivery label type.

LOGO

Specifies a graphic image of a logo associated with the object the vCard represents.

MAILER

Specifies the type of electronic mail software that is used by the individual associated with the vCard.

N

Specifies the components of the name of the object the vCard represents.

NICKNAME

Specifies the descriptive name given instead of or in addition to the one that the vCard represents. It can also be used to specify a familiar form of a proper name specified by the FN or N types.

NOTE

Specifies supplemental information or a comment that is associated with the vCard.

ORG

Specifies the organizational name and units associated with the vCard.

PHOTO

Specifies an image or photograph information that annotates some aspect of the object the vCard represents. The default is a binary value. It can also be reset to uri value.

PRODID

Specifies the identifier for the product that created the vCard object.

REV

Specifies revision information about the current vCard.

ROLE

Specify information concerning the role, occupation, or business category of the object the vCard represents.

SORT-STRING

Specifies the family name or given name text to be used for national-language-specific sorting of the FN and N types.

TEL

Specifies the telephone number of the object the vCard represents.

TZ

Specifies information related to the time zone of the object the vCard represents.

SOUND

This property is not supported.

TITLE

Specifies the job title, functional position or function of the object the vCard represents.

UID

Specifies a globally unique identifier corresponding to the individual or resource associated with the vCard component.

URL

Specifies a uniform resource locator associated with

the object that the vCard refers to.

VERSION

Specifies the version of the vCard specification used to format this vCard. The property must be present in the vCard object. The value must be "3.0".

4

Oracle Calendar SDK API Reference

This chapter contains detailed information on functions included with the Oracle Calendar SDK.

Functions

This section provides details on the following functions:

Table 4–1 Functions

Function Declaration	Description
CAPIStatus CSDK_AddConditionToQuery (CSDKQuery in_query, CSDKCondition *in_condition, CSDKOperator in_operator)	Adds a condition to a query object.
CAPIStatus CSDK_Authenticate (CAPISession in_session, CAPIFlag in_flags, const char *in_user, const char *in_password)	Authenticates a calendar user.
CAPIStatus CSDK_ConfigureACE (CAPISession in_session, CAPIFlag in_flags, const char *in_authenticationMechanism, const char *in_compressionMechanism, const char *in_encryptionMechanism)	Configures the given session to use specific ACE (Authentication, Compression, and Encryption) mechanisms between the SDK client and the Calendar server.
CAPIStatus CSDK_Connect (CAPISession in_session, CAPIFlag in_flags, const char *in_host)	Establishes a connection with a calendar service.
CAPIStatus CSDK_ConnectAsSysop (CAPISession in_session, CAPIFlag in_flags, const char *in_host, const char *in_nodeId, const char *in_password)	Logs on as SYSOP; once logged on, SYSOP can assume the identity of any user on the same node by calling CSDK_SetIdentity().
CAPIStatus CSDK_CreateCallbackStream (CAPISession in_session, CAPIStream *out_stream, CAPICallback in_sendCallback, void *in_sendUserData, CAPICallback in_recvCallback, void *in_recvUserData, CAPIFlag in_flags)	Creates a callback stream that can be used to either supply data to, or receive data from, the SDK.
CAPIStatus CSDK_CreateFileStreamFromFilenames (CAPISession in_session, CAPIStream *out_stream, const char *in_readFileName, const char *in_readMode, const char *in_writeFileName, const char *in_writeMode, CAPIFlag in_flags)	Creates a file stream to allow the SDK to read from or write to files.
CAPIStatus CSDK_CreateMemoryStream (CAPISession in_session, CAPIStream *out_stream, const char *in_readBuffer, const char **out_writeBufferPtr, CAPIFlag in_flags)	Creates a memory stream, which uses data buffers to pass data between your application and the SDK.
CAPIStatus CSDK_CreateQuery (CSDKCondition *in_condition, CSDKQuery *out_query)	Creates a query object to be used with CSDK_FetchEventsByQuery or CSDK_FetchContactsByQuery.
CAPIStatus CSDK_CreateSession (CAPIFlag in_flags, CAPISession *out_session)	Creates a new session.
CAPIStatus CSDK_Deauthenticate (CAPISession in_session, CAPIFlag in_flags)	Deauthenticates the current user.
CAPIStatus CSDK_DeleteContacts (CAPISession in_session, CAPIFlag in_flags, CAPIUIDSet in_UIDs, CSDKRequestResult *out_requestResult)	Deletes vCards specified by a set of UIDs.

Table 4–1 (Cont.) Functions

Function Declaration	Description
CAPIStatus CSDK_DeleteEvents (CAPISession in_session, CAPIFlag in_flags, CAPIUIDSet in_UIDs, const char *in_RECURRENCEID, int in_modifier, CSDKRequestResult *out_requestResult)	Deletes specified events; must be acting as the event owner for this to succeed.
CAPIStatus CSDK_DeleteTasks (CAPISession in_session, CAPIFlag in_flags, CAPIUIDSet in_UIDs, CSDKRequestResult *out_requestResult)	Deletes tasks from the current user's agenda.
CAPIStatus CSDK_DestroyHandle (CAPISession in_session, CAPIHandle *in_handle)	Destroys one handle.
CAPIStatus CSDK_DestroyMultipleHandles (CAPISession in_session, CAPIHandle *in_handles, int in_numHandles, CAPIFlag in_flags)	Destroys multiple handles returned by calls to CSDK_GetHandle().
CAPIStatus CSDK_DestroyMultipleStreams (CAPISession in_session, CAPIStream *in_streams, int in_numStreams, CAPIFlag in_flags)	Destroys streams created by the various CSDK_Create...Stream functions.
CAPIStatus CSDK_DestroyQuery (CSDKQuery *io_query)	Destroys a query object created by CSDK_CreateQuery.
CAPIStatus CSDK_DestroyResult (CSDKRequestResult *io_requestResult)	Disposes of all the results in in_requestResult.
CAPIStatus CSDK_DestroySession (CAPISession *io_session)	Destroys a session.
CAPIStatus CSDK_DestroyStream (CAPISession in_session, CAPIStream *io_stream)	Destroys a stream created by any of the various CSDK_Create...Stream functions.
CAPIStatus CSDK_Disconnect (CAPISession in_session, CAPIFlag in_flags)	Disconnects from the Oracle Calendar server.
CAPIStatus CSDK_FetchContactsByQuery (CAPISession in_session, CAPIFlag in_flags, CSDKQuery in_query, const char **in_requestProperties, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Fetches contacts which satisfy the conditions specified in the query.
CAPIStatus CSDK_FetchContactsByUID (CAPISession in_session, CAPIFlag in_flags, CAPIUIDSet in_UIDs, const char **in_requestProperties, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Fetches vCards from an authenticated connection.
CAPIStatus CSDK_FetchEventsByAlarmRange (CAPISession in_session, CAPIFlag in_flags, CAPIHandle *in_agendas, const char *in_start, const char *in_end, const char **in_requestProperties, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Fetches events which have alarms (reminders) that will trigger within the time range specified; the end of the time range is exclusive.
CAPIStatus CSDK_FetchEventsByRange (CAPISession in_session, CAPIFlag in_flags, CAPIHandle *in_agendas, const char *in_start, const char *in_end, const char **in_requestProperties, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Fetches events which occur within the time range specified.
CAPIStatus CSDK_FetchEventsByUID (CAPISession in_session, CAPIFlag in_flags, CAPIHandle in_agenda, CAPIUIDSet in_UIDs, const char *in_RECURRENCEID, int in_modifier, const char **in_requestProperties, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Fetches events by their UIDs.
CAPIStatus CSDK_FetchTasksByAlarmRange (CAPISession in_session, CAPIFlag in_flags, CAPIHandle *in_handles, const char *in_start, const char *in_end, const char **in_requestProperties, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Fetches tasks that have alarms (reminders) that will trigger within the time range specified (the end of the time range is exclusive).
CAPIStatus CSDK_FetchTasksByRange (CAPISession in_session, CAPIFlag in_flags, CAPIHandle *in_handles, const char *in_start, const char *in_end, const char **in_requestProperties, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Fetches tasks which are active within the time range specified (the end of the time range is exclusive).

Table 4–1 (Cont.) Functions

Function Declaration	Description
CAPIStatus CSDK_FetchTasksByUID (CAPISession in_session, CAPIHandle in_handle, CAPIFlag in_flags, CAPIUIDSet in_UIDs, const char **in_requestProperties, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Retrieves tasks with given UIDs on the given agenda.
CAPIStatus CSDK_GetCapabilities (CAPISession in_session, CAPICapabilityID in_capabilityID, CAPIFlag in_flags, const char **out_value)	Returns information about this SDK release and/or the Oracle Calendar server.
CAPIStatus CSDK_GetFirstFailure (CSDKRequestResult in_requestResult, CAPIHandle *out_user, const char **out_uid, CAPIStatus *out_status)	Returns the first failure obtained from the function from which in_requestResult was returned.
CAPIStatus CSDK_GetFirstParseError (CSDKRequestResult in_requestResult, CAPIStatus *out_status, const char **out_errorBuffer, const char **out_errorLocation, const char **out_message)	Returns the first parsing error obtained from a request result.
CAPIStatus CSDK_GetFirstResult (CSDKRequestResult in_requestResult, CAPIHandle *out_user, const char **out_uid, CAPIStatus *out_status)	Returns the first result obtained from the function from which in_requestResult was returned.
CAPIStatus CSDK_GetHandle (CAPISession in_session, const char *in_user, CAPIFlag in_flags, CAPIHandle *out_handle)	Returns a handle to a particular user's calendar store.
CAPIStatus CSDK_GetHandleInfo (CAPISession in_session, CAPIHandle in_handle, CAPIFlag in_flags, const char **out_info)	Returns information about the agenda of the supplied handle.
CAPIStatus CSDK_GetNextFailure (CSDKRequestResult in_requestResult, CAPIHandle *out_user, const char **out_uid, CAPIStatus *out_status)	Returns the next failure contained in a CSDKRequestResult.
CAPIStatus CSDK_GetNextParseError (CSDKRequestResult in_requestResult, CAPIStatus *out_status, const char **out_errorBuffer, const char **out_errorLocation, const char **out_message)	Returns the next parsing error obtained from a request result.
CAPIStatus CSDK_GetNextResult (CSDKRequestResult in_requestResult, CAPIHandle *out_user, const char **out_uid, CAPIStatus *out_status)	Returns the next result contained in a CSDKRequestResult.
CSDK_GetStatusCode (CAPIStatus in_status, int *out_statusCode)	A status returned by the CALENDAR_SDK is composed of a status code and some extra bits giving extra context to the error that occurred.
CSDK_GetStatusLevels (CAPIStatus in_status, unsigned long *out_field1, unsigned long *out_field2, unsigned long *out_field3, unsigned long *out_field4, unsigned long *out_field5)	Decomposes a CAPIStatus into its subparts; each part of the status code specifies more precisely the actual error.
CSDK_GetStatusString (CAPIStatus in_status, const char **out_errorString)	Returns a read-only string representation of a CAPIStatus (this is generally more useful than the numeric representation).
CAPIStatus CSDK_SetConfigFile (const char *in_configFileName, const char *in_logFileName)	Allows the SDK to read configuration settings that control error logging and the other configuration parameters listed in the "Configuration" section of this manual.
CAPIStatus CSDK_SetIdentity (CAPISession in_session, const char *in_user, CAPIFlag in_flags)	Allows an authenticated user to work on behalf of another calendar user or resource.
CAPIStatus CSDK_StoreContacts (CAPISession in_session, CAPIFlag in_flags, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Stores vCards on a server through an authenticated connection by in_session; The vCards must be passed in via a CAPIStream.
CAPIStatus CSDK_StoreEvents (CAPISession in_session, CAPIFlag in_flags, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	This function reads one VCALENDAR object from in_stream and stores each contained VEVENT on the server.
CAPIStatus CSDK_StoreTasks (CAPISession in_session, CAPIFlag in_flags, CAPIStream in_stream, CSDKRequestResult *out_requestResult)	Creates/modifies tasks on the current user's agenda depending on the store flag passed in.

CSDK_AddConditionToQuery

Adds a condition to a query object.

```
CAPIStatus CSDK_AddConditionToQuery (
  CSDKQuery in_query,
  CSDKCondition * in_condition,
  CSDKOperator in_operator
)
```

Each query may have multiple conditions, each AND'ed or OR'ed with the previous condition(s). There is no way to group conditions, and the OR operator (CSDK_LOP_OR) has a higher priority than the AND operator (CSDK_LOP_AND). Thus, C1 OR C2 AND C3 evaluates as (C1 OR C2) AND C3.

Parameters

in_query
A query object created by CSDK_CreateQuery

in_condition
Condition to add to query

in_operator
Specifies the operator to use between existing conditions and this one (for example, "OR", "AND")

Returns

CAPIStatus

Equivalent Java Method

oracle.calendar.sdk.Query.addCondition()

CSDK_Authenticate

Authenticates a calendar user.

```
CAPIStatus CSDK_Authenticate (
  CAPISession in_session,
  CAPIFlag in_flags,
  const char * in_user,
  const char * in_password
)
```

This must be done prior to making any calls to store or fetch data.

Refer to the section on User Identification for the format of the in_user parameter.

Parameters

in_session
Login session handle

in_flags
Bit flags modifying behavior. This must be CSDK_FLAG_NONE currently.

in_user
Must be a null-terminated string. Refer to the "User Identification" section for syntax.

in_password
User's password. May be NULL.

Returns

CAPIStatus

Sample

Connect to a server running on the default port of calserver.acme.com to authenticate as userid "keithm" using default ACE settings When no node is specified, either a master node or default node must be configured on the specified host):

```
{
    CAPISession mySession = CSDK_SESSION_INITIALIZER;
    status = CSDK_CreateSession(&mySession);
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_Connect(mySession, CAPI_FLAG_NONE, "calserver.acme.com");
    }
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_Authenticate(mySession,
                                     CAPI_FLAG_NONE,
                                     "keithm",
                                     "abcdefg");
    }
}
```

Sample

Connect to a server running on the default port of calserver.acme.com to authenticate as user "Keith MacDonald" using default ACE settings:

```
{
    CAPISession mySession = CSDK_SESSION_INITIALIZER;
    status = CSDK_CreateSession(&mySession);
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_Connect(mySession, CAPI_FLAG_NONE, "calserver.acme.com");
    }
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_Authenticate(mySession,
                                     CAPI_FLAG_NONE,
                                     "?/S=MacDonald/G=Keith/ND=200/",
                                     "abcdefg");
    }
}
```

Sample

Connect to a server running on the default port of calserver.acme.com to authenticate as userid keithm on node 200 using default ACE settings:

```
{
    CAPISession mySession = CSDK_SESSION_INITIALIZER;
    status = CSDK_CreateSession(&mySession);
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_Connect(mySession, CAPI_FLAG_NONE, "calserver.acme.com");
    }
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_Authenticate(mySession,
                                     CAPI_FLAG_NONE,
                                     "keithm?/ND=200/",
                                     "abcdefg");
    }
}
```

Sample

Connect to a server running on port 12345 of calserver.acme.com and use gssapi:kerberos5 authentication:

```
{
    CAPISession mySession = CSDK_SESSION_INITIALIZER;
    status = CSDK_CreateSession(&mySession);
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_Connect(mySession, CAPI_FLAG_
NONE, "calserver.acme.com:12345");
    }
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_ConfigureACE(mySession,
                                     CAPI_FLAG_NONE,
                                     "gssapi:kerberos5",
```

```
                                        NULL,
                                        NULL);
    }
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_Authenticate(mySession,
                                        CAPI_FLAG_NONE,
                                        "",     // don't pass in user string
                                        "");    // don't pass in password
    }
}
```

Cleanup

A call to CSDK_Deauthenticate must be made between calls to CSDK_Authenticate

Equivalent Java Method

oracle.calendar.sdk.Session.authenticate()

CSDK_ConfigureACE

Configures the given session to use specific ACE (Authentication, Compression, and Encryption) mechanisms between the SDK client and the Calendar server.

```
CAPIStatus CSDK_ConfigureACE (
  CAPISession in_session,
  CAPIFlag in_flags,
  const char * in_authenticationMechanism,
  const char * in_compressionMechanism,
  const char * in_encryptionMechanism
)
```

If this function is not called, the default mechanisms as set on the Oracle Calendar server will be used.

The server needs proper configuration for some ACE mechanisms

NULL values can be specified to select the server's default mechanism for any of the three types of mechanisms.

Parameters

in_session
Login session handle

in_flags
SDK_FLAG_NONE

in_authenticationMechanism
Name of authentication mechanism (for example, "cs-standard", "gssapi:kerberos5", NULL)

in_compressionMechanism
Name of compression mechanism (for example, "cs-simple", "NONE", NULL)

in_encryptionMechanism
Name of encryption mechanism (for example, "cs-acipher1", "NONE", NULL)

Returns

CAPIStatus

Equivalent Java Method

oracle.calendar.sdk.Session.configureACE()

CSDK_Connect

Establishes a connection with a calendar service.

```
CAPIStatus CSDK_Connect (
  CAPISession in_session,
  CAPIFlag in_flags,
  const char * in_host
)
```

Parameters

in_session
Login session handle

in_flags
Bit flags (pass CSDK_FLAG_NONE)

in_host
Calendar server host (with optional port number, for example "calserver.acme.com" or "calserver.acme.com:12345"). The host[:port] may optionally be followed by ?/CD=<calendar domain>/

Returns

CAPIStatus

Sample

Connect to the Oracle Calendar server calserver.acme.com. This connection can be used to authenticate as any user known to the masternode:

```
{
    CAPIStatus  myStatus  = CAPI_STAT_OK;
    CAPISession mySession = CSDK_SESSION_INITIALIZER;
    myStatus = CSDK_CreateSession(CSDK_FLAG_NONE, &mySession);
    if (myStatus == CAPI_STAT_OK)
    {
        myStatus = CSDK_connect(mySession, CSDK_FLAG_NONE, "calserver.acme.com");
    }
}
```

Cleanup

The server connection should be released by calling CSDK_Disconnect

Equivalent Java Method

oracle.calendar.sdk.Session.connect()

CSDK_ConnectAsSysop

Logs on as SYSOP; once logged on, SYSOP can assume the identity of any user on the same node by calling CSDK_SetIdentity().

```
CAPIStatus CSDK_ConnectAsSysop (
  CAPISession in_session,
  CAPIFlag in_flags,
  const char * in_host,
  const char * in_nodeId,
  const char * in_password
)
```

A node must always be specified since masternode and Calendar domain functionality is not available during logon as SYSOP.

If ACE mechanisms have been configured on the session, these will be ignored. The admin default ACE settings from the Oracle Calendar server will be used for all SYSOP connections.

SYSOP authentication is only available with version 5.3 and newer servers. An error will be returned if the specified host does not support this feature. A Calendar server may be configured to refuse SYSOP logon via the SDK in which case a security error will be returned.

The operations available to SYSOPs are limited to the following:

- Disconnecting by calling CSDK_Disconnect()
- Switching identity to a user by calling CSDK_SetIdentity()

Once the identity has been set to a user, all operations will be performed as if that user had logged in.

See CSDK_Connect() for the format of the in_host parameter.

Parameters

in_session
Login session handle

in_flags
Bit flags modifying behavior. This must be CSDK_FLAG_NONE currently.

in_host
Calendar server host name (with optional port number)

in_nodeId
Node ID to connect to as SYSOP. Node aliases are not currently supported.

in_password
SYSOP's password

Returns

CAPIStatus

See

CSDK_SetIdentity

Equivalent Java Method

oracle.calendar.sdk.Session.connectAsSysop()

CSDK_CreateCallbackStream

Creates a callback stream that can be used to either supply data to, or receive data from, the SDK.

```
CAPIStatus CSDK_CreateCallbackStream (
  CAPISession in_session,
  CAPIStream * out_stream,
  CAPICallback in_sendCallback,
  void * in_sendUserData,
  CAPICallback in_recvCallback,
  void * in_recvUserData,
  CAPIFlag in_flags
)
```

C function pointers are supplied for each action (send, receive) that the SDK will call to either read or send data.

During a CSDK_Store...() call, the SDK will call the function in_sendCallback, passing in the value in_sendUserData (which is typically used to store some context to be used by the callback function).

During a CSDK_Fetch...() call, the SDK will call the function in_recvCallback, passing in the value in_recvUserData (which is typically used to store some context to be used by the callback function).

Both types of callback functions use the same function signature:

```
 typedef int (*CAPICallback)(
      void *      in_
userData,    // user-defined data (the value supplied in CAPI_
CreateCallbackStream)
      char *      io_data,        // buffer to read or write
      size_t      in_dataSize,    // the number of characters to read or write
      size_t *    out_dataSize);   // the number of characters read or written
```

The return values from the callbacks must be one of the following:

- Send callback:
  - CAPI_CALLBACK_CONTINUE: There is more data to be read from the stream
  - CAPI_CALLBACK_DONE: There is no more data to be read from the stream
  - A positive integer: An error has occurred. This positive integer will be returned as part of the CAPIStatus returned in bit 5 with the value CAPI_STAT_API_CALLBACK_ERROR
- Receive callback:
  - CAPI_CALLBACK_CONTINUE: No error
  - A positive integer: An error has occurred (e.g. the stream cannot receive any more data). This positive integer will be returned as part of the CAPIStatus returned in bit 5 with the value CAPI_STAT_API_CALLBACK_ERROR

When the SDK has finished writing data to the receive callback, the callback will be called with in_dataSize == 0.

In many applications, it is easier to use either a memory stream or file stream than a callback stream.

Parameters

in_session
Login session handle

out_stream
On output, will point to new stream.

in_sendCallback
Send data callback

in_sendUserData
A value that will be passed to in_sendCallback

in_recvCallback
Receive data callback

in_recvUserData
A value that will be passed to in_recvCallback

in_flags
Bit flags (must be CSDK_FLAG_NONE at this time)

Returns

CAPIStatus

Cleanup

The stream returned by this function must be destroyed by calling CSDK_DestroyStreams()

Return values

CAPI_STAT_API_NULL
Both supplied callbacks were NULL

See

CSDK_CreateMemoryStream()

See

CSDK_CreateFileStreamFromFilenames()

Equivalent Java Method

None. The Java APIs only use String and StringBuffer objects to send and receive data.

CSDK_CreateFileStreamFromFilenames

Creates a file stream to allow the SDK to read from or write to files.

```
CAPIStatus CSDK_CreateFileStreamFromFilenames (
  CAPISession in_session,
  CAPIStream * out_stream,
  const char * in_readFileName,
  const char * in_readMode,
  const char * in_writeFileName,
  const char * in_writeMode,
  CAPIFlag in_flags
)
```

Parameters

in_session
Login session handle

out_stream
On output, will point to new stream

in_readFileName
Name of file from which to read

in_readMode
Mode to pass while opening in_readFileName

in_writeFileName
Name of file to which to write

in_writeMode
Mode to pass while opening in_writeFileName

in_flags
Bit flags (must be CSDK_FLAG_NONE at this time)

Returns

CAPIStatus

Cleanup

The stream returned by this function must be destroyed by calling CSDK_DestroyStreams.

Return values

CAPI_STAT_SERVICE_FILE_MODE
An invalid mode was passed in

CAPI_STAT_SERVICE_FILE_OPEN
Failed to open a file

Sample

Store events from the file "events.ics":

```
  CAPIStream myInputStream = NULL;
  CAPIStatus status = CAPI_CreateFileStreamFromFilenames(mySession,
                                                         &myInputStream,
                                                         "events.ics",
                                                         "rb",
                                                         NULL,    // no output fil
e
                                                         NULL,    // no output fil
e mode
                                                         CSDK_FLAG_NONE);
  if (status == CAPI_STAT_OK)
  {
      status = CAPI_StoreEvent(mySession,
                               myHandles,
                               numHandles,
                               handleStatus,
                               CAPI_STORE_REPLACE,
                               myInputStream);
      CAPI_DestroyStreams(mySession,
                          &myInputStream,
                          1,
                          CSDK_FLAG_NONE);
  }
```

Sample

Fetch events and write them to the file "myAgenda.ics":

```
  CAPIStream myOutputStream = NULL;
  CAPIStatus status = CSDK_CreateFileStreamFromFilenames(mySession,
                                                         &myOutputStream,
                                                         NULL,  // no input file
                                                         NULL,   // no input file
mode
                                                         "myAgenda.ics",
                                                         "wb",
                                                         CSDK_FLAG_NONE);
  if (status == CAPI_STAT_OK)
  {
      status = CAPI_FetchEventsByRange(mySession,
                                       myHandles,
                                       numHandles,
                                       handleStatus,
                                       CSDK_FLAG_NONE,
                                       "20020722T000000",
                                       "20020722T235900",
                                       NULL,
                                       0,
                                       myOutputStream);
      CAPI_DestroyStreams(mySession,
                          &myOutputStream,
                          1,
                          CSDK_FLAG_NONE);
  }
```

Equivalent Java Method

None. The Java APIs only use String and StringBuffer objects to send and receive data.

CSDK_CreateMemoryStream

Creates a memory stream, which uses data buffers to pass data between your application and the SDK.

```
CAPIStatus CSDK_CreateMemoryStream (
  CAPISession in_session,
  CAPIStream * out_stream,
  const char * in_readBuffer,
  const char ** out_writeBufferPtr,
  CAPIFlag in_flags
)
```

This is often the simplest type of stream to use.

Read buffers are read by the SDK during CSDK_Store...() calls and write buffers are written to by the SDK during CSDK_Fetch...() calls. The read buffers are managed by your application, whereas the SDK will allocate and free memory for the write buffers. The write buffer is freed by the SDK when the memory stream is destroyed.

Parameters

in_session
Login session handle

out_stream
On output, will point to new stream.

in_readBuffer
Buffer for the SDK to read from

out_writeBufferPtr
This address will point to the buffer CAPI is writing into.

in_flags
Bit flags (must be CSDK_FLAG_NONE at this time)

Returns

CAPIStatus

Cleanup

The stream returned by this function must be destroyed by calling CSDK_ DestroyStreams.

Return values

CAPI_STAT_API_NULL
: both supplied buffers were NULL

Sample

Store events from the buffer "events":

```
  const char events[] = "MIME-Version: 1.0\\r\\n"
                        "Content-Type: text/calendar\\r\\n"
```

```
                        "Content-Transfer-Encoding: quoted-printable\\r\\n\\r\\n"
                        "BEGIN:VCALENDAR\\r\\n"
                        "VERSION:2.0\\r\\n"
                        ...etc
                        "END:VCALENDAR\\r\\n";
CAPIStream myInputStream = NULL;
CAPIStatus status = CAPI_CreateMemoryStream(mySession,
                                            &myInputStream,
                                            events,
                                            NULL, // no write buffer
                                            CSDK_FLAG_NONE);
if (status == CAPI_STAT_OK)
{
    status = CAPI_StoreEvent(mySession,
                             myHandles,
                             numHandles,
                             handleStatus,
                             CAPI_STORE_REPLACE,
                             myInputStream);
    CAPI_DestroyStreams(mySession,
                        &myInputStream,
                        1,
                        CSDK_FLAG_NONE);
}
```

Sample

Fetch events and write them to a buffer:

```
const char * todaysEvents = NULL;
CAPIStream myOutputStream = NULL;
CAPIStatus status = CAPI_CreateMemoryStream(mySession,
                                            &myOutputStream,
                                            NULL, // no read buffer
                                            &todaysEvents,
                                            CSDK_FLAG_NONE);
if (status == CAPI_STAT_OK)
{
    status = CAPI_FetchEventsByRange(mySession,
                                     myHandles,
                                     numHandles,
                                     handleStatus,
                                     CSDK_FLAG_NONE,
                                     "20020722T000000",
                                     "20020722T235900",
                                     NULL,
                                     0,
                                     myOutputStream);
    if (status == CAPI_STAT_OK)
    {
        printf("Today's events:\\n%s", todaysEvents);
    }
    CAPI_DestroyStreams(mySession,
                        &myOutputStream,
                        1,
                        CSDK_FLAG_NONE);
}
```

Equivalent Java Method

None. The Java APIs only use String and StringBuffer objects to send and receive data.

CSDK_CreateQuery

Creates a query object to be used with CSDK_FetchContactsByQuery.

```
CAPIStatus CSDK_CreateQuery (
  CSDKCondition * in_condition,
  CSDKQuery * out_query
)
```

An initial condition is specified (for example, "CATEGORIES" equals "shareholders") and more conditions may be added using CSDK_AddConditionToQuery.

Parameters

in_condition
Initial condition for query

out_query
On output, will contain new query object

Returns

CAPIStatus

Cleanup

The query object MUST be destroyed by calling CSDK_DestroyQuery

Sample

Create a query that specifies CATEGORIES properties whose values contain "shareholders". Add a condition that specifies ROLE properties whose values contain "chairman". Retrieve contacts that satisfy these two conditions:

```
//
CSDKCondition cond;
//
cond.prop  = "CATEGORIES";
cond.op    = CSDK_OP_CONTAINS;
cond.value = "shareholders";
//
CSDKQuery myQuery = CSDK_QUERY_INITIALIZER;
stat = CSDK_CreateQuery(&cond,
                        &myQuery);
//
cond.prop  = "ROLE";
cond.op    = CSDK_OP_CONTAINS;
cond.value = "chairman";
//
stat = CSDK_AddConditionToQuery(myQuery,
                                &cond,
                                CSDK_LOP_OR);
//
stat = CSDK_FetchContactsByQuery(mySession,
                                 CSDK_FLAG_FETCH_MATCH_CASE,
                                 myQuery,
                                 NULL,    // get all properties
                                 myStream,
```

```
                                    &requestResult);
//
CSDK_DestroyQuery(&myQuery);
```

Equivalent Java Method

oracle.calendar.sdk.Query constructor

CSDK_CreateSession

Creates a new session.

```
CAPIStatus CSDK_CreateSession (
  CAPIFlag in_flags,
  CAPISession * out_session
)
```

Parameters

in_flags
Bit flags (currently, set this to CSDK_FLAG_NONE)

out_session
Pointer to new session

Cleanup

The session must be destroyed using CSDK_DestroySession()

Equivalent Java Method

oracle.calendar.sdk.Session constructor

Returns

CAPIStatus

CSDK_Deauthenticate

Deauthenticates the current user.

```
CAPIStatus CSDK_Deauthenticate (
  CAPISession in_session,
  CAPIFlag in_flags
)
```

An unauthenticated server connection is kept open and can be used to re-authenticate again. The server connection is kept open until either a call to CSDK_Disconnect() or the session is destroyed.

Parameters

in_session
Login session handle

in_flags
Bit flags modifying behavior. This must be CSDK_FLAG_NONE currently.

Returns

CAPIStatus

Equivalent Java Method

oracle.calendar.sdk.Session.deauthenticate()

CSDK_DeleteContacts

Deletes vCards specified by a set of UIDs.

```
CAPIStatus CSDK_DeleteContacts (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIUIDSet in_UIDs,
  CSDKRequestResult * out_requestResult
)
```

This function is blocked for SYSOP that has not assumed the identity of a user.

Parameters

in_session
Login session handle

in_flags
Bit flags:

- CSDK_FLAG_NONE
- CSDK_FLAG_CONTINUE_ON_ERROR: if the fetch fails on one uid we still fetch the other valid uid's and return the stream with these *valid* events. out_ requestResult will contain information on every uid's we tried to fetch.

in_UIDs
NULL terminated array of strings containing UIDs of vCards to delete

out_requestResult
Pointer to a RequestResult that will get filled

Returns

CAPIStatus

Return values

CAPI_STAT_API_HANDLE_NULL
The session was NULL

CAPI_STAT_API_NULL
in_UIDs was NULL

Equivalent Java Method

oracle.calendar.sdk.Session.deleteContacts()

CSDK_DeleteEvents

Deletes specified events; must be acting as the event owner for this to succeed.

```
CAPIStatus CSDK_DeleteEvents (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIUIDSet in_UIDs,
  const char * in_RECURRENCEID,
  int in_modifier,
  CSDKRequestResult * out_requestResult
)
```

This does not "uninvite" attendees; it deletes the event. Individual (or a range) of instances can be deleted using {in_RECURRENCEID, in_modifier}, but only a single UID can be used in this case.

Parameters

in_session
Login session handle

in_flags
Bit Flags:

- CSDK_FLAG_NONE: Default behavior
- CSDK_FLAG_CONTINUE_ON_ERROR: if the delete fails on one UID we still delete the other valid UIDs. out_requestResult will contain information on every UID we tried to fetch.

in_UIDs
An array of strings containing the UID(s) of the event(s) to delete. The array MUST be terminated with either a NULL (zero) pointer, or an empty (length zero) string.

in_RECURRENCEID
To delete ALL occurrences of an event, pass in NULL or an empty string. To delete individual (or a range of) occurrences, specify an iCalendar recurrence-id in either DATE or DATE-TIME format that identifies one occurrence of the event.

in_modifier
When a recurrence-id is specified using in_RECURRENCEID, this modifier determines whether the specified occurrences, or a range of occurrences will be deleted. The following are the possible values:

- CAPI_THISINSTANCE
- CAPI_THISANDPRIOR
- CAPI_THISANDFUTURE

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Equivalent Java Method

oracle.calendar.sdk.Session.deleteEvents()

CSDK_DeleteTasks

Deletes tasks from the current user's agenda.

```
CAPIStatus CSDK_DeleteTasks (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIUIDSet in_UIDs,
  CSDKRequestResult * out_requestResult
)
```

This function is blocked for SYSOP that has not assumed the identity of a user.

Parameters

in_session
Login handle session

in_flags
Bit Flags:

- CSDK_FLAG_NONE: Default behavior
- CSDK_FLAG_CONTINUE_ON_ERROR: if the delete fails on one uid we still delete the other valid UIDs. out_requestResult will contain information on every UID we tried to fetch.

in_UIDs
NULL terminated array of task UIDs

out_requestResult
Returned request result object

Returns

CAPIStatus

Return values

CAPI_STAT_OK

CAPI_STAT_API_SESSION_NULL
in_session is NULL

CAPI_STAT_API_NULL
in_UIDSet is NULL

Equivalent Java Method

oracle.calendar.sdk.Session.deleteTasks()

CSDK_DestroyHandle

Destroys one handle.

```
CAPIStatus CSDK_DestroyHandle (
  CAPISession in_session,
  CAPIHandle * in_handle
)
```

Parameters

in_session
Login session handle

in_handle
Handle (returned by CSDK_GetHandle) to destroy

Returns

CAPIStatus

Sample

Destroy one handle:

```
{
    CAPIHandle h1 = CSDK_HANDLE_INITIALIZER;
    CSDK_GetHandle(mySession, "arthur", CSDK_FLAG_NONE, &h1);
    ...
    OCAP_DestroyHandle(mySession, &h1);
}
```

Equivalent Java Method

None. oracle.calendar.sdk.Handle finalizer will destroy handles.

CSDK_DestroyMultipleHandles

Destroys multiple handles returned by calls to CSDK_GetHandle().

```
CAPIStatus CSDK_DestroyMultipleHandles (
  CAPISession in_session,
  CAPIHandle * in_handles,
  int in_numHandles,
  CAPIFlag in_flags
)
```

Parameters

in_session
Login session handle

in_handles
Array of handles (returned by CSDK_GetHandle) to destroy

in_numHandles
The size of the handle array

in_flags
Bit flags (none at this time, set to CSDK_FLAG_NONE)

Returns

CAPIStatus

Sample

Destroy two handles:

```
{
    CAPIHandle h1 = CSDK_HANDLE_INITIALIZER;
    CAPIHandle h2 = CSDK_HANDLE_INITIALIZER;
    CSDK_GetHandle(mySession, "arthur", CSDK_FLAG_NONE, &h1);
    CSDK_GetHandle(mySession, "tim...", CSDK_FLAG_NONE, &h2);
    ...
    CAPIHandle handles[] = {h1, h2};
    CSDK_DestroyMultipleHandles(mySession, handles, 2, CSDK_FLAG_NONE);
}
```

Equivalent Java Method

None. oracle.calendar.sdk.Handle finalizer will destroy handles.

CSDK_DestroyMultipleStreams

Destroys streams created by the various CSDK_Create...Stream functions.

```
CAPIStatus CSDK_DestroyMultipleStreams (
  CAPISession in_session,
  CAPIStream * in_streams,
  int in_numStreams,
  CAPIFlag in_flags
)
```

Parameters

in_session
The session with which streams are associated

in_streams
Array of streams to destroy

in_numStreams
The number of streams in in_streams to destroy

in_flags
Bit flags modifying behavior. Must be CSDK_FLAG_NONE at this time.

Returns

CAPIStatus

Equivalent Java Method

None. The Java APIs only use String and StringBuffer objects to send and receive data.

CSDK_DestroyQuery

Destroys a query object created by CSDK_CreateQuery.

```
CAPIStatus CSDK_DestroyQuery (
  CSDKQuery * io_query
)
```

Parameters

io_query
A pointer to a query object to destroy. Will point to CSDK_QUERY_INITIALIZER on exit.

Returns

CAPIStatus

Equivalent Java Method

None. oracle.calendar.sdk.Query finalizer destroys object.

CSDK_DestroyResult

Disposes of all the results in in_requestResult.

```
CAPIStatus CSDK_DestroyResult (
  CSDKRequestResult * io_requestResult
)
```

Parameters

io_requestResult
The RequestResult to destroy

Returns

CAPIStatus

Return values

CAPI_STAT_API_NULL
The result io_requestResult has a NULL value

Equivalent Java Method

oracle.calendar.sdk.Result finalizer

CSDK_DestroySession

Destroys a session.

```
CAPIStatus CSDK_DestroySession (
  CAPISession * io_session
)
```

Parameters

io_session
Pointer to session to destroy. Will point to CAPI_SESSION_INITIALIZER on output.

Returns

CAPIStatus

Equivalent Java Method

oracle.calendar.sdk.Session finalizer

CSDK_DestroyStream

Destroys a stream created by any of the various CSDK_Create...Stream functions.

```
CAPIStatus CSDK_DestroyStream (
  CAPISession in_session,
  CAPIStream * io_stream
)
```

Parameters

in_session
The session with which the stream is associated

io_stream
Stream to destroy

Returns

CAPIStatus

Equivalent Java Method

None. The Java APIs only use String and StringBuffer objects to send and receive data.

CSDK_Disconnect

Disconnects from the Oracle Calendar server.

```
CAPIStatus CSDK_Disconnect (
  CAPISession in_session,
  CAPIFlag in_flags
)
```

Parameters

in_session
Login session handle

in_flags
Bit flags modifying behavior. This must be CSDK_FLAG_NONE currently.

Returns

CAPIStatus

Equivalent Java Method

oracle.calendar.sdk.Session.disconnect()

CSDK_FetchContactsByQuery

Fetches contacts which satisfy the conditions specified in the query.

```
CAPIStatus CSDK_FetchContactsByQuery (
  CAPISession in_session,
  CAPIFlag in_flags,
  CSDKQuery in_query,
  const char ** in_requestProperties,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

The returned vCards are returned via in_sendStream, and by default will be in MIME format. Each vCard fetched vCard will be in a separate MIME part. The character set will be UTF-8.

To avoid having the stream MIME-encapsulated, pass in the flag CSDK_FLAG_ STREAM_NOT_MIME.

Currently the supported types for the queries are N, FN, BDAY, TITLE, ROLE, NOTE, CATEGORIES, NICKNAME, X-ORACLE-SPOUSE, X-ORACLE-ANNIVERSARY, X-ORACLE-OFFICE, and X-ORACLE-ASSISTANTNAME.

Parameters

in_session
Login session handle

in_flags
Bit flags

- CSDK_FLAG_NONE
- CSDK_FLAG_STREAM_NOT_MIME: Do NOT wrap the output in a MIME container

in_query
A query object containing the search criteria

in_requestProperties
To fetch only specific vCard properties of the contacts, pass in an array of property names. The array MUST be terminated with either a NULL (zero) pointer, or an empty (length zero) string. An empty array (i.e. NULL or an empty string), will cause all available properties to be returned.

in_stream
Stream for the SDK to write into

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction.

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Equivalent Java Method

oracle.calendar.sdk.Session.fetchContactsByQuery()

See

CSDK_CreateQuery

CSDK_FetchContactsByUID

Fetches vCards from an authenticated connection.

```
CAPIStatus CSDK_FetchContactsByUID (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIUIDSet in_UIDs,
  const char ** in_requestProperties,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

The fetched vCards are by default in MIME format. Each vCard fetched vCard will be in a separate MIME part. The character set will be UTF-8.

To avoid having the stream MIME-encapsulated, pass in the flag CSDK_FLAG_STREAM_NOT_MIME.

This function is blocked for SYSOP that has not assumed the identity of a user.

Parameters

in_session
Login session handle

in_flags
Bit flags:

- CSDK_FLAG_NONE
- CSDK_FLAG_STREAM_NOT_MIME: Do not wrap the output in a MIME container
- CSDK_FLAG_CONTINUE_ON_ERROR: if the fetch fails on one uid we still fetch the other valid uid's and return the stream with these *valid* events. out_requestResult will contain information on every uid's we tried to fetch.

in_UIDs
NULL terminated array of strings containing UIDs of vCards to fetch

in_requestProperties
To fetch only specific vCard properties of the contacts, pass in an array of property names. The array MUST be terminated with either a NULL (zero) pointer, or an empty (length zero) string. An empty array (i.e. NULL or an empty string), will cause all available properties to be returned.

in_stream
Stream for the SDK to write into

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction.

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Return values

CAPI_STAT_API_HANDLE_NULL
The session was NULL

CAPI_STAT_API_STREAM_NULL
The stream was NULL

CAPI_STAT_API_NULL
in_UIDSet was NULL

CAPI_STAT_API_BADPARAM
in_UIDCount was 0

Equivalent Java Method

oracle.calendar.sdk.Session.fetchContactsByUID()

CSDK_FetchEventsByAlarmRange

Fetches events which have alarms (reminders) that will trigger within the time range specified; the end of the time range is exclusive.

```
CAPIStatus CSDK_FetchEventsByAlarmRange (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIHandle * in_agendas,
  const char * in_start,
  const char * in_end,
  const char ** in_requestProperties,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

Parameters

in_session
Login session handle

in_flags
Bit flags:

- CSDK_FLAG_NONE: Default behavior
- CSDK_FLAG_FETCH_AGENDA_ATTENDEE_ONLY: Only get/return ATTENDEE for the agenda being viewed
- CSDK_FLAG_FETCH_COMBINED: Return all events in one VCALENDAR rather than one VCALENDAR per agenda. This is faster.
- CSDK_FLAG_FETCH_EXCLUDE_HOLIDAYS: Do not fetch holidays
- CSDK_FLAG_FETCH_EXCLUDE_DAILYNOTES: Do not fetch daily notes
- CSDK_FLAG_FETCH_EXCLUDE_DAYEVENTS: Do not fetch day events
- CSDK_FLAG_FETCH_EXCLUDE_APPOINTMENTS: Do not appointments
- CSDK_FLAG_FETCH_EXCLUDE_ACCEPTED: Do not fetch accepted events
- CSDK_FLAG_FETCH_EXCLUDE_DECLINED: Do not fetch declined events
- CSDK_FLAG_FETCH_EXCLUDE_UNCONFIRMED: Do not fetch unconfirmed events
- CSDK_FLAG_FETCH_LOCALTIMES: Return all dates and times in the user's time zone (the user's time zone preference stored on the Oracle Calendar server)
- CSDK_FLAG_FETCH_DO_NOT_EXPAND_RRULE: Do not expand recurrence rules. This will cause the entire event to be returned instead of only the instances which have alarms scheduled to trigger during the range.
- CSDK_FLAG_STREAM_NOT_MIME: Do not wrap the iCalendar in a MIME container

in_agendas
The agenda(s) in which to search for events. This parameter is an array of CAPIHandles, and MUST be terminated with a zero value (CSDK_HANDLE_

INITIALIZER). A NULL value, or an empty array will search on the current user's agenda.

in_start
Beginning of date/time range. May be of any of the following forms:

- DATE: For example, 20020928
- DATE-TIME: Must be in either floating (for example, 20020929T120000) or UTC time (e.g. 20020929T170000Z). Floating time uses the user's time zone (the user's time zone preference stored on the Calendar server).
- DURATION: A relative date or date-time expressed using the duration notation (for example, +PT5DT3H represents 5 days 3 hours in the future, -PT1W 1 week in the past). in_start is relative to the current time at the moment of the fetch.

in_end
End of date/time range. May be in any of the formats shown for in_start.

- NOTE for DURATION: duration specified as the in_end is relative to the in_start. ex: in_end = +P2D, means in_end = in_start + 2 Days

in_requestProperties
To fetch only specific iCalendar properties of the events, pass in an array of property names. The array MUST be terminated with either a NULL (zero) pointer, or an empty (length zero) string. An empty array (such as NULL or an empty string) will cause all available properties to be returned.

in_stream
Stream for CAPI to write into

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction.

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Equivalent Java Method

oracle.calendar.sdk.Session.fetchEventsByAlarmRange()

CSDK_FetchEventsByRange

Fetches events which occur within the time range specified.

```
CAPIStatus CSDK_FetchEventsByRange (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIHandle * in_agendas,
  const char * in_start,
  const char * in_end,
  const char ** in_requestProperties,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

The end of the time range is exclusive.

Parameters

in_session
Login session handle

in_flags
Bit flags:

- CSDK_FLAG_NONE: Default behavior
- CSDK_FLAG_FETCH_AGENDA_ATTENDEE_ONLY: Only get/return ATTENDEE for the agenda being viewed
- CSDK_FLAG_FETCH_COMBINED: Return all events in one VCALENDAR rather than one VCALENDAR per agenda. This is faster.
- CSDK_FLAG_FETCH_EXCLUDE_HOLIDAYS: Do not fetch holidays
- CSDK_FLAG_FETCH_EXCLUDE_DAILYNOTES: Do not fetch daily notes
- CSDK_FLAG_FETCH_EXCLUDE_DAYEVENTS: Do not fetch day events
- CSDK_FLAG_FETCH_EXCLUDE_APPOINTMENTS: Do not appointments
- CSDK_FLAG_FETCH_EXCLUDE_ACCEPTED: Do not fetch accepted events
- CSDK_FLAG_FETCH_EXCLUDE_DECLINED: Do not fetch declined events
- CSDK_FLAG_FETCH_EXCLUDE_UNCONFIRMED: Do not fetch unconfirmed events
- CSDK_FLAG_FETCH_LOCALTIMES: Return all dates & times in the user's time zone (the user's time zone preference stored on the Oracle Calendar server)
- CSDK_FLAG_FETCH_DO_NOT_EXPAND_RRULE: Do not expand recurrence rules. This will cause the entire event to be returned instead of only the instances which fall during the range.
- CSDK_FLAG_STREAM_NOT_MIME: Do not wrap the iCalendar in a MIME container

in_agendas
The agenda(s) in which to search for events. This parameter is an array of CAPIHandles, and MUST be terminated with a zero value (CSDK_HANDLE_

INITIALIZER). A NULL value, or an empty array will search on the current user's agenda.

in_start
Beginning of date/time range. May be of any of the following forms:

- DATE: For example, 20020928
- DATE-TIME: Must be in either floating (for example, 20020929T120000) or UTC time (e.g. 20020929T170000Z). Floating time uses the user's time zone (the user's time zone preference stored on the Calendar server).
- DURATION: A relative date or date-time expressed using the duration notation (for example, +PT5DT3H represents 5 days 3 hours in the future, -PT1W 1 week in the past). in_start is relative to the current time at the moment of the fetch.

in_end
End of date/time range. May be in any of the formats shown for in_start.

- NOTE for DURATION: duration specified as the in_end is relative to the in_start. ex: in_end = +P2D, means in_end = in_start + 2 Days

in_requestProperties
To fetch only specific iCalendar properties of the events, pass in an array of property names. The array MUST be terminated with either a NULL (zero) pointer, or an empty (length zero) string. An empty array (such as NULL or an empty string), will cause all available properties to be returned.

in_stream
Stream for CAPI to write into

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction.

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Equivalent Java Method

oracle.calendar.sdk.Session.fetchEventsByRange()

CSDK_FetchEventsByUID

Fetches events by their UIDs.

```
CAPIStatus CSDK_FetchEventsByUID (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIHandle in_agenda,
  CAPIUIDSet in_UIDs,
  const char * in_RECURRENCEID,
  int in_modifier,
  const char ** in_requestProperties,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

Specific instances of one event may be fetched using the in_RECURRENCEID and in_modifier parameters.

Specific properties can be requested using the in_requestProperties parameter. This parameter is a NULL(zero)-terminated or "empty string"-terminated array of C strings containing the property names to be returned.

For maximum performance, limit the properties you request (particularly the ATTENDEE property) to only what you need.

Parameters

in_session
Login session handle

in_flags
Bit flags:

- CSDK_FLAG_NONE: Default behavior
- CSDK_FLAG_FETCH_AGENDA_ATTENDEE_ONLY: Only get/return ATTENDEE for the agenda being viewed
- CSDK_FLAG_FETCH_LOCALTIMES: Return all dates & times in the user's time zone (the user's time zone preference stored on the Oracle Calendar server)
- CSDK_FLAG_FETCH_EXPAND_RRULE: Expand recurrence rules and return a set of VEVENTs one per instance generated by the recurrence rule
- CSDK_FLAG_CONTINUE_ON_ERROR: if the fetch fails on one uid we still fetch the other valid UIDs and return the stream with these *valid* events. out_requestResult will contain information on every UID we tried to fetch.
- CSDK_FLAG_STREAM_NOT_MIME: Do not wrap the iCalendar in a MIME container

in_agenda
The agenda on which to search for event(s) with the given UID(s). A NULL value will search on the current user's agenda.

in_UIDs
An array of strings containing the UID(s) of the events to fetch. The array MUST be terminated with either a NULL (zero) pointer, or an empty (length zero) string.

in_RECURRENCEID
To fetch ALL occurrences of an event, pass in NULL or an empty string. To fetch individual (or a range of) occurrences, specify an iCalendar recurrence-id in either DATE or DATE-TIME format which identifies one occurrence of the event.

in_modifier
When a recurrence-id is specified using in_RECURRENCEID, this modifier determines whether the specified occurrences, or a range of occurrences will be fetched. Values are:

- CAPI_THISINSTANCE
- CAPI_THISANDPRIOR
- CAPI_THISANDFUTURE

in_requestProperties
To fetch only specific iCalendar properties of the events, pass in an array of property names. The array MUST be terminated with either a NULL (zero) pointer, or an empty (length zero) string. An empty array (such as NULL or an empty string), will cause all available properties to be returned.

in_stream
Stream for CAPI to write into

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction.

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Equivalent Java Method

oracle.calendar.sdk.Session.fetchEventsByUID()

CSDK_FetchTasksByAlarmRange

Fetches tasks that have alarms (reminders) that will trigger within the time range specified (the end of the time range is exclusive).

```
CAPIStatus CSDK_FetchTasksByAlarmRange (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIHandle * in_handles,
  const char * in_start,
  const char * in_end,
  const char ** in_requestProperties,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

Parameters

in_session
Login session handle

in_flags
Bit flags:

- CSDK_FLAG_NONE
- CSDK_FLAG_STREAM_NOT_MIME: Do NOT wrap the iCalendar in a MIME container

in_handles
The agenda(s) on which to search for tasks. This parameter is an array of CAPIHandles, and MUST be terminated with a zero value (CSDK_HANDLE_ INITIALIZER). A NULL value, or an empty array will search on the current user's agenda.

in_start
Beginning of date/time range. May be in any of the following forms:

- DATE: For example, 20020928
- DATE-TIME: Must be in either floating (for example, 20020929T120000) or UTC time (for example, 20020929T170000Z)
- DURATION: A relative date or date-time expressed using the duration notation (for example, +PT5DT3H represents 5 days 3 hours in the future, -PT1W 1 week in the past). in_start is relative to the current time at the moment of the fetch.

in_end
End of date/time range. May be in any of the formats shown for in_start.

- NOTE for DURATION: duration specified as the in_end is relative to the in_start. ex: in_end = +P2D, means in_end = in_start + 2 Days

in_requestProperties
To fetch only specific iCalendar properties of the tasks, pass in an array of property names. The array MUST be terminated with either a NULL (zero) pointer, or an empty

(length zero) string. An empty array (such as NULL or an empty string), will cause all available properties to be returned.

in_stream
Stream for CAPI to write into

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction.

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Equivalent Java Method

oracle.calendar.sdk.Session.fetchTasksByAlarmRange()

CSDK_FetchTasksByRange

Fetches tasks which are active within the time range specified (the end of the time range is exclusive).

```
CAPIStatus CSDK_FetchTasksByRange (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIHandle * in_handles,
  const char * in_start,
  const char * in_end,
  const char ** in_requestProperties,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

Parameters

in_session
Login session handle

in_flags
Bit flags:

- CSDK_FLAG_NONE
- CSDK_FLAG_STREAM_NOT_MIME: Do not wrap the iCalendar in a MIME container

in_handles
The agenda(s) on which to search for tasks. This parameter is an array of CAPIHandles, and MUST be terminated with a zero value (CSDK_HANDLE_INITIALIZER). A NULL value, or an empty array will search on the current user's agenda.

in_start
Beginning of date/time range. May be of any of the following forms:

- DATE: For example, 20020928
- DATE-TIME: Must be in either floating (for example, 20020929T120000) or UTC time (for example, 20020929T170000Z)
- DURATION: A relative date or date-time expressed using the duration notation (for example, +PT5DT3H represents 5 days 3 hours in the future, -PT1W 1 week in the past). in_start is relative to the current time at the moment of the fetch.

in_end
End of date/time range. May be in any of the formats shown for in_start.

- NOTE for DURATION: duration specified as the in_end is relative to the in_start. ex: in_end = +P2D, means in_end = in_start + 2 Days

in_requestProperties
To fetch only specific iCalendar properties of the tasks, pass in an array of property names. The array MUST be terminated with either a NULL (zero) pointer, or an empty

(length zero) string. An empty array (i.e. NULL or an empty string), will cause all available properties to be returned.

in_stream
Stream for CAPI to write into

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction.

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Equivalent Java Method

oracle.calendar.sdk.Session.fetchTasksByRange()

CSDK_FetchTasksByUID

Retrieves tasks with given UIDs on the given agenda.

```
CAPIStatus CSDK_FetchTasksByUID (
  CAPISession in_session,
  CAPIHandle in_handle,
  CAPIFlag in_flags,
  CAPIUIDSet in_UIDs,
  const char ** in_requestProperties,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

Parameters

in_session
Login session handle

in_flags
Bit flags:

- CSDK_FLAG_NONE
- CSDK_FLAG_CONTINUE_ON_ERROR: if the fetch fails on one uid we still fetch the other valid UIDs and return the stream with these *valid* events. out_ requestResult will contain information on every UID we tried to fetch.
- CSDK_FLAG_STREAM_NOT_MIME: Do not wrap the iCalendar in a MIME container

in_handle
The agenda on which to search for tasks with the given UIDs. A NULL value will search on the current user's agenda.

in_UIDs
An array of strings containing the UID(s) of the tasks to fetch. The array MUST be terminated with either a NULL (zero) pointer, or an empty (length zero) string.

in_requestProperties
To fetch only specific iCalendar properties of the tasks, pass in an array of property names. The array MUST be terminated with either a NULL (zero) pointer, or an empty (length zero) string. An empty array (i.e. NULL or an empty string), will cause all available properties to be returned.

in_stream
Stream for CAPI to write into

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction.

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Equivalent Java Method

oracle.calendar.sdk.Session.fetchTasksByUID()

CSDK_GetCapabilities

Returns information about this SDK release and/or the Oracle Calendar server.

```
CAPIStatus CSDK_GetCapabilities (
  CAPISession in_session,
  CAPICapabilityID in_capabilityID,
  CAPIFlag in_flags,
  const char ** out_value
)
```

Parameters

in_session
Login session handle. If NULL, then no server capabilities can be requested.

in_capabilityID
ID for a capability (see CAPI_CAPAB_* in ctapi.h)

in_flags
CSDK_FLAG_NONE at this time

out_value
Information is returned in this parameter. The values are returned as read-only strings and are only valid until the next SDK function call that uses the same session.

Changes

CAPI 2.5: type of in_capabilityID was changed from "long" to "CAPICapabilityID"

Equivalent Java Method

oracle.calendar.sdk.Session.getCapabilities

CSDK_GetFirstFailure

Returns the first failure obtained from the function from which in_requestResult was returned.

```
CAPIStatus CSDK_GetFirstFailure (
  CSDKRequestResult in_requestResult,
  CAPIHandle * out_user,
  const char ** out_uid,
  CAPIStatus * out_status
)
```

A **failure** is a result which has a status other than CAPI_STAT_OK.

Note: A request result contains the reference to the "current" failure, so only one thread should extract failures from a given request result at a time.

Parameters

in_requestResult
The RequestResult from which to extract information

out_user
The user whose agenda was being read or written

out_uid
The uid of the calendar object being read or written, if applicable

out_status
The status code for the portion of the operation involving the user and uid specified

Returns

CAPIStatus

Return values

CAPI_STAT_API_NULL
One of the required parameters has a NULL value

CAPI_STAT_DATA_RRESULT_EOR
No failures in the RequestResult

Sample

```
Output first failure returned from call to CSDK_StoreContacts, if any
 const char * vcardUID = 0;
 CAPIStatus   vcardStatus = CAPI_STAT_OK;
 CSDKRequestResult * myRequestResult = 0;
 //
 stat = CSDK_StoreContacts(mySession,
                           myStream,
                           CSDK_FLAG_STORE_IMPORT,
                           &myRequestResult);
 //
 CAPIStatus failStat = CSDK_GetFirstFalure(myRequestResult,
```

```
                                                NULL,
                                                &vcardUID,
                                                &vcardStatus);
    //
    if (failStat == CAPI_STAT_DATA_RRESULT_EOR)
    {
        cout << "Store of VCARD with UID " << vcardUID << " suceeded." << endl;
    }
    else
    {
        const char * statusName = 0;
        CSDK_GetStatusString(vcardStatus, &statusName);
        cout << "Store of VCARD with UID " << vcardUID << " failed with CAPIStatus "
<< statusName << "." << endl;
    }
    //
    CSDK_DestroyResult(&myRequestResult);
```

Equivalent Java Method

oracle.calendar.sdk.Result.getFirstFailure()

CSDK_GetFirstParseError

Returns the first parsing error obtained from a request result.

```
CAPIStatus CSDK_GetFirstParseError (
  CSDKRequestResult in_requestResult,
  CAPIStatus * out_status,
  const char ** out_errorBuffer,
  const char ** out_errorLocation,
  const char ** out_message
)
```

A parse error can be generated by any of the CSDK_Store* functions as they attempt to interpret incoming iCalendar or vCard.

Note: A request result contains the reference to the "current" parse error, so only one thread should extract parse errors from a given request result at a time.

A pointer to a copy of the data stream is returned through out_errorBuffer, and a pointer to the parse error location in the buffer is returned via out_errorLocation. Both pointers are valid only until the request result is destroyed.

Parameters

in_requestResult
The RequestResult to extract information from

out_status
The result's status

out_errorBuffer
The beginning of the buffer with the error

out_errorLocation
The location in *out_errorBuffer where the error occurred

out_message
May contain additional information (NULL may be returned if no message is available)

Returns

CAPIStatus

Return values

CAPI_STAT_API_NULL
One of the required parameters has a NULL value

CAPI_STAT_DATA_RRESULT_EOR
No parse errors in in_requestResult

Sample

Get the first parsing error from a call to CSDK_StoreContacts:

```
CAPIStatus          stat = CAPI_STAT_OK;
CSDKRequestResult * myRequestResult = 0;
```

```
  //
  stat = CSDK_StoreContacts(mySession,
                            myStream,
                            CSDK_FLAG_STORE_IMPORT,
                            &myRequestResult);
  //
  const char * buffer = 0;
  const char * errorLocation = 0;
  const char * message = 0;
  //
  CAPIStatus parseStat = CSDK_GetFirstParseError(myRequestResult,
                                                 NULL,
                                                 &buffer,
                                                 &errorLocation,
                                                 &message);
  //
  if (parseStat != CAPI_STAT_DATA_RRESULT_EOR)
  {
      cout << "Error (" << message << ") parsing vCard.  Buffer:'" << vcardUID <<
"'  Error starting at:'" << errorLocation << "'" << endl;
  }
  //
  CSDK_DestroyResult(&myRequestResult);
```

Equivalent Java Method

oracle.calendar.sdk.Result.getFirstParseError()

CSDK_GetFirstResult

Returns the first result obtained from the function from which in_requestResult was returned.

```
CAPIStatus CSDK_GetFirstResult (
  CSDKRequestResult in_requestResult,
  CAPIHandle * out_user,
  const char ** out_uid,
  CAPIStatus * out_status
)
```

A **result** is either a failure or a success. A **failure** is a result which has a status other than CAPI_STAT_OK.

Note: A request result contains the reference to the "current" result, so only one thread should extract result from a given request result at a time.

Parameters

in_requestResult
The RequestResult from which to extract information

out_user
The user whose agenda was being read or written

out_uid
The uid of the calendar object being read or written, if applicable

out_status
The status code for the portion of the operation involving the user and uid specified

Returns

CAPIStatus

Return values

CAPI_STAT_API_NULL
One of the required parameters has a NULL value

CAPI_STAT_DATA_RRESULT_EOR
No results in the RequestResult

Sample

Output first result from call to CSDK_StoreContacts:

```
const char * vcardUID = 0;
CAPIStatus   vcardStatus = CAPI_STAT_OK;
CSDKRequestResult * myRequestResult = 0;
//
stat = CSDK_StoreContacts(mySession,
                          myStream,
                          CSDK_FLAG_STORE_IMPORT,
                          &myRequestResult);
//
```

```
CSDK_GetFirstResult(myRequestResult,
                    NULL,
                    &vcardUID,
                    &vcardStatus);
//
if (vcardStatus == CAPI_STAT_OK)
{
    cout << "Store of VCARD with UID " << vcardUID << " suceeded." << endl;
}
else
{
    const char * statusName = 0;
    CSDK_GetStatusString(vcardStatus, &statusName);
    cout << "Store of VCARD with UID " << vcardUID << " failed with CAPIStatus "
<< statusName << "." << endl;
}
//
CSDK_DestroyResult(&myRequestResult);
```

Equivalent Java Method

oracle.calendar.sdk.Result.getFirstResult()

CSDK_GetHandle

Returns a handle to a particular user's calendar store.

```
CAPIStatus CSDK_GetHandle (
  CAPISession in_session,
  const char * in_user,
  CAPIFlag in_flags,
  CAPIHandle * out_handle
)
```

With this handle, subsequent calls can access items in this agenda. If an error is returned no CAPIHandle will be allocated and no cleanup is required.

The in_user string follows the same format as that of the string used by CSDK_ Authenticate.

A handle to the current user is returned if in_user is NULL.

This function is blocked for sysop that has not assumed the identity of a user.

Parameters

in_session
Login session handle

in_user
User as defined for CSDK_Authenticate. May be NULL in which case a handle to the current user is returned.

in_flags
Bit flags (none at this time, set to CSDK_FLAG_NONE)

out_handle
Handle for in_user. Must point to NULL on entry.

Returns

CAPIStatus

Return values

CAPI_STAT_OK

CAPI_STAT_DATA_USERID

CAPI_STAT_SERVICE_MEM

CAPI_STAT_SERVICE_FILE

CAPI_STAT_SERVICE_NET

CAPI_STAT_API_FLAGS

CAPI_STAT_API_NULL

CAPI_STAT_API_HANDLE

CAPI_STAT_API_SESSION

CAPI_STAT_LIBRARY

Cleanup

This function allocates a handle which must be cleaned up with a call to CSDK_DestroyHandle. If an error is returned no handle is allocated and no clean up is required.

Sample

Get a handle for a user whose userid is "roger":

```
{
    CAPIHandle shrubber = CSDK_HANDLE_INITIALIZER;
    stat = CSDK_GetHandle(mySession, "roger", CSDK_FLAG_NONE, &shrubber);
}
```

Sample

Get a handle for a user named "Arnold Layne" (**S** urname Layne, **G** iven name Arnold):

```
{
    CAPIHandle arnold = CSDK_HANDLE_INITIALIZER;
    stat = CSDK_GetHandle(mySession, "?/S=Layne/G=Arnold/, CSDK_FLAG_
NONE, &arnold);
}
```

Sample

Get a handle for a resource named "keg" on node "1234":

```
{
    CAPIHandle keg = CSDK_HANDLE_INITIALIZER;
    stat = CSDK_GetHandle(mySession, "?/RS=keg/ND=1234/", CSDK_FLAG_NONE, &keg);
}
```

Sample

Get a handle for the current user:

```
{
    CAPIHandle currUser = CSDK_HANDLE_INITIALIZER;
    stat = CSDK_GetHandle(mySession, NULL, CSDK_FLAG_NONE, &currUser);
}
```

Changes

CAPI 2.5: Resource names must be an exact match. (There used to be an implicit wildcard at the end of the string.)

Equivalent Java Method

oracle.calendar.sdk.Session.getHandle()

CSDK_GetHandleInfo

Returns information about the agenda of the supplied handle.

```
CAPIStatus CSDK_GetHandleInfo (
  CAPISession in_session,
  CAPIHandle in_handle,
  CAPIFlag in_flags,
  const char ** out_info
)
```

Three pieces of information can be returned, chosen by the value of in_flags. The information is returned as a pointer to a static read-only string.

The following are the types of information that can be returned:

- CAPI_HANDLE_TYPE indicates the type of the handle. This can be "user" or "resource" and indicates what type of agenda this is.
- CAPI_HANDLE_NAME returns the name of the agenda owner, or resource, in the form of a sequence of field-value pairs, separated by "/". This string, when prepended with a '?' is of an appropriate format to be passed to CSDK_GetHandle. A description of this format is given in "User identification" section of this manual.
- CAPI_HANDLE_MAILTO returns the email address of who the agenda belongs to. Since not all users (and no resources) will have e-mail addresses set on the Oracle Calendar server, an error (CAPI_STAT_DATA_EMAIL_NOTSET) will be returned when no e-mail address is set.

Parameters

in_session
Login session handle

in_handle
Handle to get info for

in_flags
CAPI_HANDLE_TYPE, CAPI_HANDLE_NAME or CAPI_HANDLE_MAILTO

out_info
Read-only handle information

Returns

CAPIStatus

Changes

CAPI 2.5: now returns CAPI_STAT_DATA_EMAIL_NOTSET if no e-mail address is set on the server.

Sample

Print the name of the logged in user:

```
{
    CAPIHandle   loginUser = CSDK_HANDLE_INITIALIZER;
    const char * fullName = NULL;
```

```
    stat = CSDK_GetHandle(mySession, NULL, CSDK_FLAG_NONE, &loginUser);
    stat = CSDK_HandleInfo(mySession, loginUser, CAPI_HANDLE_NAME, &fullName);
    cout << "Currently logged in as " << fullName << endl;
    CSDK_DestroyHandle(mySession,
                       &loginUser);
}
```

Sample

Print out Doctor Winston's e-mail address:

```
{
    CAPIHandle   doctor = CSDK_HANDLE_INITIALIZER;
    const char * email  = NULL;
    stat = CAPI_GetHandle(mySession, "drwinston", CSDK_FLAG_NONE, &doctor);
    stat = CAPI_HandleInfo(mySession, doctor, CAPI_HANDLE_MAILTO, &email);
    cout << "drwinston's email address is " << email << endl;
    CSDK_DestroyHandle(mySession,
                       &doctor);
}
```

CSDK_GetNextFailure

Returns the next failure contained in a CSDKRequestResult.

```
CAPIStatus CSDK_GetNextFailure (
  CSDKRequestResult in_requestResult,
  CAPIHandle * out_user,
  const char ** out_uid,
  CAPIStatus * out_status
)
```

A call to CSDK_GetFirstFailure must precede this call.

Note: A request result contains the reference to the "current" failure, so only one thread should extract failures from a given request result at a time.

Parameters

in_requestResult
The RequestResult from which to extract information

out_user
The user whose agenda was being read or written

out_uid
The uid of the calendar object being read or written, if applicable

out_status
The status code for the portion of the operation involving the user and uid specified

Returns

CAPIStatus

Return values

CAPI_STAT_API_NULL
One of the required parameters has a NULL value

CAPI_STAT_DATA_RRESULT_EOR
No more failure in the RequestResult

Sample

Get all failures from a call to CSDK_StoreContacts by calling CSDK_GetNextFailure in a while loop

```
const char * vcardUID = 0;
CAPIStatus   stat = CAPI_STAT_OK;
CSDKRequestResult * myRequestResult = 0;
//
stat = CSDK_StoreContacts(mySession,
                          myStream,
                          CSDK_FLAG_STORE_IMPORT,
                          &myRequestResult);
//
stat = CSDK_GetFirstFailure(myRequestResult,
                            NULL,
```

```
                                &vcardUID,
                                &vcardStatus);
  //
  while (stat != CAPI_STAT_DATA_RRESULT_EOR)
  {
      const char * statusName = 0;
      CSDK_GetStatusString(vcardStatus, &statusName);
      cout << "Store of VCARD with UID " << vcardUID << " failed with status " <<
statusName << endl;
      //
      stat = CSDK_GetNextFailure(myRequestResult,
                                 NULL,
                                 &vcardUID,
                                 &vcardStatus);
  }
  //
  CSDK_DestroyResult(&myRequestResult);
```

Equivalent Java Method

oracle.calendar.sdk.Result.getNextFailure()

CSDK_GetNextParseError

Returns the next parsing error obtained from a request result.

```
CAPIStatus CSDK_GetNextParseError (
  CSDKRequestResult in_requestResult,
  CAPIStatus * out_status,
  const char ** out_errorBuffer,
  const char ** out_errorLocation,
  const char ** out_message
)
```

A call to CSDK_GetFirstParseError must precede this call.

Note: A request result contains the reference to the "current" parse error, so only one thread should extract parse errors from a given request result at a time.

A pointer to a copy of the data stream is returned through out_errorBuffer, and a pointer to the parse error location in the buffer is returned via out_errorLocation. Both pointers are valid only until the request result is destroyed.

Parameters

in_requestResult
The RequestResult to extract information from

out_status
The result's status

out_errorBuffer
The beginning of the buffer with the error

out_errorLocation
The location in *out_errorBuffer where the error occurred

out_message
May contain additional information (NULL may be returned if no message is available)

Returns

CAPIStatus

Return values

CAPI_STAT_API_NULL
One of the required parameters has a NULL value

CAPI_STAT_DATA_RRESULT_EOR
No parse errors in in_requestResult

Sample

Get all the parsing errors from a call to CSDK_StoreContacts by calling CSDK_ GetNextParseError in a while loop

```
  CAPIStatus          stat = CAPI_STAT_OK;
  CSDKRequestResult * myRequestResult = 0;
```

```
  //
  stat = CSDK_StoreContacts(mySession,
                            myStream,
                            CSDK_FLAG_STORE_IMPORT,
                            &myRequestResult);
  //
  const char * buffer = 0;
  const char * errorLocation = 0;
  const char * message = 0;
  //
  CAPIStatus parseStat = CSDK_GetFirstParseError(myRequestResult,
                                                 NULL,
                                                 &buffer,
                                                 &errorLocation,
                                                 &message);
  //
  while (parseStat != CAPI_STAT_DATA_RRESULT_EOR)
  {
      cout << "Error (" << message << ") parsing vCard.  Buffer:'" << vcardUID <<
"'  Error starting at:'" << errorLocation << "'" << endl;
      parseStat = CSDK_GetNextParseError(myRequestResult,
                                         NULL,
                                         &buffer,
                                         &errorLocation,
                                         &message);
  }
  //
  CSDK_DestroyResult(&myRequestResult);
```

Equivalent Java Method

oracle.calendar.sdk.Result.getNextParseError()

CSDK_GetNextResult

Returns the next result contained in a CSDKRequestResult.

```
CAPIStatus CSDK_GetNextResult (
  CSDKRequestResult in_requestResult,
  CAPIHandle * out_user,
  const char ** out_uid,
  CAPIStatus * out_status
)
```

A call to CSDK_GetFirstResult must precede this call.

Note: A request result contains the reference to the "current" result, so only one thread should extract result from a given request result at a time.

Parameters

in_requestResult
The RequestResult from which to extract information

out_user
The user whose agenda was being read or written

out_uid
The uid of the calendar object being read or written, if applicable

out_status
The status code for the portion of the operation involving the user and uid specified

Returns

CAPIStatus

Return values

CAPI_STAT_API_NULL
One of the required parameters has a NULL value

CAPI_STAT_DATA_RRESULT_EOR
No more results in the RequestResult

Sample

Get all results from a call to CSDK_StoreContacts by calling CSDK_GetNextResult in a while loop:

```
const char * vcardUID = 0;
CAPIStatus   stat = CAPI_STAT_OK;
CSDKRequestResult * myRequestResult = 0;
//
stat = CSDK_StoreContacts(mySession,
                          myStream,
                          CSDK_FLAG_STORE_IMPORT,
                          &myRequestResult);
//
stat = CSDK_GetFirstResult(myRequestResult,
                           NULL,
```

```
                              &vcardUID,
                              &vcardStatus);
  //
  while (stat != CAPI_STAT_DATA_RRESULT_EOR)
  {
      const char * statusName = 0;
      CSDK_GetStatusString(vcardStatus, &statusName);
      cout << "Store of VCARD with UID " << vcardUID << " returned status " << sta
tusName << endl;
      //
      stat = CSDK_GetNextResult(myRequestResult,
                                NULL,
                                &vcardUID,
                                &vcardStatus);
  }
  //
  CSDK_DestroyResult(&myRequestResult);
```

Equivalent Java Method

oracle.calendar.sdk.Result.getNextResult()

CSDK_GetStatusCode

A status returned by the CALENDAR_SDK is composed of a status code and some extra bits giving extra context to the error that occurred.

```
CSDK_GetStatusCode (
  CAPIStatus in_status,
  int * out_statusCode
)
```

This can cause the API user to have to mask these extra bits to compare the status codes. So this helper returns the status without these extra bits s.t. it is comparable with other status codes.

Parameters

in_status
CAPI status

out_statusCode
contains a statusCode with extra level bits removed

Equivalent Java Method

oracle.calendar.sdk.Api.getStatusCode()

CSDK_GetStatusLevels

Decomposes a CAPIStatus into its subparts; each part of the status code specifies more precisely the actual error.

```
CSDK_GetStatusLevels (
 CAPIStatus in_status,
 unsigned long * out_field1,
 unsigned long * out_field2,
 unsigned long * out_field3,
 unsigned long * out_field4,
 unsigned long * out_field5
)
```

Parameters

in_status
CAPI status

out_field1
Contains the int result for level1

out_field2
Contains the int result for level2

out_field3
Contains the int result for level3

out_field4
Contains the int result for level4

out_field5
Contains the int result for level5

Equivalent Java Method

oracle.calendar.sdk.Api.getStatusLevels()

Changes

CAPI 2.5: types of "out_level[12345]" changed from "int *" to "unsigned long *"

CSDK_GetStatusString

Returns a read-only string representation of a CAPIStatus (this is generally more useful than the numeric representation).

```
CSDK_GetStatusString (
  CAPIStatus in_status,
  const char ** out_errorString
)
```

Parameters

in_status
CAPI status

out_errorString
Contains const pointer to the result string

Cleanup

None. The string returned is a const string that cannot be freed.

Equivalent Java Method

oracle.calendar.sdk.Api.getStatusString()

CSDK_SetConfigFile

Allows the SDK to read configuration settings that control error logging and the other configuration parameters listed in the "Configuration" section of this manual.

```
CAPIStatus CSDK_SetConfigFile (
  const char * in_configFileName,
  const char * in_logFileName
)
```

If called, this function should be the first SDK function called by your process and should not be called by each thread.

If you require assistance from Oracle Support for your development with the SDK, you should expect to be asked to call this function in order to generate logs.

Parameters

in_configFileName
A null-terminated string containing the filename of the config file.

in_logFileName
The name of a file to write log messages to. If this file cannot be created or written to, output will be sent to a file named Console.log in the current directory.

Returns

CAPIStatus

Return values

CAPI_STAT_API_NULL
One of the input parameters was NULL

CAPI_STAT_CONFIG_CANNOT_OPEN
Failed to open in_configFileName

Equivalent Java Method

oracle.calendar.sdk.Api.init()

See

The Configuration section.

Sample

Create a file "capi.ini" with the contents:

```
[LOG]
log_activity = true
log_modulesinclude = { CAPI }
```

Sample

Call CSDK_SetConfigFile (after creating capi.ini). This will turn on "activity" level logging in CAPI and the output will go into capi.log:

```
CAPIStatus status = CSDK_SetConfigFile("capi.ini", "capi.log");
```

Sample

Using absolute and relative paths:

```
CAPIStatus status = CSDK_
SetConfigFile("../config/csdk.ini", "/var/log/csdk.log");
```

CSDK_SetIdentity

Allows an authenticated user to work on behalf of another calendar user or resource.

```
CAPIStatus CSDK_SetIdentity (
  CAPISession in_session,
  const char * in_user,
  CAPIFlag in_flags
)
```

For subsequent calls to this function to work, designate rights must have been granted to the authenticated user.

The format of the in_user parameter is the same as in the CSDK_Authenticate function. The authenticated user may revert to his or her original identity by using NULL as username.

If you've logged in as Calendar SYSOP (CSDK_ConnectAsSysop), then designate rights are ignored and you will be able to work as any Calendar user or resource. All Calendar operations will appear to have been done by the user, rather than on behalf of the user by a designate.

Parameters

in_session
Login session handle

in_user
Person (or resource) to work as, an X400 or UID

in_flags
Bit flags (CSDK_FLAG_NONE at this time)

Returns

CAPIStatus

Sample

Work on behalf of another user:

```
 myStatus = CSDK_SetIdentity(mySession, "keithm", CSDK_FLAG_NONE);
 myStatus = CSDK_SetIdentity(mySession, "?/S=MacDonald/G=Keith/", CSDK_FLAG_NONE);
 myStatus = CSDK_SetIdentity(mySession, "?/RS=Conference Room/ND=1234/", CSDK_
FLAG_NONE);
```

Changes

CAPI 2.5: Resource names must be an exact match. (There used to be an implicit wildcard at the end of the string.)

Changes

9.0.4: SetIdentity can be used to work on behalf of a user on another node using designate rights. This does NOT apply to connections opened via CSDK_ ConnectAsSysop().

Equivalent Java Method

oracle.calendar.sdk.Session.setIdentity()

CSDK_StoreContacts

Stores vCards on a server through an authenticated connection by in_session; The vCards must be passed in via a CAPIStream.

```
CAPIStatus CSDK_StoreContacts (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

By default, the incoming stream is assumed to be MIME-encapsulated vCard. When storing a stream that is not MIME-encapsulated, specify the flag CSDK_FLAG_STREAM_NOT_MIME.

Versions 2.1 and 3.0 of vCard are supported.

When storing multiple vCards, every vCard must be in a separate MIME part and any MIME part containing a vCard to be stored must contain the "Content-Type: text/x-vcard" header. The only supported character sets for the MIME parts are UTF-8 and US-ASCII.

The following are the store modes that can be used:

- CAPI_FLAG_NONE: A regular store of one or more vCards. If the vCard contains a UID property, that property is read and a verification is made to ensure that the contact does not already exist on the server. If it does, CAPI_STAT_DATA_VCARD_DUPERROR is returned and the contact is not stored on the server.
- CSDK_FLAG_STORE_REPLACE: This completely replaces a vCard that already exists on the server. It reads the UID contained within the UID property of the given vCard, removes that contact from the server, and stores the new one. An error is returned if no contact with the given UID exists on the server.
- CSDK_FLAG_STORE_MODIFY: Updates a contact already on the server with the new vCard. The UID is extracted in the same way as in _MODIFY and _NONE. The vCard with that UID is then updated: all properties contained in the vCard on the server that are present in the passed-in vCard are modified to contain the property values of the passed-in vCard. Also, All properties that exist in the passed-in vCard that don't exist on the server vCard are added to the server vCard. All other properties not present in the passed-in vCard that exist on the server are ignored.
- CSDK_FLAG_STORE_REMOVE: The contact on the server is fetched, the properties contained within the passed-in vCard are deleted from the fetched vCard, and then the fetched vCard is stored onto the server.
- CSDK_FLAG_STORE_IMPORT: This mode checks if the contact already exists on the server via the UID. If it does, then it acts exactly as if CSDK_FLAG_STORE_REPLACE was passed in. Otherwise, it acts exactly like as if CSDK_FLAG_STORE_CREATE was passed in.

The flags supplied are used for each vCard supplied. Results are written to the CSDKRequestResult.

This version of the SDK cannot preserve the supplied UIDs when adding contacts to the server. This ability is planned for the next major release of the SDK with the next major server release.

The CSDKRequestResult contains the UIDs which can be used to refer to the stored vCards.

The CSDKRequestResult may contain information about errors parsing the vCard streams.

This function is blocked for SYSOP that has not assumed the identity of a user.

Parameters

in_session
Login session handle

in_flags
Flags modifying behavior. One of the following:

- CSDK_FLAG_STORE_CREATE: Create if no task with the given UID exists, otherwise return an error
- CSDK_FLAG_STORE_REPLACE: Completely replace task on server with this copy - error if object doesn't exist
- CSDK_FLAG_STORE_IMPORT: If task exists, CSDK_FLAG_STORE_REPLACE, else CSDK_FLAG_STORE_CREATE
- CSDK_FLAG_STORE_MODIFY: Modify only the supplied properties of an existing task
- CSDK_FLAG_STORE_REMOVE: Remove specified properties

and optionally:

- CSDK_FLAG_STREAM_NOT_MIME: Incoming stream is not inside a MIME wrapper

in_stream
The stream containing vCards

out_requestResult
Pointer to a RequestResult that will get filled (pass NULL if you don't want this information returned).

Returns

CAPIStatus

Return values

CAPI_STAT_API_HANDLE_NULL
The session was NULL

CAPI_STAT_API_STREAM_NULL
The stream was NULL

CAPI_STAT_DATA_VCARD_DUPERROR
Tried to store vCard in CSDK_FLAG_STORE_CREATE mode that already exists on the server.

CAPI_STAT_DATA_UID_NOTFOUND
Tried to update, replace, or delete properties but no UID was found in the passed-in vCard.

CAPI_STAT_API_BADPARAM
Invalid flag, or multiple store flags were set

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Sample

```
Store a non-MIME formatted contact:
 stat = CSDK_StoreContacts(mySession,
                           myStream,
                           CSDK_FLAG_STORE_IMPORT | CSDK_FLAG_STREAM_NOT_MIME,
                           &myRequestResult);
```

Sample

Store a contact with vCard:

```
strcpy(outVCard,
"MIME-Version: 1.0\015\012\
Content-Type: multipart/mixed;\015\012\
boundary=\"------------CA94974D4D8713DE5B12E6CD\"\015\012\
\015\012\
This is a multi-part message in MIME format.\015\012\
--------------CA94974D4D8713DE5B12E6CD\015\012\
Content-Type: text/x-vcard; charset=UTF-8;\015\012\
name=\"example.vcf\"\015\012\
Content-Disposition: attachment;\015\012\
filename=\"example.vcf\"\015\012\
Content-Transfer-Encoding: quoted-printable\015\012\
\015\012\
BEGIN:VCARD\015\012\
URL:http://www.somewebsite.com\015\012\
ORG:steltor;windows;\015\012\
TITLE:worker\015\012\
EMAIL;TYPE=INTERNET:someone@somewhere.com\015\012\
ADR;TYPE=WORK:;;who knows;snodown;qc;h1l 2H1;Canada\015\012\
NOTE;ENCODING=QUOTED-PRINTABLE;:This is a note\015\012\
N;ENCODING=QUOTED-PRINTABLE:Last;First;Middle\015\012\
FN;ENCODING=QUOTED-PRINTABLE:First Middle Last\015\012\
REV:20011105T145136Z\015\012\
VERSION:2.1\015\012\
END:VCARD\015\012\
\015\012\
--------------CA94974D4D8713DE5B12E6CD--\015\012\015\012");
//
CAPIStatus stat;
//
stat = CAPI_CreateMemoryStream(mySession,
                               &myStream,
                               outVCard,
                               NULL,
                               CAPI_FLAG_NONE);
//
stat = CSDK_StoreContacts(mySession,
                          myStream,
                          CSDK_FLAG_STORE_IMPORT,
```

```
                                        &myRequestResult);
```

Equivalent Java Method

oracle.calendar.sdk.Session.storeContacts()

CSDK_StoreEvents

This function reads one VCALENDAR object from in_stream and stores each contained VEVENT on the server.

```
CAPIStatus CSDK_StoreEvents (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

Attendees

Unlike CAPI_StoreEvent, the only attendees of the event will be those specified using ATTENDEE properties in the iCalendar (with the exception of the case where CSDK_FLAG_STORE_INVITE_SELF is used, in which case the logged-in user will always be invited regardless of whether an ATTENDEE property is supplied for that user).

The address specified in the ATTENDEE properties is used to identify calendar users. If no calendar user exists with the address specified in the ATTENDEE property value, then the attendee is considered "external" and will be invited using an Internet standard protocol such as iMIP if the Oracle Calendar server is capable of doing so.

The ATTENDEE PARTSTAT parameter is ignored except for the following:

- The logged in user
- External (non-calendar) attendees

Resources

The Oracle Calendar server stores a PARTSTAT value for each resource, but resources do not have e-mail addresses. To permit the usage of the ATTENDEE property for inviting resources, the following syntax is supported:

ATTENDEE;CUTYPE=RESOURCE;CN=projecter:MAILTO:ignored@foobar.com

Groups

Oracle Calendar Groups can be invited using a non-standard (but legal) ATTENDEE property of the following form:

ATTENDEE;CUTYPE=GROUP;CN=developers:MAILTO:ignored@foobar.com

As suggested by the example, the property value (MAILTO:ignored@foobar.com) is NOT used. In this example, the group "developers" will be expanded and the members will be invited as calendar users. When fetching this event, the members of the group (at the time of the call to CSDK_StoreEvents) will be returned as individual ATTENDEE properties.

The server does not enforce uniqueness of group names - if multiple matches are found, an error will be returned.

Errors

Detailed error information is returned through the out_requestResult parameter. Unless a parse error result is returned, there will be at least one result per VEVENT stored, containing a CAPI_STATUS value for storing that VEVENT. Passing in a NULL (zero) value for out_requestResult will prevent the request results from being returned, but is not considered an error.

Recurrence Rules

Recurrence rules (RRULE) are supported by this function and require that the event's DTSTART be specified in local (using a TZID=... and a VTIMEZONE component) or floating time (as per RFC 2445). A limitation of the Oracle Calendar server requires that no more that one RRULE can be specified for a given VEVENT, nor can the RRULE be changed when modifying an event (the only way to change the occurrences is to use RDATEs and/or EXDATEs).

UIDs

The Oracle Calendar server prevents any user/resource from owning more than one event with a given UID. However, UIDs are not necessarily unique on the Oracle Calendar server, so a user/resource may be invited to more than one event with a given UID. Users of the SDK should attempt to provide globally unique UIDs when adding events to the Oracle Calendar server.

Storing an event without a UID will result in a new UID being generated by the Oracle Calendar server and there will be a small performance penalty. The generated UIDs are returned as part of the results in out_requestResult.

Per-Instance Properties (DESCRIPTION and ATTACH)

Currently there are only two such properties, DESCRIPTION and ATTACH. These properties are in one of the following states for a given instance:

- The property is not defined in the instance
- The property is defined in the instance
- The property is defined in the event (the instance uses the event's property)

If an event has only one instance, by default, that instance property becomes the event's property.

To replace or modify these properties, use the flags CSDK_FLAG_STORE_MODIFY and CSDK_FLAG_STORE_REPLACE.

To modify, replace, or remove a property defined in an instance, the RECURRENCE-ID of the instance must be specified in the VEVENT passed as input.

To modify, replace, or remove a property defined in an event, **do not** specify a RECURRENCE-ID in the VEVENT passed in input.

Example 1: You have a meeting with three instances and all instances use the ATTACH property defined in the event. This property refers to the file attach1.txt. You want to modify the attachment of that event (for each of the 3 instances) so that the property refers to the file attach2.txt instead. You therefore need to use the flag CSDK_FLAG_STORE_MODIFY and a VEVENT without a RECURRENCE-ID that contains an ATTACH property that refers to attach2.txt. The ATTACH property will be modified at the event level, so all three instances will now refer to attach2.txt.

Example 2: You have a meeting with three instances and all the instances use the event's ATTACH property. This property refers to attach1.txt. You want to modify the ATTACH property of the third instance so that it refers to attach2.txt. You therefore need to use the flag CSDK_FLAG_STORE_MODIFY and a VEVENT with the RECURRENCE-ID of the third instance that contains an ATTACH property that refers to the new attachment. The ATTACH property will be modified for the third instance, which will now refer to attach2.txt. The first two instances will still refer to attach1.txt.

How Can I...

- add instances to an event?

  - For recurring events (which use RRULEs), simply store a VEVENT with the event's UID and one or more RDATE properties using the flag CSDK_FLAG_STORE_MODIFY.
  - For repeating events (not using RRULEs), store a VEVENT with the event's UID and the flag CSDK_FLAG_STORE_MODIFY.
- add attendees to an event?
  - Store VEVENT(s) with ATTENDEE properties for new attendees using flag CSDK_FLAG_STORE_MODIFY. RECURRENCE-ID property can be specified in the VEVENT to invite the attendee to only the specified instance.
- remove attendees from an event?
  - Fetch event, remove the ATTENDEE property for the user to uninvite, then store using the mode CSDK_FLAG_STORE_REPLACE

Parameters

in_session
Login session handle

in_flags
Bit flags modifying behavior. It may be one of the following:

- CSDK_FLAG_STORE_CREATE: Create if no event with the given UID exists, otherwise return an error
- CSDK_FLAG_STORE_REPLACE: Completely replace event on server with this copy, error if object doesn't exist
- CSDK_FLAG_STORE_IMPORT: If event exists, CSDK_FLAG_STORE_REPLACE, else CSDK_FLAG_STORE_CREATE
- CSDK_FLAG_STORE_MODIFY: Modify only the supplied properties of an existing event
- CSDK_FLAG_STORE_REPLY: Any attendee of an event can use this mode to update their own attendance status and alarms

and optionally, a combination of:

- CSDK_FLAG_STORE_INVITE_SELF: Add current user as an attendee, even if no ATTENDEE is in the iCal
- CSDK_FLAG_STREAM_NOT_MIME: Incoming stream is not inside a MIME wrapper
- CAPI_NOTIFY_EMAIL: Send e-mail notification (default is to NOT send)
- CAPI_NOTIFY_SMS: Send SMS notification (default is to NOT send)

in_stream
Stream for CAPI to read data from

out_requestResult
If non-NULL, will be filled in with detailed results of the transaction. This may include error messages from reading the iCalendar data or any other errors encountered while processing the request.

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Sample

Simple case of adding an event into the current user's calendar:

```
  {
      static const char * ical = {"BEGIN:VCALENDAR\\r\\n"
                                  "VERSION:2.0\\r\\n"
                                  "BEGIN:VEVENT\\r\\n"
                                  "DTSTART:20021225T100000Z\\r\\n"
                                  "DTEND:20021225T233000Z\\r\\n"
                                  "SUMMARY:work\\r\\n"
                                  "LOCATION:office\\r\\n"
                                  "END:VEVENT\\r\\n"
                                  "END:VCALENDAR\\r\\n"};
      //
      CAPIStream memoryStream = CSDK_STREAM_INITIALIZER;
      status = CSDK_CreateMemoryStream(mySession,
                                       &memoryStream,
                                       ical,
                                       NULL,
                                       CSDK_FLAG_NONE);
      if (!status)
      {
          status = CSDK_StoreEvents(mySession,
                                    CSDK_FLAG_STORE_CREATE | CSDK_FLAG_STORE_
INVITE_SELF | CSDK_FLAG_STREAM_NOT_MIME,
                                    memoryStream,
                                    NULL);
          //
          CSDK_DestroyStream(mySession,
                             &memoryStream);
      }
  }
```

Sample

Invite several people to a meeting

```
  {

      const char * attendees[] = {"?/S=Who/G=Cindy Lou/",
                                  "?/S=Who/G=Lou Lou/",
                                  "?/S=Who/G=Betty Lou/"
                                  "grinch");
      //
      const int     numAttendees   = (sizeof(attendees) / sizeof(attendees[0]));
      const char ** emailAddresses = (const char **)malloc((numAttendees + 1) * si
zeof(const char *));
      CAPHandle  *  handles        = (CAPHandle *)malloc(numAttendees * sizeof(CAP
Handle *));
      //
      // get handles:
      for (int i = 0; !status && (i < numAttendees); i++)
```

```
      {
          status = CSDK_GetHandle(mySession, attendees[i], CSDK_FLAG_
NONE, &handles[i]);
      }
      //
      // terminate the array:
      handles[numAttendees] = CSDK_HANDLE_INITIALIZER;
      //
      if (!status)
      {
          // get e-mail addresses for each handle using CSDK_GetHandleInfo()
          ...
      }
      //
      if (!status)
      {
          static const char * iCalEvent1 = {"BEGIN:VCALENDAR\\r\\n"
                                            "VERSION:2.0\\r\\n"
                                            "BEGIN:VEVENT\\r\\n"
                                            "DTSTART:20021225T100000Z\\r\\n"
                                            "DTEND:20021225T103000Z\\r\\n"
                                            "SUMMARY:SING\\r\\n"
                                            "LOCATION:Town square\\r\\n"};
          //
          static const char * iCalEvent2 = {"END:VEVENT\\r\\n"
                                            "END:VCALENDAR\\r\\n"};
          //
          string iCalEvent = iCalEvent1;
          for (int attendee = 0; attendee < numAttendees; attendee++)
          {
              if (emailAddresses[attendee])
              {
                  iCalEvent += "ATTENDEE:mailto:";
                  iCalEvent += emailAddresses[attendee];
                  iCalEvent += "\\r\\n";
              }
          }
          iCalEvent += iCalEvent2;
      }
      //
      if (emailAddresses)
      {
          free(emailAddresses);
      }
      //
      CSDK_DestroyMultipleHandles(mySession,
                                  handles,
                                  numAttendees,
                                  CSDK_FLAG_NONE);
      if (handles)
      {
          free(handles);
      }
      //
      if (!status)
      {
          CAPIStream memoryStream = CSDK_STREAM_INITIALIZER;
          status = CSDK_CreateMemoryStream(mySession,
                                           &memoryStream,
                                           iCalEvent.c_str(),
```

```
                                          NULL,
                                          CSDK_FLAG_NONE);
          if (!status)
          {
              status = CSDK_StoreEvents(mySession,
                                        CSDK_FLAG_STORE_CREATE | CSDK_FLAG_STREAM_
NOT_MIME,
                                        memoryStream,
                                        NULL);
          }
          //
          CSDK_DestroyStream(mySession,
                             &memoryStream);
      }
  }
```

Equivalent Java Method

oracle.calendar.sdk.Session.storeEvents

CSDK_StoreTasks

Creates/modifies tasks on the current user's agenda depending on the store flag passed in.

```
CAPIStatus CSDK_StoreTasks (
  CAPISession in_session,
  CAPIFlag in_flags,
  CAPIStream in_stream,
  CSDKRequestResult * out_requestResult
)
```

Only one store flag should be used. If multiple flags are passed the error CAPI_STAT_API_FLAGS will be returned. There are five possible flags that can be used:

- CSDK_FLAG_STORE_IMPORT: Stores the task if it does not exist and replaces the task if it exists.
- CSDK_FLAG_STORE_CREATE: Stores the task. If the task exists the error will be returned.
- CSDK_FLAG_STORE_REPLACE: Replaces the existing task. If the task does not exist the error is returned.
- CSDK_FLAG_STORE_MODIFY: Modifies specified properties
- CSDK_FLAG_STORE_REMOVE: Deletes specified properties

Other flags may be specified along with one of the above store flags:

- CSDK_FLAG_STREAM_NOT_MIME: Incoming stream is not inside a MIME wrapper

This function is blocked for SYSOP that has not assumed the identity of a user.

Parameters

in_session
Login session handle

in_flags
Store bit flags

in_stream
Input stream

out_requestResult
Returned request result object (may be NULL)

Returns

CAPIStatus

Cleanup

The request result must be destroyed using CSDK_DestroyRequestResult

Return values

CAPI_STAT_OK

CAPI_STAT_API_SESSION_NULL
in_session is NULL

CAPI_STAT_API_STREAM_NULL
in_inputStream is NULL

CAPI_STAT_API_FLAGS
in_flags is invalid

Equivalent Java Method

oracle.calendar.sdk.Session.storeTasks

5

Oracle Calendar SDK Flags, Capabilities and Type Definitions

This chapter documents all the flags, capabilities, and type definitions of Oracle Calendar SDK.

Oracle Calendar SDK Flags

The following flags are defined in the ctapi.h header file.

Table 5–1 Defines

Define Declaration	Description
CSDK_FLAG_CONTINUE_ON_ERROR	Can be passed to CSDK_Fetch*ByUID or CSDK_Delete*ByUID calls to request that, if possible, we continue the operations even if we encountered an error. Furthermore we return partial data if applicable (CSDK_ Fetch*ByUID). Example, if we do a CSDK_Fetch*ByUID with multiple UID, and there is a UID that is not found. Instead of the function returning a STATUS CODE and no data (default behavior), we would return STATUS CODE CSDK_STAT_OK (error would only be reported in RequestResult) and the partial data found for the valid UID's.
CSDK_FLAG_FETCH_AGENDA_ATTENDEE_ONLY	Used with CSDK_FetchEvent* calls to filter out ATTENDEE properties for all attendees other than the agenda being viewed.
CSDK_FLAG_FETCH_COMBINED	Used with CSDK_FetchEvent* calls to return all events in a single VCALENDAR rather than one VCALENDAR per agenda. This is faster and produces smaller output.
CSDK_FLAG_FETCH_DO_NOT_EXPAND_RRULE	Can be passed to CSDK_FetchEvent* calls to request that recurrence rules not be expanded. This flag is set by default with CSDK_FetchEventsByUID and can be overridden by using CSDK_FLAG_FETCH_EXPAND_ RRULE.
CSDK_FLAG_FETCH_EXCLUDE_ACCEPTED	Used with CSDK_FetchEvent* calls to exclude events the caller has accepted.
CSDK_FLAG_FETCH_EXCLUDE_APPOINTMENTS	Used with CAPI_FetchEvent* and CSDK_FetchEvent* calls to exclude regular meetings (appointments).
CSDK_FLAG_FETCH_EXCLUDE_DAILYNOTES	Used with CAPI_FetchEvent* and CSDK_FetchEvent* calls to exclude daily notes.

Table 5–1 (Cont.) Defines

Define Declaration	Description
CSDK_FLAG_FETCH_EXCLUDE_DAYEVENTS	Used with CAPI_FetchEvent* and CSDK_FetchEvent* calls to exclude day events.
CSDK_FLAG_FETCH_EXCLUDE_DECLINED	Used with CSDK_FetchEvent* calls to exclude events the caller has declined.
CSDK_FLAG_FETCH_EXCLUDE_HOLIDAYS	Used with CAPI_FetchEvent* and CSDK_FetchEvent* calls to exclude holidays.
CSDK_FLAG_FETCH_EXCLUDE_NOTOWNER	Used with CSDK_FetchEvent* calls to exclude events which are not owned by the caller.
CSDK_FLAG_FETCH_EXCLUDE_UNCONFIRMED	Used with CSDK_FetchEvent* calls to exclude events the caller has not confirmed.
CSDK_FLAG_FETCH_EXPAND_RRULE	Can be passed to CSDK_FetchEvent* calls to request that recurrence rules be expanded. This flag is set by default with CSDK_FetchEventsByRange and CSDK_FetchEventsByAlarmRange and can be overridden by using CSDK_FLAG_FETCH_DO_NOT_EXPAND_RRULE.
CSDK_FLAG_FETCH_LOCALTIMES	Used with CSDK_FetchEvent* and CSDK_FetchTask* calls to request that times be returned in the "local" timezone. The current user's preferred timezone as set on the Oracle Calendar server is considered the "local" timezone.
CSDK_FLAG_FETCH_RESOURCES_WITHOUT_ADDRESSES	Can be passed to CSDK_FetchEvent* calls to request that resources without email addresses not be exported using invalid email addresses.
CSDK_FLAG_FETCH_USERS_WITHOUT_ADDRESSES	Can be passed to CSDK_FetchEvent* calls to request that users without email addresses not be exported using invalid email addresses.
CSDK_FLAG_FETCH_VCARD_VERSION_2_1	Used with CSDK_FetchContacts* calls to request version 2.1 vCards.
CSDK_FLAG_FETCH_VCARD_VERSION_3_0	Used with CSDK_FetchContacts* calls to request version 3.0 vCards (default).
CSDK_FLAG_NONE	Used to select the default behavior (same as CAPI_FLAG_NONE).
CSDK_FLAG_STORE_CREATE	Used with CSDK_Store* calls to create a new object on the server. If an object already exists with the same UID, an error will be returned
CSDK_FLAG_STORE_IMPORT	Used with CSDK_Store* calls to create (CAPI_FLAG_STORE_CREATE) a new object (task/contact) on the server if none exists with the given UID, or to completely replace (CAPI_FLAG_STORE_REPLACE) an existing object.
CSDK_FLAG_STORE_INVITE_SELF	Used with CSDK_StoreEvents to invite the current user without requiring an ATTENDEE property.

Table 5–1 (Cont.) Defines

Define Declaration	Description
CSDK_FLAG_STORE_MODIFY	Used with CSDK_Store* calls to add, or modify the given properties to an existing object on the server.
CSDK_FLAG_STORE_REMOVE	Used with CSDK_Store* calls to remove the given properties from an existing object on the server. The UID property must be specified in the input, but will not itself be removed from the server. This mode will not completely delete an object on the server - use CSDK_Delete* instead.
CSDK_FLAG_STORE_REPLACE	Used with CSDK_Store* functions to specify that the object (event/task/contact) on the server should be completely replaced by the given object.
CSDK_FLAG_STORE_REPLY	Used with CSDK_StoreEvents to reply (set ATTENDEE and VALARM) to events organized by other people.
CSDK_FLAG_STREAM_NOT_MIME	Used with CSDK_Store* and CSDK_Fetch* calls to specify that the stream should not be MIME encapsulated.

CSDK Capabilities

The following capabilities are defined in the ctapi.h header file.

Table 5–2 Calendar SDK Capabilities

Define Declaration	Description
CAPI_CAPAB_ABOUT_BOX	Returns information about CAPI.
CAPI_CAPAB_AUTH	Returns the authentication mechanisms supported by the server (e.g. "cs-standard,gssapi:kerberos5,sasl:KERBEROS_V4"). A server connection must exist to read this capability.
CAPI_CAPAB_CAPI_VERSION	Returns the SDK version as a string. (e.g. "9.0.4")
CAPI_CAPAB_COMP	Returns the compression mechanisms supported by the server (e.g. "cs-simple,none"). A server connection must exist to read this capability.
CAPI_CAPAB_ENCR	Returns the encryption mechanisms supported by the server (e.g. "cs-light,none"). A server connection must exist to read this capability.
CAPI_CAPAB_MAXDATE	Returns the largest date which CAPI can handle ("20371129").
CAPI_CAPAB_MINREFRESHRATE	Returns the minimum idle refresh rate for SDK clients.
CAPI_CAPAB_SERVER_VERSION	Returns the server version as a string. (e.g. "6.0"). A server connection must exist to read this capability.

Table 5–2 (Cont.) Calendar SDK Capabilities

Define Declaration	Description
CAPI_CAPAB_UNSUPPORTED_ICAL_COMP	Returns a comma delimited list of iCal components which CAPI does not process. ("VJOURNAL,VFREEBUSY")
CAPI_CAPAB_UNSUPPORTED_ICAL_PROP	Returns a comma delimited list of iCal properties which CAPI does not process. ("GEO,COMMENT"). A server connection must exist to read this capability.
CAPI_CAPAB_VERSION	Same as CAPI_CAPAB_CAPI_VERSION.

Oracle Calendar SDK Type Definitions

The Oracle Calendar SDK has the following type definitions defined:

Table 5–3 Typedefs

Typedef Declaration	Description
typedef int(* CAPICallback)(void *in_userData, char *io_data, size_t in_dataSize, size_t *out_datSize)	
typedef long CAPICapabilityID	
typedef unsigned long CAPIFlag	
typedef void* CAPIHandle	
typedef void* CAPISession	
typedef unsigned long CAPIStatus	
typedef void* CAPIStream	
typedef char const* const* CAPIUIDSet	
typedef struct CSDKCondition { const char * prop; CSDKOperator op; const char * value; } CSDKCondition	
typedef int CSDKOperator	
typedef void* CSDKQuery	
typedef struct ScapiAbstractRequestResult* CSDKRequestResult	

6

Oracle Calendar SDK Status Codes

This chapter documents all CAPIStatus values that may be returned by the SDK functions, in alphabetical order. The functions CSDK_GetStatusString and CSDK_GetStatusLevels may be useful when interpreting CAPIStatus values.

Status Codes

The following codes are defined in the ctapi.h header file.

Table 6–1 Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_API	API class status.
CAPI_STAT_API_BADPARAM	A bad parameter was passed.
CAPI_STAT_API_CALLBACK	There was a problem with a callback.
CAPI_STAT_API_CALLBACK_ERROR	The callback returned an error, which is returned in bit field 5.
CAPI_STAT_API_FLAGS	Bad flags were passed.
CAPI_STAT_API_HANDLE	There was a problem with a handle.
CAPI_STAT_API_HANDLE_BAD	The passed handle was corrupt.
CAPI_STAT_API_HANDLE_NOTNULL	The passed handle was not null.
CAPI_STAT_API_HANDLE_NULL	The passed handle was null.
CAPI_STAT_API_INCOMPLETE_TRANSACTION	There was a problem with the transaction (probably a store operation) and the data has been stored partially, so the data on the server might be inconsistent with the data the user was trying to store.
CAPI_STAT_API_NULL	A null pointer was passed.
CAPI_STAT_API_POOL	There was a problem with the connection pool.

Table 6–1 (Cont.) Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_API_POOL_LOCKFAILED	The connection pool couldn't obtain a lock.
CAPI_STAT_API_POOL_NOCONNECTIONS	The connection pool has no appropriate connections available.
CAPI_STAT_API_POOL_NOTINITIALIZED	The connection pool was not initialized.
CAPI_STAT_API_SESSION	There was a problem with a session.
CAPI_STAT_API_SESSION_BAD	The passed session was corrupt.
CAPI_STAT_API_SESSION_NOTNULL	The passed session was not null.
CAPI_STAT_API_SESSION_NULL	The passed session was null.
CAPI_STAT_API_STREAM	There was a problem with a stream.
CAPI_STAT_API_STREAM_BAD	The passed stream was corrupt.
CAPI_STAT_API_STREAM_NOTNULL	The passed stream was not null.
CAPI_STAT_API_STREAM_NULL	The passed stream was null.
CAPI_STAT_CONFIG	Configuration class status.
CAPI_STAT_CONFIG_CANNOT_OPEN	Failed to open the configuration file passed to CSDK_ SetConfigFile.
CAPI_STAT_CONFIG_CANNOT_OPEN_ TMPDIRECTORYPATH	Failed to open the tmpDirectoyPath specified in the configuration.
CAPI_STAT_DATA	Data class status.
CAPI_STAT_DATA_COOKIE	Information about the supplied cookie.
CAPI_STAT_DATA_DATE	Information about a date.
CAPI_STAT_DATA_DATE_FORMAT	The format of the date data is incorrect.
CAPI_STAT_DATA_DATE_INVALID	A specified date is invalid (for example, February 30th)
CAPI_STAT_DATA_DATE_NOT_LOCAL	A UTC DTSTART (and/or DTEND) was specified for a recurring event. Must be floating or local.
CAPI_STAT_DATA_DATE_OUTOFRANGE	A specified date is out of the range supported by this implementation.
CAPI_STAT_DATA_DATE_RANGE	The date range is incorrect.

Table 6–1 (Cont.) Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_DATA_EMAIL	Information about email.
CAPI_STAT_DATA_EMAIL_NOTSET	No email address is set on the server for one or more users/resources.
CAPI_STAT_DATA_ENCODING	Information about the encoding of supplied data.
CAPI_STAT_DATA_EVENTTYPE	Information about data for the event type is incorrect.
CAPI_STAT_DATA_HOSTNAME	Information about a hostname.
CAPI_STAT_DATA_HOSTNAME_FORMAT	The format of the hostname string was wrong.
CAPI_STAT_DATA_HOSTNAME_HOST	The hostname string could not be resolved to a host.
CAPI_STAT_DATA_HOSTNAME_SERVER	No server could be found on the specified host and port.
CAPI_STAT_DATA_ICAL	Information about iCalendar data.
CAPI_STAT_DATA_ICAL_ATTACH	The operation encountered an error only with the attachment.
CAPI_STAT_DATA_ICAL_ATTACH_INVALID_FILE_URI	The attachment file URI was invalid.
CAPI_STAT_DATA_ICAL_CANTMODIFYRRULE	An attempt was made to modify the recurrence rule for a calendar entry. This is not supported.
CAPI_STAT_DATA_ICAL_COMPEXTRA	An extra component was encountered. Either multiple specifications of a component that should only appear once, or a component that should not appear
CAPI_STAT_DATA_ICAL_COMPMISSING	An expected or required component was missing.
CAPI_STAT_DATA_ICAL_COMPNAME	There was a problem with a component name.
CAPI_STAT_DATA_ICAL_COMPVALUE	There was a problem with what a component contained.
CAPI_STAT_DATA_ICAL_FOLDING	There was a problem in the line folding.
CAPI_STAT_DATA_ICAL_IMPLEMENT	A problem with this particular iCalendar implementation.
CAPI_STAT_DATA_ICAL_INVALIDEXTENSIONDATA	An Oracle iCalendar extension was used incorrectly.
CAPI_STAT_DATA_ICAL_LINEOVERFLOW	One of the iCal data lines was too long, breaching the iCalendar specification (RFC 2445).
CAPI_STAT_DATA_ICAL_MISSING_DTSTART	The iCalendar data given did not contain a DTSTART property required to perform the requested operation.

Table 6–1 (Cont.) Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_DATA_ICAL_MISSING_UID	One or more VEVENTs were missing UID properties and an attempt was made to do something other than a CREATE.
CAPI_STAT_DATA_ICAL_MISSINGRECURID	Multiple VEVENTs were supplied with the same UID, and at least two of them did not have a RECURRENCE-ID property.
CAPI_STAT_DATA_ICAL_NOATTENDEES	An attempt was made to store a calendar event without any ATTENDEE properties. This is only supported with the flag CSDK_FLAG_STORE_INVITE_SELF
CAPI_STAT_DATA_ICAL_NONE	The provided data was not iCalendar data.
CAPI_STAT_DATA_ICAL_NOTANATTENDEE	An attempt was made to reply to a calendar event which the user is not attending.
CAPI_STAT_DATA_ICAL_OVERFLOW	There was an overflow when parsing the iCalendar data. This is caused by an internal limitation of the iCalendar library, and not by a breach of the spec
CAPI_STAT_DATA_ICAL_PARAMEXTRA	An extra parameter was encountered. Either multiple specifications of a parameter which should only appear once, or a parameter which should not appear
CAPI_STAT_DATA_ICAL_PARAMMISSING	An expected or required parameter was missing.
CAPI_STAT_DATA_ICAL_PARAMNAME	There was a problem with a parameter name.
CAPI_STAT_DATA_ICAL_PARAMVALUE	There was a problem with a parameter value.
CAPI_STAT_DATA_ICAL_PROPEXTRA	An extra property was encountered. Either multiple specifications of a property which should only appear once, or a property which should not appear
CAPI_STAT_DATA_ICAL_PROPMISSING	An expected or required property was missing.
CAPI_STAT_DATA_ICAL_PROPNAME	There was a problem with a property name.
CAPI_STAT_DATA_ICAL_PROPVALUE	There was a problem with a property value.
CAPI_STAT_DATA_ICAL_RECURMODE	There was a problem with the recurrence specification. The rules laid out in the description of CAPI_StoreEvent were breached
CAPI_STAT_DATA_ICAL_UNKNOWNDATA	The iCalendar data contained unrecognized iCalendar tokens.
CAPI_STAT_DATA_MIME	Information about MIME data.
CAPI_STAT_DATA_MIME_CHARSET	An unsupported character set was specified in a MIME header.
CAPI_STAT_DATA_MIME_COMMENT	A comment could not be parsed.

Table 6–1 (Cont.) Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_DATA_MIME_ENCODING	The encoding specified in the MIME object is not supported.
CAPI_STAT_DATA_MIME_HEADER	A header could not be parsed.
CAPI_STAT_DATA_MIME_IMPLEMENT	A restriction specific to this MIME implementation was breached.
CAPI_STAT_DATA_MIME_IMPLEMENT_NESTING	The MIME object was nested too deeply.
CAPI_STAT_DATA_MIME_LENGTH	One of the header lines was too long.
CAPI_STAT_DATA_MIME_NOICAL	No MIME parts were found whose headers indicated that they contain iCalendar data.
CAPI_STAT_DATA_MIME_NONE	No MIME data was found.
CAPI_STAT_DATA_MIME_OVERFLOW	An overflow occurred while reading MIME data.
CAPI_STAT_DATA_QUERY	Information about queries.
CAPI_STAT_DATA_QUERY_CONDITION_ILLEGAL_OPERATOR	A CSDKCondition contained an operator which was not legal for the query being performed.
CAPI_STAT_DATA_QUERY_CONDITION_NULL	A NULL CSDKCondition object was passed into a API function.
CAPI_STAT_DATA_QUERY_CONDITION_PROPERTY_NULL	A CSDKCondition contained a NULL property.
CAPI_STAT_DATA_QUERY_CONDITION_PROPERTY_TOO_LONG	A CSDKCondition contained a property name which was longer than expected.
CAPI_STAT_DATA_QUERY_CONDITION_UNKNOWN_OPERATOR	A CSDKCondition contained an unknown operator.
CAPI_STAT_DATA_QUERY_CONDITION_VALUE_NULL	A CSDKCondition contained a NULL value.
CAPI_STAT_DATA_QUERY_CONDITION_VALUE_TOO_LONG	A CSDKCondition contained a property name which was longer than expected.
CAPI_STAT_DATA_QUERY_ILLEGAL_OPERATOR	An illegal operator was passed to CSDK_AddConditionToQuery. (Only CSDK_LOP_OR and CSDK_LOP_AND are supported.)
CAPI_STAT_DATA_QUERY_NULL	A NULL CSDKQuery object was passed into a API function.
CAPI_STAT_DATA_RRESULT	Information about a request result.
CAPI_STAT_DATA_RRESULT_EOR	No more results were found in the CSDKRequestResult.

Table 6–1 (Cont.) Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_DATA_SERVER	Information about data on the server.
CAPI_STAT_DATA_SERVER_TOOMANYATTENDEES	The event has too many attendees for the server to handle.
CAPI_STAT_DATA_SERVER_TOOMANYINSTANCES	The event has too many recurrences for the server to handle.
CAPI_STAT_DATA_UID	Information about a UID.
CAPI_STAT_DATA_UID_ALREADYEXISTS	An object with the specified UID already exists.
CAPI_STAT_DATA_UID_FORMAT	The format of the UID string was wrong.
CAPI_STAT_DATA_UID_MULTIPLEMATCHES	Multiple objects with the specified UID exist.
CAPI_STAT_DATA_UID_NOTFOUND	Data with the supplied UID could not be found.
CAPI_STAT_DATA_UID_RECURRENCE	The specified object could not be found.
CAPI_STAT_DATA_USERID	Information about a userid.
CAPI_STAT_DATA_USERID_EXT	There was a problem with the Extended part of the UserId string.
CAPI_STAT_DATA_USERID_EXT_CONFLICT	Either userid AND X.400 were specified, or both a node and a calendar domain were specified.
CAPI_STAT_DATA_USERID_EXT_FORMAT	The format of the extended string was bad.
CAPI_STAT_DATA_USERID_EXT_INIFILE	There was a problem with the inifile.
CAPI_STAT_DATA_USERID_EXT_MANY	Multiple users were identified by the string.
CAPI_STAT_DATA_USERID_EXT_NODE	The specified node could not be found.
CAPI_STAT_DATA_USERID_EXT_NONE	No users were identified by the string.
CAPI_STAT_DATA_USERID_FORMAT	The format of the UserId string was wrong.
CAPI_STAT_DATA_USERID_ID	There was a problem with the Id part of the UserId string.
CAPI_STAT_DATA_VCARD	Information about vCard data.
CAPI_STAT_DATA_VCARD_COMPNAME	There was a problem with a component name.

Table 6–1 (Cont.) Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_DATA_VCARD_DUPERROR	An attempt was made to store a vCard using the mode CSDK_FLAG_STORE_CREATE but a contact with the same UID already existed on the server.
CAPI_STAT_DATA_VCARD_PARAMEXTRA	An extra parameter was encountered. Either multiple specifications of a parameter which should only appear once, or a parameter which should not appear
CAPI_STAT_DATA_VCARD_PARAMMISSING	An expected or required parameter was missing.
CAPI_STAT_DATA_VCARD_PARAMNAME	There was a problem with a parameter name.
CAPI_STAT_DATA_VCARD_PARAMVALUE	There was a problem with a parameter value.
CAPI_STAT_DATA_VCARD_PROPEXTRA	An extra property was encountered. Either multiple specifications of a property which should only appear once, or a property which should not appear
CAPI_STAT_DATA_VCARD_PROPMISSING	An expected or required property was missing.
CAPI_STAT_DATA_VCARD_PROPNAME	There was a problem with a property name.
CAPI_STAT_DATA_VCARD_PROPVALUE	There was a problem with a property value.
CAPI_STAT_DATA_VCARD_PROPVALUE_VIOLATED_SERVERRULE	There was a problem with a property which does not respect a server rule. Example: Suppose dayStart = 9 a.m. and we try to store dayEnd = 8 a.m. This is an error since dayEnd < dayStart.
CAPI_STAT_DATA_VCARD_VERSION_UNSUPPORTED	A vCard with an unrecognized VERSION was encountered.
CAPI_STAT_LIBRARY	Library class status.
CAPI_STAT_LIBRARY_IMPLEMENTATION	The feature is not fully implemented.
CAPI_STAT_LIBRARY_INTERNAL	An internal error occurred in the library.
CAPI_STAT_LIBRARY_INTERNAL_CONTEXT	Invalid context for a dependant library.
CAPI_STAT_LIBRARY_INTERNAL_COSMICRAY	Something completely unexpected happened internally.
CAPI_STAT_LIBRARY_INTERNAL_DATA	There was a corruption of data in the library.
CAPI_STAT_LIBRARY_INTERNAL_EXPIRY	The function has expired in this library.
CAPI_STAT_LIBRARY_INTERNAL_FUNCTION	The library miscalled a function.
CAPI_STAT_LIBRARY_INTERNAL_OVERFLOW	Some internal maximum was exceeded.

Table 6–1 (Cont.) Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_LIBRARY_INTERNAL_PROTOCOL	The library abused a protocol.
CAPI_STAT_LIBRARY_INTERNAL_UNKNOWN_ EXCEPTION	CAPI received an unknown C++ exception.
CAPI_STAT_LIBRARY_INTERNAL_UNKNOWN_LIBRARY_ ERRCODE	Failed to map an error code from a dependant library.
CAPI_STAT_LIBRARY_SERVER	A limitation of or occurrence on the server.
CAPI_STAT_LIBRARY_SERVER_BUSY	The server cannot service the request right now because it is busy.
CAPI_STAT_LIBRARY_SERVER_SUPPORT	The server does not provide support.
CAPI_STAT_LIBRARY_SERVER_SUPPORT_CHARSET	There is no support for the required character set.
CAPI_STAT_LIBRARY_SERVER_SUPPORT_STANDARDS	There is no support for CAPI on this server.
CAPI_STAT_LIBRARY_SERVER_SUPPORT_UID	There is no support for storing UIDs.
CAPI_STAT_LIBRARY_SERVER_TIMEZONE	There was an error dealing with timezones from the Oracle Calendar server.
CAPI_STAT_LIBRARY_SERVER_UNAVAILABLE	The server is running, but unavailable for some reason, e.g. the desired node is down for maintenance
CAPI_STAT_LIBRARY_SERVER_USERDATA	There is some problem with user data on the server.
CAPI_STAT_OK	Operation completed successfully. Value 0
CAPI_STAT_SECUR	Security class status.
CAPI_STAT_SECUR_CANTBOOKATTENDEE	One or more attendees could not be booked. This could be due to lack of access rights, or because the attendee is already booked.
CAPI_STAT_SECUR_INSUFFICIENTRIGHTS	User doesn't have sufficient rights to perform the operation.
CAPI_STAT_SECUR_LOGON	There was a security error on logon.
CAPI_STAT_SECUR_LOGON_AUTH	Logon authentication failed.
CAPI_STAT_SECUR_LOGON_LOCKED	The specified account is locked.
CAPI_STAT_SECUR_LOGON_LOCKED_RESOURCE	Logon is locked for resources.
CAPI_STAT_SECUR_LOGON_LOCKED_SYSOP	Logon is locked for Sysops.

Table 6–1 (Cont.) Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_SECUR_READ	There was a security error on read.
CAPI_STAT_SECUR_READ_ALARM	There was a security error reading alarm data.
CAPI_STAT_SECUR_READ_PROPS	There was a security error reading properties.
CAPI_STAT_SECUR_SERVER	There was a security error in the server.
CAPI_STAT_SECUR_SERVER_LICENSE	There was a licensing error on the server.
CAPI_STAT_SECUR_SERVER_SET_IDENTITY_SYSOP	The server requires a SetIdentity call on the sysop logon to perform the operation.
CAPI_STAT_SECUR_SERVER_SET_IDENTITY_SYSOP_ REMOTE	Cannot set identity as a remote user while logged in as node sysop.
CAPI_STAT_SECUR_WRITE	There was a security error on write.
CAPI_STAT_SECUR_WRITE_AGENDA	There was a security error writing to an agenda.
CAPI_STAT_SECUR_WRITE_EVENT	There was a security error writing to an event.
CAPI_STAT_SERVICE	Service class status.
CAPI_STAT_SERVICE_ACE	There was a problem caused by one of the ACE plug-ins.
CAPI_STAT_SERVICE_ACE_LOAD	Required ACE plug-in could not be loaded.
CAPI_STAT_SERVICE_ACE_SUPPORT	Requested ACE option not supported.
CAPI_STAT_SERVICE_FILE	There was a problem with system file services.
CAPI_STAT_SERVICE_FILE_CLOSE	There was a problem closing a file.
CAPI_STAT_SERVICE_FILE_DELETE	There was a problem deleting a file.
CAPI_STAT_SERVICE_FILE_MODE	There was a problem with the read or write mode for a file.
CAPI_STAT_SERVICE_FILE_OPEN	There was a problem opening a file.
CAPI_STAT_SERVICE_FILE_READ	There was a problem reading from a file.
CAPI_STAT_SERVICE_FILE_TEMP	There was a problem allocating a temporary file.
CAPI_STAT_SERVICE_FILE_WRITE	There was a problem writing to a file.

Table 6–1 (Cont.) Oracle Calendar SDK Status Codes

Status Code	Description
CAPI_STAT_SERVICE_LIBRARY	There was a problem with the standard library services.
CAPI_STAT_SERVICE_MEM	There was a problem with system memory services.
CAPI_STAT_SERVICE_MEM_ALLOC	Could not allocate memory.
CAPI_STAT_SERVICE_NET	There was a problem with network services.
CAPI_STAT_SERVICE_NET_TIMEOUT	Timeout while waiting for network services.
CAPI_STAT_SERVICE_THREAD	There was a problem with system thread services.
CAPI_STAT_SERVICE_TIME	There was a problem with the standard time services.
CAPI_STAT_SERVICE_TIME_GMTIME	GMTime could not be obtained.

7

Oracle Calendar SDK Configuration Settings

This chapter contains detailed information on Oracle Calendar SDK configuration settings and Oracle Calendar server parameters that affect applications that use the Oracle Calendar SDK.

Calendar SDK Configuration Settings

These settings may be placed in a text file, the name of which must be passed to the function CSDK_SetConfigFile. The structure of the file is:

```
[<section>]
<keyword>=<value>
...
```

The following is a sample configuration file suitable for debugging:

Example 7–1 calendar_config.ini

```
[SDK]
client_name = My Sample Client
client_version = 10.1.2

[LOG]
log_trace = true
log_debug = true
log_activity = true
log_state = false
log_modulesinclude = {CAPI, VATTR, SAPPI, ICAL}
```

This configuration file configures the Calendar SDK to do the following:

- Activate the logging mechanism
- Log the activity of the Calendar C API module

The following tables describe the valid configuration settings for each section.

Table 7–1 SDK Section

Setting	Values	Default Value	Description
client_name	Any string	""	Sets the application name that be visible in the server stats.
client_version	Any string	""	Sets the application version that be visible in the server stats.

Table 7–1 (Cont.) SDK Section

Setting	Values	Default Value	Description

tmpDirectoryPath	String (a valid path to a directory on the local filesystem)	""	Specifies a path to a directory on the local filesystem that will be used by the SDK to store temporary files. It is the user's responsibility to empty that directory. Currently, that directory is only used for attachments. Attachment files from meetings will be stored in subdirectories of the form `./<UID>-<RECURRENCE-ID>/<filename>` of that temporary directory.
			The vale of this setting should not be in quotation marks. For example, the following is valid in the configuration file:
			`tmpDirectoryPath = /tmp`
			However, the following might not be valid on certain platforms:
			`tmpDirectoryPath = "/tmp"`

Table 7–2 CACHE Section

Setting	Values	Default Value	Description
cncachesize	[0..U32MAX]	512	Sets the maximum number of entries to hold in common name cache.
direntrycachesize	[0..U32MAX]	512	Sets the maximum number of entries to hold in the directory entry cache.
emailcachesize	[0..U32MAX]	512	Sets the maximum number of entries to hold in the email address cache.
itemcachesize	[0..U32MAX]	512	Sets the maximum number of entries to hold in the item cache.
securitycachesize	[0..U32MAX]	512	Sets the maximum number of entries to hold in the security record cache.
tzcachesize	[0..U32MAX]	256	Sets the maximum number of entries to hold in the timezone record cache.

Table 7–3 STATS Section

Setting	Values	Default Value	Description
apitime	true/false	false	When enabled, logs transaction times for each API function call.

Table 7–3 (Cont.) STATS Section

Setting	Values	Default Value	Description
server	true/false	false	When enabled, logs server stats for each API function call.

Table 7–4 LOG Section

Setting	Values	Default Value	Description
log_activity	true/false	false	Enables "activity" (high-level) logging.
log_debug	true/false	false	Enables "debug" logging.
log_filename	filename	No default value	Specifies the file to which logging information will be written.
log_modulenames	true/false	false	Controls whether module names are printed with each message.
log_modulesinclude	"" or "{CAPI, VATTR, SAPPI, ICAL}", each of the values in the braces are optional.	""	Controls which modules have logging enabled. This should be set, at minimum, to "{CAPI}", otherwise no logging will be performed even if it is enabled (by setting log_activity = true).
log_state	true/false	false	Enables "state" logging.
log_trace	true/false	false	Enables "trace" logging. **Warning**: This setting can result in a large amount of logging, as well as a significant performance hit, depending on the modules included.

The following settings may be placed in the following sections:

- [CONNPOOL]
- [CONNPOOL:alias]

If a setting is specified in [CONNPOOL], it sets the value to be used with any host not specifically configured in the connection pool. If a setting is specified in [CONNPOOL:*alias*], it will only apply to the host specified by *alias*.

Table 7–5 CONNPOOL Section

Setting	Values	Default Value	Description
blocking	true/false	true	Indicates whether a connection request will block or return an error if there are no available connections.

Table 7–5 (Cont.) CONNPOOL Section

Setting	Values	Default Value	Description
connect_on_startup	true/false	true	Indicates whether the pool will open the minimum number of connections immediately on startup, or instead wait for connections to be requested before opening them.
host	host,node	""	Sets the host and node for a connection pool entry. This keyword may only be placed inside the section for a specific host and node (a section named [CONNPOOL:*alias*].
max_caldomain	[0..S32MAX]	0	Sets the maximum number of caldomain connections for the given server name/node ID in the connection pool.
max_masternode	[0..S32MAX]	0	Sets the maximum number of masternode connections for the given server name/node ID in the connection pool.
max_sysop	[0..S32MAX]	0	Sets the maximum number of SYSOP connections for the given server name/node ID in the connection pool.
max_user	[0..S32MAX]	none, value required to use connection pooling	Sets the maximum number of user connections for the given server name/node ID in the connection pool.
min_caldomain	[0..?]	0	Sets the minimum number of caldomain connections for the given server name/node ID in the connection pool.
min_masternode	[0..?]	0	Sets the minimum number of masternode connections for the given server name/node ID in the connection pool.
min_sysop	[0..?]	0	Sets the minimum number of SYSOP connections for the given server name/node ID in the connection pool.
min_user	[0..?]	0	Sets the minimum number of user connections for the given server name/node ID in the connection pool.

Calendar Server Parameters

The following table describes Oracle Calendar server parameters that affect applications that use the Oracle Calendar SDK:

Table 7–6 ENG Section

Setting	Values	Default Value	Description
allowsysoplogon_capi	true/false	false	If Oracle Calendar SDK users want to log in as Sysop, they need to set this setting to "true" for *each* host or installation they wish to log in.
sessionexpiry_csdk	Any positive integer or 0	0	Specifies the amount of time, in minutes, before the connection between the Oracle Calendar SDK and the Oracle Calendar server is terminated. The actual expiry is set within plus or minus 30% of this value. This implies that with a setting of 2160 minutes (36 hours), the actual expiry will be within one or two days of the connection being established. A value of 0 indicates that no limit should be enforced by the server.

8

Oracle Calendar SDK FAQ and Troubleshooting

This chapter contains frequently asked questions and troubleshooting information for the Oracle Calendar SDK.

Frequently Asked Questions

Will my applications written with the CAPI functions from older Oracle Calendar SDK work with Oracle Calendar server Release 2 (9.0.4) and up?

Yes, your older applications will continue to work and you can continue to write applications using the CAPI functions of Release 2 (9.0.4) of the Oracle Calendar SDK.

However, these older CAPI functions store and retrieve some iCalendar properties on the Oracle Calendar server in a format only understood by these CAPI functions. So clients like Oracle Connector for Outlook or applications written with CSDK functions of the Oracle Calendar SDK will not see these properties. Since Release 2 (9.0.4) of Oracle Collaboration Suite, the Oracle Calendar SDK has introduced the CSDK functions that completely respect the Oracle Calendar data representation to eliminate this incompatibility. Oracle encourages you to migrate your Oracle Calendar SDK applications to use the CSDK functions.

The following iCalendar properties are the ones that are incompatible:

- COMMENT
- CONTACT
- RELATED-TO
- SEQUENCE
- UID
- URL
- Any X-ORACLE properties

Likewise, applications using the older CAPI functions will not obtain the values set for those properties by applications using CSDK functions of the Oracle Calendar SDK or clients like Oracle Connector for Outlook.

Can I write an Oracle Calendar SDK program using Visual Basic or other programming languages?

The Oracle Calendar SDK is a package of C/C++ function calls, so any language that can natively support C can be used to create a wrapper and access these functions.

Options include using Visual Basic, Perl, Java, and Python. There may be other independent efforts in existence; the Oracle Calendar SDK forum on Oracle OTN is a good place to look for such information.

What is the minimal information required to create a new event?

You need the following properties to create an event:

- ATTENDEE
- DTSTART
- DTEND or DURATION

However, if you use the CSDK_StoreEvents function with the CSDK_FLAG_STORE_INVITE_SELF flag, you do not have to include an ATTENDEE property.

How do I uninvite someone from an event?

Replace the event by calling CSDK_StoreEvents and supplying an iCalendar without an ATTENDEE property for the user.

How do I invite someone to an event?

Provide an ATTENDEE property for the user when calling CSDK_StoreEvents.

Can I log in as the Oracle Calendar server administrator (Oracle Calendar SYSOP) using the Oracle Calendar SDK?

Yes. Oracle Calendar SYSOP login is supported as of release 9.0.4 of the Oracle Calendar SDK.

How do I delete an event completely?

Use the function CSDK_DeleteEvents, supplying the event UID.

What is the default access level of an event or task?

If you do not specify the CLASS property in an event or task, that event or task will have an access level of PUBLIC. This is the behavior as defined in RFC 2445.

Can I fetch ATTENDEE properties for all attendees (resources and users) of an event, even if some of those attendees do not have an email address?

Yes. Use one of the following flags with the functions CSDK_FetchEventsByRange, CSDK_FetchEventsByAlarmRange, or CSDK_FetchEventsByUID:

- CSDK_FLAG_FETCH_RESOURCES_WITHOUT_ADDRESSES: The Oracle Calendar SDK will generate an ATTENDEE property for any resource without an email address. It will create a mailto URI of the form "mailto:<*guid*>@email.invalid" (see RTC 2606 for more information about reserved invalid domain names).
- CSDK_FLAG_FETCH_USERS_WITHOUT_ADDRESSES: The Oracle Calendar SDK will generate an ATTENDEE property for any normal user without an email address. It will create a mailto URI of the form "mailto:<*guid*>@email.invalid".

How do I modify the start time (DTSTART) of an event?

To modify the start time of an event instance (it is irrelevant whether the event is the only instance or has multiple instances), set the RECURRENCE-ID property to the *current* start time of the instance as it exists on the Oracle Calendar server, and set DTSTART to the new start time. In this way, the Oracle Calendar SDK can determine precisely which instance to reschedule. If you set only the DTSTART to a time different

than the current start time, the Oracle Calendar SDK will attempt to add a new instance to your event at the new start time.

For example, suppose the following event is stored on the Oracle Calender server:

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID://Oracle//CSDK//EN
BEGIN:VEVENT
UID:event_we_want_to_modify-oracle
DTSTART:20050101T120000Z
DTEND: 20050101T130000Z
END:VEVENT
END:VCALENDAR
```

To modify this event's DTSTART from January 1 to February 1, store the following event with the CSDK_StoreEvent function and the flag CSDK_FLAG_STORE_MODIFY:

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID://Oracle//CSDK//EN
BEGIN:VEVENT
RECURRENCE-ID:20050101T120000Z
UID:event_we_want_to_modify-oracle
DTSTART:20050201T120000Z
DTEND:20050201T130000Z
END:VEVENT
END:VCALENDAR
```

How do I accept a meeting?

In order to accept a meeting, or set the reply status of an event not owned by the logged-in user, use the CSDK_FLAG_STORE_REPLY with the CSDK_StoreEvent function.

Do not use the CSDK_FLAT_STORE_MODIFY flag. For example, Abe (as the logged in user) creates an event with UID:abe@example.com. Abe invites Bea through the ATTENDEE property. If Bea (as the logged in user) tries to update her status on that event (with UID:abe@example.com) with the CSDK_FLAG_STORE_MODIFY flag, she will receive a CAPI_STAT_SECUR_WRITE error.

My calendar users are spread across different time zones. How can I accurately retrieve one day's events for a particular user?

Day events and daily notes have start times which are dependent on the time zone in which they were created. Because of this, to correctly retrieve all of them for a particular day, the time range needs to be extended up to twelve hours at both ends of the time range. This can be done separately from fetching regular meetings (using the exclusion flags), or you can fetch them all at once and manually filter out regular meetings that fall outside the desired time range.

How can I add an attendee to all instances of an event?

To add an attendee to all instances of an event, use the MODIFY flag to store an iCalendar object containing a VEVENT with only the UID of the event and the ATTENDEE property you wish to add.

If the event has any exceptions (as represented by separate VEVENTs that contain the RECURRENCE-ID property when fetched from the Oracle Calendar server), these need to be handled individually by also providing a VEVENT with the UID and

corresponding RECURRENCE-ID of the instance, along with the ATTENDEE property to be added. Modifications to the series and the exceptions can be done in the same event store operation by including all the relevant VEVENTs in the same VCALENDAR object.

Troubleshooting

When I run the Oracle Calendar SDK demos, I get the error "libcapi.so not found, no such file or directory", or "Cannot load library libcapi". How do I fix this?

See the latest Oracle Calendar SDK Readme file for this information.

I'm getting error 2148073984 from calling an Oracle Calendar SDK function. What does that mean? Where can I find more information on Oracle Calendar SDK errors?

Chapter 6, "Oracle Calendar SDK Status Codes" contains a list of Oracle Calendar SDK status codes. Each code can be divided into five fields, each describing a different level of the problem.

The two helper functions, CSDK_GetStatusLevels() and CSDK_GetStatusString(), can help decode the error easily. You can also look at the demo applications that ship with the Oracle Calendar SDK for an example of how to decode error codes.

Why do my accented "é" characters appear as =C3=A9 in events I retrieve using the Oracle Calendar SDK?

The Oracle Calendar SDK encodes its output in UTF-8, in which the character is represented as 2 bytes: 0xC3 and 0xA9. When returning MIME-encapsulated data, the Oracle Calendar SDK further encodes those bytes in quoted-printable strings, which results in =C3=A9.

There seem to be a lot of extra =3D characters in the ATTENDEE property. Did I just discover a bug?

No, this is not a bug. `=3D` is really an equal sign (=) encoded in quoted-printable. So, where you might expect to see `partstat=confirmed`, you would actually see `partstat=3Dconfirmed`.

Why do I get a LNK1106 error when compiling the Oracle Calendar SDK with Microsoft Visual C++?

The Oracle Calendar SDK was compiled with Visual C++ 6; older versions need a patch to work. Search the Microsoft Web site for available patches.

Ensuring that you are using Visual C++ will mitigate linking errors you may encounter with other C compilers. Refer to the demo applications project for recommended compiler and linker settings.

Why is my program aborting whenever I use CSDK_CreateFileStream()? I tried the other type of streams and they worked fine.

You need to link your application using Microsoft's C runtime library (the /MD switch if you're using the command line) as opposed to the static C libraries. This is required because the FILE pointer used by the file stream has different definitions depending on which version of the C library is being used.

CSDK_CreateFileStreamFromFilenames can be used, in which case the FILE pointer will not need to be passed between the SDK and your application.

Why does CSDK_GetHandle() tell me it can't find my resource?

There are a couple of possible answers:

- Your resource name may contain accented or special characters such as "‡". The Oracle Calendar SDK expects user and resource identification strings to be in UTF-8. Be sure to use the proper UTF-8 values to describe accented or special characters.
- Your resource name contains a forward-slash (/) character, such as `"Training / Meeting Room"`. The use of this character inside resource names conflicts with the forward-slash used by default as the field delimiter. To fix this, simply use another character as your delimiter. For example, RS=Work/.

When I create a meeting, the STATUS says "CONFIRMED". Why does it say "Will confirm later" when I look at the meeting with the native client?

"Will confirm later" is the participation status of the attendee, while "CONFIRMED" is the STATUS property of the VEVENT. The Oracle Calendar SDK sets the STATUS property to "CONFIRMED" by default. You may set the STATUS property to other values including "TENTATIVE" and "CANCELLED".

The user's attendance status is stored in the PARTSTAT parameter of the ATTENDEE property corresponding to that user.

Why am I getting CAPI_STAT_DATA_MIME_HEADER errors?

By default, iCalendar objects passed to the Oracle Calendar SDK must have a MIME header. Otherwise, the Oracle Calendar SDK would return the error CAPI_STAT_DATA_MIME_HEADER otherwise.

Alternately, you can specify the CSDK_FLAG_STREAM_NOMIME flag, which is much simpler.

For example, the Oracle Calendar SDK considers the following MIME encapsulated iCalendar as valid:

```
String my_ical = "MIME-Version: 1.0\n" +
  "Content-Type: text/calendar\n" +
  "Content-Transfer-Encoding: quoted-printable\n\n" +
  "BEGIN:VCALENDAR\n" +
  "VERSION:2.0\n" +
  "PRODID:-//ORACLE//NONSGML CSDK 9.0.4.1 - Java SDK Demo 9.0.4.1//EN\n" +
  "BEGIN:VEVENT\n" +
  "DTSTART:20040210T183000Z\n" +
  "DTEND:  20040210T193000Z\n" +
  "SUMMARY:Asker Demo\n" +
  "LOCATION:office\n" +
  "ATTENDEE:MAILTO:john@example.com\n" +
  "ATTENDEE:MAILTO:jane@example.com\n" +
  "END:VEVENT\n" +
  "END:VCALENDAR\n";
```

Why am I getting CAPI_STAT_DATA_ICAL_NOATTENDEES errors?

In order to add an event to a user's agenda, that event must have an ATTENDEE property. Otherwise, the Oracle Calendar SDK would return the error CAPI_STAT_DATA_ICAL_NOATTENDEES.

Alternatively, the caller of the function (such as CSDK_StoreEvents) can specify the CSDK_FLAG_STORE_INVITE_SELF flag, which implicitly adds the caller as an attendee.

Part II

Oracle Calendar Web Services Toolkit

This part of the Oracle Calendar Application Developer's Guide describes the Oracle Calendar Web services toolkit.

This part contains the following chapters:

9

Oracle Calendar Web Services Toolkit Overview

This chapter provides an overview of Oracle Calendar Web services and the Web services toolkit.

Related documents:

- Internet Calendaring and Scheduling Core Object Specification (iCalendar) http://www.ietf.org/rfc/rfc2445.txt
- Web Services Activity http://www.w3.org/2002/ws/
- Web Services Description Language (WSDL) http://www.w3.org/TR/wsdl
- Simple Object Access Protocol (SOAP) 1.1 http://www.w3.org/TR/SOAP/
- *Oracle Calendar Web Services Java API Reference*

About Web Services

Oracle Calendar Web services is a component of the Oracle Calendar application system, which handles application-level services. Web services allows applications to manage, through common XML queries, calendar data from any portal, client application, or backend server. iCal data is coded in XML, wherein iCal becomes xCal. SOAP is used to encapsulate the messages for delivery. The calendar data Web services SOAP is stored directly on the Oracle Calendar server store. This is in effect the CWSL, or Calendar Web services Language.

This current implementation does not provide any WSDL or UDDI support. However, future versions may provide the ability to publish WSDL to a UDDI registry.

The Web Services Toolkit

Developers can use the Oracle Calendar Web services toolkit to build Web services applications and create SOAP 1.1 queries. The toolkit contains the functionality to search, create, modify, and delete calendar events, as well as search tasks. It gives SOAP access to the Oracle Calendar server database through a series of Java classes, known as the Calendarlet. This allows developers to use a Java IDE, abstracting the XML structure required to build applications.

Use the Calendarlet to create your own clients and integrate calendar data into your own applications. To integrate calendar data within any portal, client application, or backend server, you need to be able to make an HTTP connection to the Web server hosting Web services, generate SOAP messages and parse the SOAP responses (using any technology that can send and receive HTTP strings), and make use of an existing

XML toolkit to generate outgoing and parse incoming HTTP strings with a SOAP client toolkit. The toolkit supports the use of HTTP proxies.

Toolkit Contents

The Oracle Calendar Web services toolkit includes:

- `Calendarlet.tar`: The Calendarlet JAR file.
- `Javadoc.tar`: The Javadoc HTML documentation for the Calendarlet.
- `Ws_testtool.tar`: The Java source for the Calendar Web services toolkit testing tool, including sample source code.

10

Oracle Calendar Web Services SOAP Commands

This chapter describes how the Oracle Calendar Web services toolkit uses Extended Markup Language (XML) and Simple Object Access Protocol (SOAP) to retrieve and store iCalendar objects.

SOAP Request and Reply Syntax

The following is the structure of a SOAP request:

```
HTTP header

<?xml version='1.0' encoding='UTF-8'?>
<!-- SOAP envelope -->
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <SOAP-ENV:Header>
      <!-- If Basic Authentication is used,
           it is placed here in the SOAP header -->
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
      <!-- The SOAP body contains the methods
           used to perform actions on the Calendar Server -->
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The following is the structure of a SOAP reply:

```
HTTP header

<?xml version="1.0" encoding="utf-8" ?>
<!-- SOAP envelope -->
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <soap:Body>
    <!-- The SOAP body contains the result of the SOAP command
         or a SOAP fault if the command was not successful -->
  </soap:Body>
</soap:Envelope>
```

HTTP Headers

The HTTP header for a proper SOAP v1.1 transaction must contain the following elements:

```
POST <uri> HTTP/1.1
Content-Type: text/xml; charset="UTF-8"
Content-Length: <char length>
SOAPAction: <urn>

...soap envelope...
```

The <uri> is typically the URI for the Oracle Calendar application system (typically, /ocas-bin/ocas.fcgi). This is used mainly by the Web Server (Oracle HTTP Server or Apache) to identify the application system, invoke its internal fcgi protocol module, and pass the request to Web services.

Within Oracle Collaboration Suite, to bypass Oracle Single Sign-On mechanism (SSO), a separate URL may be required (typically, /ocws-bin/ocas.fcgi).

The Content-Type charset identifier is optional. If it is not provided, UTF-8 is assumed. However, UTF-8 is the only charset encoding supported. All other charsets will result in an error.

The HTTP response for an error is a 500 status code (for Internal Server Error). This is returned if the actual SOAP envelope is corrupt (in other words, we cannot determine what the data coming in is) or if a SOAP level error occurs. Keep in mind that all application level errors are returned within a SOAP Fault, along with the 500 HTTP status code.

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/xml
Content-Length: <char length>

...Optional soap envelope...
```

If the SOAP envelope can be properly executed, the SOAP information is correct, and the application level function succeeds, the 200 status code (success) will be returned.

```
HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: <char length>

... Soap envelope...
```

Security and Authentication

This section describes:

- Security and Authentication Design
- Basic Authentication
- Proxy Authentication

Although data encryption is a very important security element, at the present time there are no plans to encrypt data within SOAP requests.

Design

Within the SOAP domain, there are many efforts underway to define and standardize the authentication, security, and encryption of SOAP messages. Groups such as W3C, IETF, OASIS, and WS-I are all working toward the same end. Unfortunately, at the time of development of Oracle Calendar Web services, no definitive specification had been approved. However, some general trends were respected when defining the features that Web services supports, including:

- HTTP SSL and Web-based certificates
- Simple authentication
- An application-specific authentication mechanism (for Oracle Collaboration Suite).

> **Note:** The proposed specification "WS-Security" outlines some authentication mechanisms; however this is a working draft that does not carry industry approval.

The adopted practice with all these mechanisms is to include the required information within a series of SOAP headers, with the exception of HTTP level functionality (that is, SSL and certificates).

```
<SOAP-ENV:Envelope>
     <SOAP-ENV:Header>
          ... some encryption, signature, and
              authentication info goes here ...
     </SOAP-ENV:Header>
     <SOAP-ENV:Body>
          ... a soap method goes here ...
     </SOAP-ENV:Body>
</SOAP-ENV:Envelope >
```

At the application layer, only plain text authentication is supported. The user's password must be provided in plaintext only (NOT a base-64 encoded string) at the beginning of each transaction.

Security is provided at the transport, protocol and application levels. At the HTTP layer, there are two options: Normal or SSL. This layer is handled completely at the Web server level (that is, Apache and Oracle HTTP Server), providing encrypted data between the HTTP client and HTTP server. The Calendar Application Server has no dependencies on this layer.

The SOAP client must support SSL; not all toolkits do.

Basic Authentication

The Web services Basic Authentication is implemented using the SOAP header.

The initial version requires a BasicAuth element in the header for each request. If the element is not present, a SOAP Fault is generated.

```
HTTP/1.1 200 OK
Content-Type: text/xml:charset="UTF-8"
Content-Length: <char length>

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
          soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
     <soap:Header>
          <auth:BasicChallenge
```

```
                    xmlns:auth=
                "http://www.soap-authentication.org/2002/01/">
                <Realm>Oracle Web Services</Realm>
            </auth:BasicChallenge>
        </soap:Header>
        <soap:Body>
            <Reply xmlns:cwsl=
                "http://www.oracle.com/WebServices/Calendaring/1.0/">
            </Reply>
        </soap:Body>
</soap:Envelope>
```

<Realm> is used to provide a hint to the client. This is a configurable parameter in the ocws.conf file.

```
[basicauth]
Realm=Oracle Web Services           # default
```

A typical SOAP session with Basic Authentication contains the user's credentials within the soap header of the first message.

```
POST <uri> HTTP/1.1
Content-Type: text/xml; charset="UTF-8"
Content-Length: <char length>
SOAPAction: <urn>

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
          soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
     <soap:Header>
          <auth:BasicAuth
               xmlns:auth="http://www.soap-authentication.org/2002/01/">
               <Name>myname</Name>
               <Password>mypassword</Password>
          </auth:BasicAuth>
     </soap:Header>
     <soap:Body>
          ...
     </soap:Body>
</soap:Envelope>
```

The user name must be the Calendar Server's User ID. X.400 login is not permitted. Also, the User ID and Password must be properly XML encoded.

If the Basic Authentication fails, a SOAP fault is returned, indicating the source of the problem.

```
HTTP/1.1 500 Internal Server Error
Content-Type: text/xml
Content-Length: <char length>

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
          soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
     <soap:Body>
          <soap:Fault>
               <faultcode>soap:Server::Data::CalConnection</faultcode>
               <faultstring>A security error occurred</faultstring>
               <detail>
                    <cwsl:Error xmlns:cwsl=
               "http://www.oracle.com/WebServices/Calendaring/1.0/">
                         <Code>0020-00-00-00000017</Code>
                         ...
```

```
                    </cwsl:Error>
                </detail>
            </soap:Fault>
        </soap:Body>
</soap:Envelope>
```

The BasicAuth mechanism is to be used mainly for development and testing purposes. Alone, the mechanism provides little security, due to the use of plain text passwords. If this mechanism is to be used in a production environment, an SLL Web configuration is highly recommended.

Proxy Authentication

Proxy authentication, or application-to-application authentication, allows any application developed with Calendar Web Services Toolkit to establish a trusted authenticated link to the Calendar Server. This application does not require the authentication of the end-user using it.

Requirements

The following components are required to enable proxy authentication:

- Calendar Web Services Toolkit (Calendarlet.jar)
- Calendar Web Services (OCAS)
- Calendar Server (Calserv)
- Oracle Internet Directory (OID)

You must configure your Collaboration Suite deployment so that the Calendar Server is connected to OID. This should have been done by default. Proxy authentication is designed to use OID security schemes.

You must have access to the following:

- OID administrator account
- LDAP tools (located in $ORACLE_HOME/ldap/bin)
- Oracle Calendar server administrator password

Configuring System for Proxy Authentication

The following steps describe how to configure OID and grant proxy privileges to your application.

Step 1 Create an entry for your application product in OID

Create the following entry in OID, where *MyApplicationProduct* is the name of your application product:

- cn=OracleContext
- cn=Products
- cn=*MyApplicationProduct*

To create this entry, create the following LDIF file named MyApplicationProduct.ldif:

```
dn: cn= MyApplicationProduct, cn=Products, cn=OracleContext
objectClass: orclContainer
objectClass: top
```

Enter the following command to add the entry defined in MyApplicationProduct.ldif to OID:

```
./ldapadd -h HOSTNAME.COM -p OIDPORT -D "cn=orcladmin" -w PASSWROD -f
./MyApplicationProduct.ldif
```

- `HOSTNAME.COM` is the OID server hostname
- `PASSWORD` is the password for the OID directory

  `OIDPORT` is the OID port

Step 2 Create an application entity for your application in OID

Create the following entry in OID, where *MyAppName* is the name of your application:

- cn=OracleContext
- cn=Products
- cn=*MyApplicationProduct*
- orclApplicationCommonName=*MyAppName*

To create this entry, create the following LDIF file named MyAppName.ldif:

```
dn: orclApplicationCommonName= MyAppName,
cn= MyApplicationProduct, cn=Products,
cn=OracleContext
objectClass: orclApplicationEntity
objectClass: top
orclApplicationCommonName: MyAppName
userpassword: test1
```

Enter the following command to add the entry defined in MyAppName.ldif to OID:

```
./ldapadd -h HOSTNAME.COM -p OIDPORT -D "cn=orcladmin" -w PASSWORD -f
./MyAppName.ldif
```

Step 3 Ensure the application entity entry is properly configured

Perform an LDAP search to search for the entry's distinguished name, which is as follows:

```
"orclApplicationCommonName= MyAppName,
cn= MyApplicationProduct, cn=Products,
cn=OracleContext"
```

To do this enter the following command:

```
./ldapsearch -h HOSTNAME.COM -p OIDPORT -D "cn=orcladmin" -w PASSWORD -b
"cn=MyApplicationProduct,cn=Products,cn=OracleContext"
"objectclass=orclApplicationEntity" "c"
```

Step 4 Grant proxy privileges to the new application entity

Enter the following command from the directory $ORACLE_HOME/ocal/bin. You will need the Oracle Calendar server admin password:

```
./unioidconf -grantproxyprivilege \
"orclApplicationCommonName= MyAppName,
cn=MyApplicationProduct, cn=Products,
cn=OracleContext"
```

This command will create the following entry in OID:

- dc=com
- dc=oracle
- dc=us
- cn=OracleContext
- cn=Products
- cn=Calendar
- cn=UserProxyPrivilege
- uniquemember: orclApplicationCommonName= MyAppName, cn= MyApplicationProduct, cn=Products, cn=OracleContext

Using Proxy Authentication

Once you have successfully configured OID and Oracle Calendar server, you enable proxy authentication following these steps in your Java application:

1. Replace the BasucAuth class with the ProxyAuth class.
2. Set the end user identity, proxy application name, and proxy application password that you have registered in OID.

Your Java code will look similar to the following:

```
ProxyAuth auth = new ProxyAuth();

auth.setApplicationName("orclApplicationCommonName=MyAppName,
cn=MyApplicationProduct, cn=Products, cn=OracleContext");
auth.setApplicationPassword("test1");
auth.setName(myUserId);
```

Your application will no longer need to pass the end-user's password to Calendar Web Services. From now on, it is your application's responsibility to authenticate the end-user.

Identifying Calendar Objects

SOAP queries make use of Universal Identifiers (UIDs) and Global Unique Identifiers (GUIDS). The Web services API is based around the ability to uniquely identify a Calendar store object, retrieve it, and store a reference for last use. In Web services, the data-independent property to use is:

```
x-oracle-data-guid
```

This Data GUID maps to various data type specific properties stored on the Oracle Calendar server. For events, the following properties are available:

```
uid                             # a UID settable upon creation
x-oracle-event-guid             # identifier of the main event
x-oracle-eventinstance-guid     # identifier of the instance within
                                # the event
x-oracle-data-guid              # mapped to x-oracle-eventinstance-guid
```

For tasks, the following properties are available:

```
uid                          # a UID settable upon creation
x-oracle-data-guid           # generated internally by the Application
                             # Server (OCAS) and cannot be used against
                             # any other product. This will be changed
```

```
                                    # once the Calendar Store supports the
                                    # x-oracle-task-guid attribute.
```

SOAP Envelope

The SOAP Envelope is a predefined XML packet used to identify the SOAP message:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
     soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
     xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
     xmlns:xsd="http://www.w3.org/1999/XMLSchema">
     ...soap header...
     ...soap body...
</soap:Envelope>
```

Xsi and xsd are options defining a namespace used within the message; these will appear if required (i.e. if there is an element in the soap header or soap body requiring it).

Xsd is used to provide basic predefined type definitions, such as string, integer, etc. Xsi is used to define the "type" attribute for an entry.

```
<location xsi:type="xsd:string">Soleil</location>
```

There are 3 main ways of providing type information within SOAP:

- The data content types are agreed to by both parties ahead of time. This is not useful for general SOAP interaction, only one-to-one site integrations.
- Using XML Schemas, where the schema and namespace is used to relate all typing information.
- Using XML Schemas and explicit type attributes, where each element in the SOAP XML tree requires an xsi:type attribute.

Since xCal and CWSL have their own XML Schemas, they do not use explicit type attributes.

There are important issues to be outlined at this point; most current SOAP implementations add an XML document header line before the SOAP envelope. However this is not part of the current SOAP v1.1 specification, but an improvement included in SOAP v1.2.

```
<?xml version="1.0" encoding="UTF-8" ?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
               soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
     xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
     xmlns:xsd="http://www.w3.org/1999/XMLSchema">
     ...soap header...
     ...soap body...
</soap:Envelope>
```

In order to maintain consistency between SOAP implementations, the default behavior is to provide the XML document header if the original request has one.

```
POST <uri> HTTP/1.1
Content-Type: text/xml; charset="UTF-8"
Content-Length: <char length>
User-Agent: <user agent>
SOAPAction: <urn>
```

```
...soap envelope...
```

SOAP Body

The SOAP body contains the actual methods used to perform actions on Calendar Server and Web services errors.

SOAP Faults

When any kind of error is returned, a SOAP Fault element appears in the Body of the SOAP response. Within a SOAP fault, there are specific elements to be provided:

- A faultcode, which can be one of the following values:
  - VersionMismatch indicating the SOAP namespace is incorrect.
  - Client indicating a problem originating from the incoming message.
  - Server indicating a problem occurred during the processing of the request.
- A faultstring, which is the textual message of the error that has occurred. This is the application system error string. The default string language is English.
- A detail element, used as the container to provide extended information. In our case, the complete application system error log entry is returned to the SOAP client. If server side event logging is set to debug in ocas.conf, then Line, FileName, Version, LastMod, and Author are returned.

```
<soap:Body>
     <soap:Fault>
          <faultcode>soap:Server</faultcode>
     <faultstring>Unable to locate the entry in
                    the preferences
          </faultstring>
          <detail>
               <cwsl:Error xmlns:cwsl=
               "http://www.oracle.com/WebServices/Calendaring/1.0/">
                    <Class>Error::Data::CalConnection</Class>
                    <Code>000C-01-00-00000029</Code>
                    <Line>1450</Line>
                    <FileName>UniapiConnection.cpp,v</FileName>
                    <Version>1.43</Version>
                    <LastMod>2002/05/23 20:54:48</LastMod>
                    <Author>frederic</Author>
                    <Date>Web May 29 14:05:42 2002</Date>
                    <PID>19458</PID>
                    <TID>5</TID>
               </cwsl:Error>
          </detail>
     </soap:Fault>
</soap:Body>
```

As an example, the preceding Code tag indicates the type of error as follows:

Generally you need only concern yourself with the first and last segments, which in this case are:

- Module 000C = SYS_MODULE_DATAACCESS
- Error 00000029 = e_soapSOAPRequestCode_MissingModifyCmd

For a list of Module and Error values, see Chapter 13, "Oracle Calendar Web Services Status Codes".

A fault can occur at any point in the access of interaction with various components within the application system and the Calendar Server.

Calendar Web Service Language (CWSL)

CWSL defines the grammar to be used to exchange data between a calendar SOAP client and calendar SOAP server. The following methods, taken directly from the CAP draft dated March 2002, provide the main functionality for the Calendar Web service language. It should be noted that some of the CAP method names are reused here in the CWSL, but the semantics and meaning are changed to reflect a Web-based protocol environment.

The Calendar language uses "http://www.oracle.com/WebServices/Calendaring/1.0/" as the namespace.

The following session command is supported:

- NoOp performs no operation on the data store, but is used to preauthenticate.

The following calendar commands are supported:

- Ping performs a simple check to ensure that Web services is active.
- Create performs a create of a new meeting.
- Delete performs a delete of an event or instance.
- Modify performs an update of an event or instance for specific properties.
- Search performs a request to retrieve data through the service.
- Summary counts the number of unconfirmed events, open active tasks, and overdue active tasks in a given time range.

It is important to note that some of these methods are greatly restricted in this release.

NoOp

Verifies the validity of an authentication SOAP header, without executing anything on the server. The NoOp command can only fail if there is a SOAP header problem.

Sample Request

```
<SOAP-ENV:Body>
  <cwsl:NoOp xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/"/>
</SOAP-ENV:Body>
```

Sample Successful Reply

```
 <soap:Body>
   <NoOpReply>
   </NoOpReply>
 </soap:Body>
```

Ping

Tests to see if the Oracle Calendar Web service application server is active. The command has no other effect on Web services.

Sample Request

```
<SOAP-ENV:Body>
  <cwsl:Ping xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/"/>
</SOAP-ENV:Body>
```

Sample Successful Reply

```
 <soap:Body>
   <PingReply>
   </PingReply>
 </soap:Body>
```

Create

Creates <vevent> components on the Oracle Calendar server. In this release, the Create method has the following issues and limitations:

- The method does not support the creation of <vtodo> components.
- There is no current mechanism for creating a recurring meeting.
- There is no current mechanism for creating a meeting with additional attendees.

The response will be either a SOAP fault or a CreateReply containing the GUID of the event just created.

Sample Request

Certain properties must be specified in the Create command. Chapter 12, "Oracle Calendar Web Services Supported Data Components and Properties" indicates these properties. The following is an example of creating an event:

```
<SOAP-ENV:Body>
  <cwsl:Create xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>Testoraptor Command 31</CmdId>
    <iCalendar>
      <vcalendar prodid="-//Oracle//Calendaring//Calendarlet//EN" version="2.0">
        <vevent>
          <class>PRIVATE</class>
          <description>Sample Event Create</description>
          <dtstart value="DATE">20050113</dtstart>
          <duration>PT00H00M</duration>
          <location>Vision Corporation</location>
          <summary>EventCreate Alarm Display Type Day Event Test</summary>
          <uid>TESTORAPTOR-UID-2</uid>
          <x-oracle-eventtype>DAY EVENT</x-oracle-eventtype>
          <x-oracle-isrtcenabled>false</x-oracle-isrtcenabled>
          <valarm>
            <action>DISPLAY</action>
            <trigger>-PT1H05M</trigger>
          </valarm>
        </vevent>
      </vcalendar>
    </iCalendar>
  </cwsl:Create>
</SOAP-ENV:Body>
```

Sample Successful Reply

The following is an example of a successful reply from a request to create a new event:

```
<soap:Body>
  <cwsl:CreateReply
    xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>Testoraptor Command 31</CmdId>
    <x-oracle-data-guid>E1+352322565+14503+442957968</x-oracle-data-guid>
  </cwsl:CreateReply>
</soap:Body>
```

Delete

The Delete SOAP method provides the ability to delete a meeting, daily note or day event. This includes the ability to delete an instance of a repeating/recurring meeting.

Request Syntax

The vQuery uniquely identifies a previously returned data GUID. The GUID contains all the information to uniquely identify the instance of a repeating meeting or the recurrence of a recurrence rule.

```
<SOAP-ENV:Body>
  <cwsl:Delete xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>Testoraptor Command 768</CmdId>
    <vQuery>
      <From>VEVENT</From>
      <Where>
        x-oracle-data-guid =
          'Data GUID of an event or an instance of a repeating event'</Where>
    </vQuery>
  </cwsl:Delete>
</SOAP-ENV:Body>
```

The vQuery is used to identify the item to be deleted; only one item can be deleted at a time.

Sample Request

```
<SOAP-ENV:Body>
  <cwsl:Delete xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>Testoraptor Command 768</CmdId>
    <vQuery>
      <From>VEVENT</From>
      <Where>x-oracle-data-guid = 'E1+469763101+14503+442958019'</Where>
    </vQuery>
  </cwsl:Delete>
</SOAP-ENV:Body>
```

Sample Successful Reply

DeleteReply will contain the GUID of the deleted item.

```
<soap:Body>
  <DeleteReply>
    <CmdId>Testoraptor Command 768</CmdId>
    <x-oracle-data-guid>E1+469763101+14503+442958019</x-oracle-data-guid>
  </DeleteReply>
</soap:Body>
```

Modify

Modifies, adds, or deletes an event's properties. Only a few properties may be modified, added, or deleted.

Request Syntax

The Modify method is made up of three sections: the query, the iCalendar object properties, and the new iCalendar object properties:

```
<SOAP-ENV:Body>
  <cwsl:Modify xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <vQuery>
      <From>
        <!-- VEVENT-->
      </From>
      <Where>
        <!-- The event query -->
      </Where>
    </vQuery>
    <iCalendar>
      <!-- The event properties (and values) to be changed -->
    </iCalendar>
    <iCalendar>
      <!-- The new event properties and values -->
    </iCalendar>
  </cwsl:Modify>
</SOAP-ENV:Body>
```

Only certain properties may be modified with the Modify command. Chapter 12, "Oracle Calendar Web Services Supported Data Components and Properties" indicates these properties.

If there are other properties within the modify SOAP method, a SOAP fault will be generated (for example, class, uid, x-oracle-eventtype, Web Conferencing attributes).

vQuery Section

The vQuery section is used to identify the calendar component to be modified. Only one calendar component can be modified at a time.

The vQuery uniquely identifies a previously returned data GUID. The GUID contains all the information to uniquely identify the instance of a repeating meeting or the recurrence of a recurrence rule. The vQuery is used to identify the item to be modified; only one item can be modified at a time, including one simple event (meeting, daily note, or day event) or one instance of a repeating meeting. In a modify operation, properties can be changed, added, or removed.

Modifying Properties

The first iCalendar object contains the properties to be modified, along with the original values. If there are attributes associated with them, those must be present as well. The second iCalendar object contains the new properties values to be applied.

```
<!-- The original event property/values -->
<iCalendar>
  <vcalendar>
    <vevent>
      <summary>My old title</summary>
```

```
      <location>My old location</location>
    </vevent>
  </vcalendar>
</iCalendar>
<!-- The modified event property/values -->
<iCalendar>
  <vcalendar>
    <vevent>
      <summary>My new title</summary>
      <location>My new location</location>
    </vevent>
  </vcalendar>
</iCalendar>
```

Adding Properties

The first iCalendar object does not contain any reference to the property to be added. The second iCalendar object contains the new property and value.

```
<!-- The original event property/values -->
<iCalendar>
  <vcalendar>
    <vevent>
    </vevent>
  </vcalendar>
</iCalendar>
<!-- The modified event property/values -->
<iCalendar>
  <vcalendar>
    <vevent>
      <summary>My new title</summary>
    </vevent>
  </vCalendar>
</iCalendar>
```

Deleting Properties

The first iCalendar object contains the original property and value. The second iCalendar object does not contain the property.

```
<!-- The original event property/values -->
<iCalendar>
  <vcalendar>
    <vevent>
      <summary>My old title</summary>
    </vevent>
  </vcalendar>
</iCalendar>
<!-- The modified event property/values -->
<iCalendar>
  <vcalendar>
    <vevent>
    </vevent>
  </vcalendar>
</iCalendar>
```

Reply Syntax

The Modify reply returns the GUID of the modified event. It is very important to note that the GUID can change depending on the type of change applied to the Oracle Calendar server. Some updates require a delete/recreate type of interaction.

```
<!-Received from Web Services -->
<!-- failure will result in a SOAP fault -->
<Body>
     <ModifyReply>
          <CmdId>a command id</CmdId>
          <x-oracle-data-guid>cccc</x-oracle-data-guid>
     </ModifyReply>
</Body>
```

Sample Request

```
<SOAP-ENV:Body>
  <cwsl:Modify xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>Testoraptor Command 468</CmdId>
    <vQuery>
      <From>VEVENT</From>
      <Where>x-oracle-data-guid = 'E1+335545478+14503+442957967'</Where>
    </vQuery>
    <iCalendar>
      <vcalendar prodid="-//Oracle//Calendaring//Calendarlet//EN" version="2.0">
        <vevent>
          <class>PUBLIC</class>
          <x-oracle-isrtcenabled>false</x-oracle-isrtcenabled>
        </vevent>
      </vcalendar>
    </iCalendar>
    <iCalendar>
      <vcalendar prodid="-//Oracle//Calendaring//Calendarlet//EN" version="2.0">
        <vevent>
          <class>CONFIDENTIAL</class>
          <x-oracle-isrtcenabled>false</x-oracle-isrtcenabled>
        </vevent>
      </vcalendar>
    </iCalendar>
  </cwsl:Modify>
</SOAP-ENV:Body>
```

Sample Successful Reply

The Modify method reply returns the GUID of the modified calendar component. It is very important to note that the GUID can change depending on the type of change applied to the Oracle Calendar server. Some updates require a delete/recreate type of interaction.

```
<soap:Body>
  <ModifyReply>
    <CmdId>Testoraptor Command 468</CmdId>
    <x-oracle-data-guid>E1+335545478+14503+442957967</x-oracle-data-guid>
  </ModifyReply>
</soap:Body>
```

Search

Retrieve events, tasks, contacts, and user information from the Oracle Calendar server.

Request Syntax

<CmdId> is a SOAP client-provided string and appears in the response to identify the originating Search entry.

<vQuery> is the search query criteria and can only appear once. (See the following section, "vQuery".)

```
<soap:Body>
  <cwsl:Search xmlns:cwsl=
    "http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>
      <!-- ID string provided by the client -->
    </CmdId>
    <vQuery>
      <!-- The search query criteria -->
    </vQuery>
    <x-oracle-limitattendees>
      <!-- Integer value -->
    </x-oracle-limitattendees>
    <x-oracle-overlap
      <!-- Either the string "on" or "off" -->
    </x-oracle-overlap>
    <x-oracle-searchhandle></x-oracle-searchhandle>
    <x-oracle-timestamp>
      <!-- Timestamp string in UTC format -->
    </x-oracle-timestamp>
    <x-oracle-basicsearch>
      <!-- Search string -->
    </x-oracle-basicsearch>

  </cwsl:Search>
</soap:Body>
```

The following properties are optional:

<x-oracle-limitattendees>
Integer value that limits the number of attendees from the Calendar store. If no attendee information is required, set this value to "0". Retrieving attendee information is typically a very expensive operation on the Calendar store.

If this property is not set, all attendee information will be returned, limited by Web Services configuration and Calendar store settings.

<x-oracle-overlap>
If this property is set to "on", the Search method will retrieve events that overlap the specified date range. For example, if the date range is today, and this property is set to "on", the Search method would retrieve an event that starts yesterday and finishes tomorrow. If this property is set to "off", the Search method would not retrieve this event.

By default, this property is set to "off".

The following properties are required if you want to use the capabilities of Ultra Search:

<x-oracle-searchhandle>
If this element exists in the SOAP request, the internal Calendar store search API will be used. Set the value of this property to the empty string, "".

<x-oracle-timestamp>
An xCal UTC string that represents the timestamp of the SOAP request. Ultra Search uses this timestamp if no dstart information is provided.

The following property is only available if you are using the capabilities of Ultra Search:

<x-oracle-basicsearch>
The Search method will retrieve events that contain the specified string in any of the <title>, <location>, or <description> properties.

Reply Syntax

The data returned is contained within a <cwsl:Reply>. There is one <cwsl:Reply> element for each <cwsl:Search> element.

```
<soap:Body>
  <cwsl:Reply xmlns:cwsl=
    "http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>
      <!-- ID string from the originating Search SOAP request -->
    </CmdId>
      <!-- Data returned from the Search command -->
  </cwsl:Reply>
</soap:Body>
```

vQuery

The vQuery forms the basis for requesting data from the Calendar store.

Searching for Events

Events can be queried by a single unique identifier or by a series of identifiers:

```
<!-- Ability to fetch a single event -->
<!-- Remove the Where clause to return all events -->
<vQuery>
     <From>VEVENT</From>
     <Where>x-oracle-data-guid='event guid'</Where>
</vQuery>

<!-- Ability to fetch a multiple events -->
<vQuery>
     <From>VEVENT</From>
     <Where>
          x-oracle-data-guid ='event id 1' OR
          x-oracle-data-guid ='event id 2' OR
          x-oracle-data-guid ='event id 3'
     </Where>
</vQuery>
```

Events can be queried by date range:

```
<!-- Ability to fetch events within a time range -->
<vQuery>
     <From>VEVENT</From>
     <Where>dtstart &gt;= 'starttime' AND dtend &lt;= 'endtime'</Where>
</vQuery>
```

In this example, starttime and endtime provide the time range, in UTC, to be returned. Note the proper XML encoding of the string within the <Where> clause.

Note: All other event query forms will generate an error. In particular, date range queries bust be in the following form:

```
DTEND >= start_date && DTSTART <= end_date
```

Other date range queries are possible only with the Calendarlet or the public static String getDateRangeQuery(Calendar in_startDate, Calendar in_endDate) method in the CalendarUtils class.

The event query result set is returned using the xCal draft specification, embedded within the <cwsl:Reply> tag.

```
<soap:Body>
     <cwsl:Reply xmlns:cwsl=
          "http://www.oracle.com/WebServices/Calendaring/1.0/">
          <CmdId>a client id</CmdId>
<xCal:iCalendar xmlns:xCal=
          "http://www.oracle.com/WebServices/Calendaring/1.0/">
     <vcalendar version="2.0" prodid=...>
          <vevent>
               <x-oracle-data-guid>fjldjfdslkjfdksj
               </x-oracle-data-guid>
               <dtstamp>19980309T231000Z</dtstamp>
               <uid>ffdtasfdtasfdta</uid>
               <summary>My event</summary>
               <location>Soleil</location>
               <x-oracle-eventtype>PUBLIC</x-oracle-eventtype>
          </vevent>
     </vcalendar>
</xCal:iCalendar>
     </cwsl:Reply>
</soap:Body>
```

There will be no sorting of returned data.

Searching for Tasks

Tasks can be queried by a single unique identifier or by a series of identifiers:

```
<!-- Ability to fetch a single task -->
<!-- Remove the Where clause to return all tasks -->
<vQuery>
     <Select>*</Select>
     <From>VTODO</From>
     <Where>x-oracle-data-guid = 'task guid'</Where>
</vQuery>

<!-- Ability to fetch multiple tasks -->
<vQuery>
     <Select>*</Select>
     <From>VTODO</From>
     <Where>
          x-oracle-data-guid = 'task id 1' OR
          x-oracle-data-guid = 'task id 2' OR
          x-oracle-data-guid = 'task id 3'
     </Where>
```

```
</vQuery>
```

The <Where> clause contains the x-oracle-data-guid = string where the right-hand side is an iCal task GUID.

Active tasks can be queried by date range:

```
<!-- Ability to fetch active tasks by time range -->
<vQuery>
     <From>VTODO</From>
     <vCall>
          <ActiveTasks>
               <StartTime>20020701T000000Z</StartTime>
               <EndTime>20020801T000000Z</EndTime>
          </ActiveTasks>
     </vCall>
</vQuery>
```

<Select> may be provided, however it is not supported in the current version of Web services. All attributes are returned in the reply.

There is no sort order for the returned data.

The <vCall> element indicates the use of an internal procedure (like a database stored procedure). The child element provides the name of the stored procedure to be invoked, as well as the arguments required by the call.

The task query result set is returned using the xCal draft specification, embedded within the <cwsl:Reply> tag.

```
<soap:Body>
     <cwsl:Reply xmlns:cwsl=
          "http://www.oracle.com/WebServices/Calendaring/1.0/">
          <CmdId>a client id</CmdId>
<xCal:iCalendar xmlns:xCal=
     "http://www.oracle.com/WebServices/Calendaring/1.0/">
     <vcalendar version="2.0" prodid=...>
          <vtodo>
               <x-oracle-data-guid>JKLFJLJK</x-oracle-data-guid>
               <uid>ffdtasfdtasfdta</uid>
               <dtstamp>19980309T231000Z</dtstamp>
               <summary>My task</summary>
               <priority>2</priority>
          </vtodo>
     </vcalendar>
</xCal:iCalendar>
     </cwsl:Reply>
</soap:Body>
```

Ultra Search Capabilities

In addition to retrieving events in a given date range, you may use the capabilities of Ultra Search to search for text strings in <summary>, <description>, and <location> properties.

The following example searches for events that take place between February 25, 2005, 5:00am and February 26, 2006, 4:59am and contain the string "Montreal" in the <location> property:

```
<vQuery>
  <From>VEVENT</From>
  <Where>DTSTART &gt;= '20050225T050000Z'
```

```
    AND DTEND &lt;= '20050226T045900Z'
    AND LOCATION=Montreal
  </Where>
  <x-oracle-searchhandle>
  </x-oracle-searchhandle>
  <x-oracle-timestamp>20050225T050000Z</x-oracle-timestamp>
</vQuery>
```

You may also use the <x-oracle-basicsearch> property to search for events that contain a specified string in any of the <title>, <location>, or <description> properties.

You may use Ultra Search capabilities only for searching events.

Performance Issues

To limit the potential impact on the Oracle Calendar server, the ocws.conf file has a few settings to override any query that is received.

```
[webservices]
maxattendee=200          # limit the total number of attendees that can
                         # be returned per instance. The default is 200.
maxresults=200           # limit the total number of meetings or tasks
                         # that can be returned in one query
```

If you use Ultra Search to retrieve events by searching for a string (as opposed to searching by <x-oracle-data-guid>), the attendees of those retrieved events will not be listed, other than the one specified in the property <x-oracle-loginuser>. However, if you search by <x-oracle-data-guid>, the attendees of the retrieved events will be listed.

Attendees are listed only for searches by <x-oracle-data-guid> because retrieving attendee lists may have a negative impact on performance.

Sample Request

```
<SOAP-ENV:Body>
  <cwsl:Search xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>Testoraptor Command 458</CmdId>
    <vQuery>
      <From>VEVENT</From>
      <Where>LOCATION=OracleUltraSearchLocationText
        AND DTSTART &gt;=20040913T040000Z AND DTEND &lt;=20050113T050000Z
      </Where>
      <x-oracle-searchhandle>
      </x-oracle-searchhandle>
      <x-oracle-timestamp>20050113T050000Z</x-oracle-timestamp>
    </vQuery>
  </cwsl:Search>
</SOAP-ENV:Body>
```

Sample Successful Reply

```
<soap:Body>
  <cwsl:Reply xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>Testoraptor Command 458</CmdId>
    <x-oracle-searchhandle>
    </x-oracle-searchhandle>
    <iCalendar
      xmlns:xcal="http://www.oracle.com/WebServices/Calendaring/1.0/">
      <vcalendar version="2.0" prodid="-//Oracle//Calendaring//OCAS//EN">
        <vevent>
          <uid>TESTORAPTOR-UID-76</uid>
```

```
    <transp>OPAQUE</transp>
    <summary>OracleUltraSearchText</summary>
    <status>CONFIRMED</status>
    <priority>5</priority>
    <organizer cn="Web Services">
      mailto:webserivces@us.oracle.com
    </organizer>
    <location>OracleUltraSearchLocationText</location>
    <description>OracleUltraSearchText</description>
    <dtstart>20041129T050000Z</dtstart>
    <dtend>20041129T060000Z</dtend>
    <class>PRIVATE</class>
    <attendee cutype="INDIVIDUAL" cn="Web Services" partstat="ACCEPTED">
      mailto:webserivces@us.oracle.com
    </attendee>
    <x-oracle-areothersinvited>FALSE</x-oracle-areothersinvited>
    <x-oracle-data-guid>E1+251659385+14503+442958018</x-oracle-data-guid>
    <x-oracle-isextrainstance>FALSE</x-oracle-isextrainstance>
    <x-oracle-isexception>FALSE</x-oracle-isexception>
    <x-oracle-loginuser cn="Web Services">
      mailto:webserivces@us.oracle.com
    </x-oracle-loginuser>
    <x-oracle-eventinstance-guid>
      I1+251659385+14503+1+442958018
    </x-oracle-eventinstance-guid>
    <x-oracle-event-guid>
      E1+251659385+14503+442958018
    </x-oracle-event-guid>
    <x-oracle-eventtype>APPOINTMENT</x-oracle-eventtype>
    <x-oracle-search-relevance>84</x-oracle-search-relevance>
  </vevent>
</vcalendar>
<vcalendar version="2.0" prodid="-//Oracle//Calendaring//OCAS//EN">
  <vevent>
    <uid>TESTORAPTOR-UID-78</uid>
    <transp>OPAQUE</transp>
    <summary>OracleUltraSearchText</summary>
    <status>CONFIRMED</status>
    <priority>5</priority>
    <organizer cn="Web Services">
      mailto:webserivces@us.oracle.com
    </organizer>
    <location>OracleUltraSearchLocationText</location>
    <description>OracleUltraSearchText</description>
    <dtstart>20041129T050000Z</dtstart>
    <dtend>20050309T050000Z</dtend>
    <class>PRIVATE</class>
    <attendee cutype="INDIVIDUAL" cn="Web Services" partstat="ACCEPTED">
      mailto:webserivces@us.oracle.com
    </attendee>
    <x-oracle-areothersinvited>FALSE</x-oracle-areothersinvited>
    <x-oracle-data-guid>
      E1+469763101+14503+442958019
    </x-oracle-data-guid>
    <x-oracle-isextrainstance>FALSE</x-oracle-isextrainstance>
    <x-oracle-isexception>FALSE</x-oracle-isexception>
    <x-oracle-loginuser cn="Web Services">
      mailto:webserivces@us.oracle.com
    </x-oracle-loginuser>
    <x-oracle-eventinstance-guid>
```

```
            I1+469763101+14503+1+442958019
          </x-oracle-eventinstance-guid>
          <x-oracle-event-guid>
            E1+469763101+14503+442958019
          </x-oracle-event-guid>
          <x-oracle-eventtype>APPOINTMENT</x-oracle-eventtype>
          <x-oracle-search-relevance>100</x-oracle-search-relevance>
        </vevent>
      </vcalendar>
    </iCalendar>
  </cwsl:Reply>
</soap:Body>
```

Summary

Counts the number of unconfirmed events, open active tasks, and overdue active tasks in a given time range.

Request Syntax

The Summary command consists of one or two <vCall> elements. To count the number of unconfirmed events in a given time range, use the <EventsCount> element in a <vCall> element. To count the number of open or overdue active tasks in a given time range, use the <ActiveTasksCount> element.

Either an <EventCount> or an <ActiveTasksCount> element may appear in a <vCall> element. However, you may count the number of both unconfirmed events and open and overdue tasks. In this case, use two <vCall> elements.

Specify the start and end times of the time ranges in UTC time.

The following is the structure of the SOAP body of the Summary command:

```
<SOAP-ENV:Body>
  <cwsl:Summary xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>
      <!-- The command ID. This string is defined by the developer. -->
    </CmdId>
    <vCall>
      <EventsCount>
        <StartTime>
          <!-- UTC date-time value. The date and time of the start
               of the time range. -->
        </StartTime>
        <EndTime>
          <!-- UTC date-time value. The date and time of the end
               of the time range. -->
        </EndTime>
        <Unconfirmed>
          <!-- Boolean. If set to TRUE, the SOAP reply will indicate
               the number of unconfirmed tasks within the given
               time range. -->
        </Unconfirmed>
      </EventsCount>
    </vCall>
    <vCall>
      <ActiveTasksCount>
        <StartTime>
          <!-- UTC date-time value. The date and time of the start
               of the time range. -->
        </StartTime>
        <EndTime>
          <!-- UTC date-time value. The date and time of the end
               of the time range. -->
        </EndTime>
        <Open>
          <!-- Boolean. If set to TRUE, the SOAP reply will indicate
               the number of open active tasks within the given
               time range. -->
        </Open>
        <Overdue>
          <!-- Boolean. If set to TRUE, the SOAP reply will indicate
```

```
                the number of overdue active tasks within the given
                time range. -->
          </Overdue>
        </ActiveTasksCount>
      </vCall>
    </cwsl:Summary>
</SOAP-ENV:Body>
```

Reply Syntax

Depending on the value of <Unconfirmed>, <Open>, and <Overdue>, a successful Summary command reply will contain <Unconfirmed>, <Open>, and <Overdue> elements that indicate the number of unconfirmed events, open active tasks, and overdue active tasks, respectively.

The following is the structure of the SOAP reply of the Summary command:

```
<soap:Body>
  <cwsl:SummaryReply
    xmlns:cwsl="http://www.oracle.com/WebServices/Calendaring/1.0/">
    <CmdId>Testoraptor Command 801</CmdId>
    <EventsCount>
      <Unconfirmed>
        <!-- Integer. Number of unconfirmed events. -->
      </Unconfirmed>
    </EventsCount>
    <ActiveTasksCount>
      <Open>
        <!-- Integer. Number of open active tasks. -->
      </Open>
      <Overdue>
        <!-- Integer. Number of overdue active tasks. -->
      </Overdue>
    </ActiveTasksCount>
  </cwsl:SummaryReply>
</soap:Body>
```

11

Oracle Calendar Web Services Client-Side Java Implementation

This chapter describes the design of the set of Java classes used to provide contextual collaboration through the access of calendar data through Oracle Calendar Web services. These "Calendarlet" classes attempt to hide the many details of using Web services technology in a Java environment.

See *Oracle Calendar Web Services Java API Reference* for more information about these classes.

The class implementation does not attempt to provide all the iCalendar properties and attributes.

Note: You can find JavaDoc information and TestTool samples in the Oracle Calendar Web services toolkit.

Java Classes

There are a few general steps to follow when using the Calendarlet classes:

- Initialize your authentication mechanism.
- Initialize your Query, including data type.
- Bind the authentication and query object to a Calendarlet instance.
- Set the target URL in the Calendarlet instance.
- Make the SOAP call.
- Parse the results.

The Calendarlet class implementation relies heavily on Apache SOAP classes to perform most of the protocol level handling. For incoming and outgoing messages, these same Apache SOAP classes are used, along with W3C DOM classes. To generate outgoing messages, Calendarlet and iCalendar classes are instantiated and set on parent classes. To generate the final XML stream, all classes implement a getElement() method. This is intended to build an XML DOM representation of the SOAP message to be transmitted. The lower level Apache SOAP calls require this DOM structure to obtain the final stream.

For incoming messages, the Calendarlet and iCalendar classes are reconstructed through the unmarshall() static method on each class, again using the XML DOM received from the lower level Apache SOAP classes. This unmarshalling of the DOM consists of the parent class recognizing a child tag and invoking that child's class unmarshall() method.

If for any reason there is an XML parsing error, a low level Apache SOAP exception is thrown; the Calendarlet classes will never get a chance to parse the data. If there is a contextual error, meaning the XML is valid but elements are in the wrong place or not recognized, a Calendarlet exception will be thrown.

Ideally, all incoming xCal (the XML binding of iCalendar) can have extended elements within the data. However, for this implementation, extended elements will only be handled at the vEvent level.

The Calendarlet class provides some debugging support. There are two main features:

- The ability to capture the input and output buffers; the method setWantIOBuffers() must be called before invoking a SOAP method (not recommended for a final deployment). Both the input and output buffers are captured and stored in the CalendaringResponse class.
- The ability to get the total processing time (in milliseconds) of the SOAP request, also stored within the CalendarletResponse.

Classpath

In order to use Oracle Calendar Web services, the following .jar files must be in your classpath:

- `$ORACLE_HOME/j2ee/home/lib/*`
- `$ORACLE_HOME/ocal/jlib/ocal_clnt.jar`
- `$ORACLE_HOME/soap/lib/soap.jar`
- `$ORACLE_HOME/lib/xmlparserv2.jar`

Creating Events and Web Conferences

The following code uses basic authentication to create one event or Web conference. This code consists of the following classes:

- Initialization: Connects to the Oracle Calendar server with basic authentication
- MyEventCreateTest: Creates one event or Web conference depending on the value of the parameter isWebConference in the MyEventCreateTest constructor.

Example 11–1 Initialization.java

```
package oracle.test.cal;

public class Initialization
{
  public Initialization() { }

  public void initBasicAuth(
    oracle.calendar.soap.client.Calendarlet cws,
    String name,
    String password,
    String endPointURL)
  {
    cws.setEndPointURL(endPointURL);
    cws.setWantIOBuffers(true);

    // Initialize the authentication information
    oracle.calendar.soap.client.authentication.BasicAuth auth =
```

```
      new oracle.calendar.soap.client.authentication.BasicAuth();
    auth.setName(name);
    auth.setPassword(password);

    // Set the basic authentication header
    cws.setAuthenticationHeader(auth.getElement());
  }
}
```

Example 11–2 MyEventCreateTest.java

```
parckage oracle.test.cal;

public class MyEventCreateTest
{
  private oracle.calendar.soap.iCal.vEvent vevent;
  private String currentEventGUID;
  private String k_startTime;
  private String k_baseDuration = "PT01H00M";
  private String k_baseLocation = "Tecumseh, Ontario";
  private String m_testName = "Event Create Test";
  private String m_uid = "UID-TEST-1";
  private String m_eventClass;
  private String m_xEventType;
  private String m_cmdid = "CMDID-TEST-CREATE-1";
  private String m_cmdid_delete = "CMDID-TEST-DELETE-1";
  private boolean m_isWebConference;

  public oracle.calendar.soap.iCal.vEvent getvEvent() { return vevent; }

  public String getEventGUID() { return currentEventGUID; }

  public MyEventCreateTest() { }

  public MyEventCreateTest(
    String startTime,
    String duration,
    String location,
    String summary,
    String UID,
    String eventClass,
    String xEventType,
    String commandID,
    boolean isWebConference)
  {
    k_startTime = startTime;
    k_baseDuration = duration;
    k_baseLocation = location;
    m_testName = summary;
    m_uid = UID;
    m_eventClass = eventClass;
    m_xEventType = xEventType;
    m_cmdid = commandID;
    m_isWebConference = isWebConference;
  }

  public void run()
  {
    try
    {
```

```
// Create the iCalendar that is to be
// created on the Oracle Calendar server

oracle.calendar.soap.iCal.iCalendar ical =
  new oracle.calendar.soap.iCal.iCalendar();
oracle.calendar.soap.iCal.vCalendar vcal =
  new oracle.calendar.soap.iCal.vCalendar();
vevent = new oracle.calendar.soap.iCal.vEvent();

ical.addvCalendar(vcal);
vcal.addvComponent(vevent);

// set the vEvent attributes
vevent.setEventClass(m_eventClass);

// Start time
vevent.setDtStart(k_startTime);

// Duration
vevent.setDuration(k_baseDuration);

// Location
vevent.setLocation(k_baseLocation);

// Summary
vevent.setSummary(m_testName);

// UID
vevent.setUid(m_uid);

// Event type
vevent.setXEventType(m_xEventType);

// Description
vevent.setDescription(ical.toString());

// Make this event enabled for Web conference
if (m_isWebConference) {
  vevent.setWebConferenceEnabled(true);
  vevent.setWebConferenceType(vevent.k_webConferenceTypePublic);
}

// Initialize the event create command
oracle.calendar.soap.client.CreateCommand create =
  new oracle.calendar.soap.client.CreateCommand();

create.setCmdId(m_cmdid);
create.setiCalendar(ical);

// Create the Oracle Calendar client SOAP stub
// and set the basic authentication header

System.out.println("Creating the Oracle Calendar client SOAP stub");
oracle.calendar.soap.client.Calendarlet cws =
  new oracle.calendar.soap.client.Calendarlet();

// Login
if (cws == null) {
  return;
}
```

```
    Initialization myInit = new Initialization();
      myInit.initBasicAuth(
        cws,
        "loginuser",
        "password",
        "http://www.example.com:7777/ocws-bin/ocas.fcgi");

    // Next, make the SOAP call
    System.out.println("Making the SOAP call");
    oracle.calendar.soap.client.CalendaringResponse response =
      cws.Create(create.getElement());

    // Now display the results
    System.out.println("SOAP send buffer:");
    System.out.println(response.getSendBuffer());
    System.out.println("SOAP receive buffer:");
    System.out.println(response.getReceiveBuffer());

    // Get the created event's GUID
    oracle.calendar.soap.client.CreateReply myCreateReply = null;

    try
    {
      System.out.println("Creating CreateReply");
      myCreateReply =
        (oracle.calendar.soap.client.CreateReply)
          response.getCalendarReply();
    }
    catch (Exception e)
    {
      myCreateReply = null;
    }
    if (myCreateReply == null)
    {
      System.out.println("Unable to create CreateReply");
      // There is nothing to do
      return;
    }

    currentEventGUID = myCreateReply.getDataGuid();

  }
  catch (Exception e)
  {
    System.out.println("Exception enountered:");
    System.out.println(e.getMessage());
    e.printStackTrace();

  }

}

/**
 * Main method
 *
 */
public static void main(String[] args)
{
  MyEventCreateTest myEventCreateTest =
```

```
      new MyEventCreateTest(
        "20050714T040000Z",
        "PT01H00M",
        "Somewhere exotic",
        "MyEventCreateTest8",
        "MyEventCreateTest-UID-8",
        oracle.calendar.soap.iCal.vEvent.k_eventClassPublic,
        oracle.calendar.soap.iCal.vEvent.k_eventTypeAppointment,
        "CommandID-MyEventCreateTest",
        true);

    myEventCreateTest.run();
  }
}
```

Creating Web Conferences

To create a Web conference, create an event with the following properties

- <x-oracle-isrtcenabled> set to true
- <x-oracle-rtc-securitytype> set to the security type of the Web conference. This can be either "REGULAR", "PUBLIC", or "RESTRICTED".
- <dtstart> set to a time in the future
- <summary> set to a short description of the Web conference

You may also configure the following Web conference properties:

- <x-oracle-rtc-attendee-url>
- <x-oracle-rtc-dialininfo>
- <x-oracle-rtc-host-url>
- <x-oracle-rtc-meetingid>
- <x-oracle-rtc-password>

See Chapter 12, "Oracle Calendar Web Services Supported Data Components and Properties" for more information about these properties.

Fetching Data

The following sample code performs the following:

- Creates a query that searches for all events within a date range of one week starting on today's date
- Uses basic authentication services to connect to the Oracle Calendar server
- Traverses through all iCalendar objects. Traverses through all vCalendar objects in each iCalendar object. Traverses through all vEvent objects in each vCalendar object, and outputs information about each vEvent object.

This code consists of the classes Initialization (which is listed previously) and MyFetchTest.

Example 11–3 MyFetchTest.java

```
package oracle.test.cal;

public class MyFetchTest
```

```
{
  public MyFetchTest() { }

  public oracle.calendar.soap.client.Reply run(
    oracle.calendar.soap.client.Calendarlet cws)
  {
    try {

      // Initialize the event search command and query
      oracle.calendar.soap.client.SearchCommand search =
        new oracle.calendar.soap.client.SearchCommand();
      search.setCmdId("MyUpdaeTestCommandID-1");

      // Create a query to retrieve events
      oracle.calendar.soap.client.query.vQuery query =
        new oracle.calendar.soap.client.query.vQuery();
      query.setFrom
        (oracle.calendar.soap.client.query.vQuery.k_queryFromEvent);

      // Determine the datestamps for a weeks worth of events.
      // Use the CalendarUtils to get a proper timestamp with
      // time zone information set properly

      java.util.Calendar today =
        oracle.calendar.soap.client.CalendarUtils.getToday();
      int dayOfWeek = today.get(java.util.Calendar.DAY_OF_WEEK);
      java.util.Calendar beginWeek = (java.util.Calendar)today.clone();
      java.util.Calendar endWeek = (java.util.Calendar)today.clone();
      beginWeek.add(java.util.Calendar.DATE, 1 - dayOfWeek);
      endWeek.add(java.util.Calendar.DATE, 8 - dayOfWeek);
      endWeek.add(java.util.Calendar.MINUTE, -1);

      // Use the CalendarUtils to help generate a date range query
      query.setWhere
        (oracle.calendar.soap.client.CalendarUtils.getDateRangeQuery
          (beginWeek, endWeek));
      search.setQuery(query);

      // Make the SOAP call
      oracle.calendar.soap.client.CalendaringResponse response
        = cws.Search(search.getElement());

      // Now display the results
      System.out.println("SOAP send buffer:");
      System.out.println(response.getSendBuffer());
      System.out.println("SOAP receive buffer:");
      System.out.println(response.getReceiveBuffer());

      // Get the SOAP reply
      oracle.calendar.soap.client.Reply reply =
        (oracle.calendar.soap.client.Reply)
          response.getCalendarReply();

      return reply;
    } catch (Exception e)
    {
      System.out.println("Exception enountered:");
      System.out.println(e.getMessage());
      e.printStackTrace();
    }
```

```
  return null;
}

public String outputReply (
  oracle.calendar.soap.client.Reply replyArg)
{
    String returnString = "";

    // Traverse all the iCalendar objects
    java.util.Vector someiCalendars =
      oracle.calendar.soap.iCal.iCalendar.unmarshallVector(
        replyArg.getEntries());

    int numiCalendars = someiCalendars.size();

    for (int i = 0; i < numiCalendars; i++)
    {
      oracle.calendar.soap.iCal.iCalendar iCalObj =
        (oracle.calendar.soap.iCal.iCalendar)
          someiCalendars.get(i);

      // Traverse all the vCalendar objects
      java.util.Vector somevCalendars = iCalObj.getvCalendars();

      int numvCalendars = somevCalendars.size();

      for (int j = 0; j < numvCalendars; j++)
      {
        oracle.calendar.soap.iCal.vCalendar vCalObj =
          (oracle.calendar.soap.iCal.vCalendar)
            somevCalendars.get(j);

        // Traverse all the vEvent objects
        java.util.Vector somevEvents = vCalObj.getComponents();
        int numvEvents = somevEvents.size();
        for (int k = 0; k < numvEvents; k++)
        {
          oracle.calendar.soap.iCal.vEvent vEventObj =
            (oracle.calendar.soap.iCal.vEvent)somevEvents.get(i);

          // Get the specific properties
          String title = vEventObj.getSummary();
          String dtstart = vEventObj.getDtStart();
          String dtend = vEventObj.getDtEnd();
          String eventType = vEventObj.getXEventType();

          returnString += "iCalendar " + i + ", vCalendar " + j
            + ", vEvent " + k + ":" + "\n";

          returnString += "Title: " + title + "\n";
          returnString += "Start time: " + dtstart + "\n";
          returnString += "End type: " + dtend + "\n";
          returnString += "Event type: " + eventType + "\n";

          // Do something interesting with the meeting info
        }
      }
    }
    return returnString;
}
```

```
  /**
   * Main method
   */

  public static void main(String[] args)
  {
    // Create the Oracle Calendar client SOAP stub
    // and set the basic authentication header
    oracle.calendar.soap.client.Calendarlet cws =
    new oracle.calendar.soap.client.Calendarlet();

    // Specify login username, password, and host
    Initialization myInit = new Initialization();
    myInit.initBasicAuth(
      cws,
      "loginuser",
      "password",
      "http://www.example.com:7777/ocws-bin/ocas.fcgi");

    MyFetchTest myFetchTest = new MyFetchTest();
    oracle.calendar.soap.client.Reply mainReply =
      myFetchTest.run(cws);
    System.out.println(myFetchTest.outputReply(mainReply));
  }
}
```

SOAP Faults and Exceptions

There are two types of errors that can occur with the Web services toolkit: A Java exception or a SOAP fault.

The Java Exception originates from the Calendarlet class, the underlying Apache SOAP or W3C DOM classes, or the Java Runtime. For each SOAP method that is invoked on the Calendarlet class, an exception may be thrown as a result of some internal processing error or an XML parsing problem. These are typically client-side unexpected errors that must be properly handled.

The SOAP Fault is the result of an error from the Oracle Calendar Web service (that is, a remote server-side error). Whenever a server-side error occurs, the Web service returns a SOAP Fault as the response to the HTTP transaction. There is no Java-based exception thrown. Within a SOAP Fault, the details field may contain an Oracle Calendar Web services Error object, with an important error code. For a list of error codes, see Chapter 13, "Oracle Calendar Web Services Status Codes".

A CalendarUtils method can help determine whether a SOAP fault has occurred and retrieve the Web services error.

```
// Calendar response
CalendaringResponse response = cws.Search(...);

// get the vector of entries embedded
// in the SOAP body
Vector bodyEntries = response.getBodyEntries();

// determine if there was a SOAP Fault
if (!CalendarUtils.isSOAPFault(bodyEntries))
{
  // do regular processing
```

```
}
else
{
  // get the SOAP fault object
  org.apache.soap.Fault soapFault =
  CalendarUtils.getSOAPFault(bodyEntries);

  // get the Oracle Calendar Web services error
  Error calendaringError = Error.unmarshall(soapFault);

  // get the Web services error code
  String errorCode = calendaringError.getCode();
}
```

Local Time

There are two important date formats to be aware of: Date and DateTime. The DateTime format contains both date and time information within the string, while Date contains only date information. DateTime is generally used for regular meetings, while Date is used for Day Events, Daily Notes, and Holidays.

To generate UTC datetime strings for a vQuery, use the CalendarUtils.getUTCDateTime method. It takes a standard Java Calendar class object and generates a string of the form *yyyyMMdd*T*hhmmss*Z. Java's Calendar class can have a Java time zone associated with it. It is up to the user of the Calendarlet classes to determine the proper time zone and set it in the Java Calendar object.

```
// Set the date through some mechanism
// Ensure the proper time zone is set
TimeZone localTimezone = TimeZone.getDefault();
Calendar theDate       = Calendar.getInstance(localTimezone);

String utcString = CalendarUtils.getUTCDateTime(theDate);
```

To generate UTC date strings for a vQuery, use the CalendarUtils.getUTCDate method. It takes a standard Java Calendar class object and generates a string of the form *yyyyMMdd*. Java's Calendar class can have a Java time zone associated with it. It is up to the user of the Calendarlet classes to determine the proper time zone and set it in the Java Calendar object.

```
// Set the date through some mechanism
// Ensure the proper time zone is set
TimeZone localTimezone = TimeZone.getDefault();
Calendar theDate       = Calendar.getInstance(localTimezone);

String utcString = CalendarUtils.getUTCDate(theDate);
```

Since many calendar query operations are relative to today's date, an additional CalendarUtils method is provided to help base vQuery datetime stamps. The method returns a datetime stamp of midnight today, local time of the form *yyyyMMdd*T*hhmm*00Z, where*hhmm* is the hour and minute offset from UTC (note that some time zones are half-hour offsets).

```
String utcToday = CalendarUtils.getToday();
```

Typical Oracle Calendar server query time ranges are from local midnight of a specific date to one minute before midnight of the day before the last date. For example, if today is June 01, 2003 in EST time, the getToday() method will return 20030602T040000Z. For a day date range, the end date would be 20030603T035900Z.

12

Oracle Calendar Web Services Supported Data Components and Properties

The data format of Oracle Calendar events and tasks are based on *iCalendar DTD Document (xCal)*. This document provides an alternative, XML representation for the standard iCalendar syntax defined in *RFC 2445 - Internet Calendaring and Scheduling Core Object Specification (iCalendar)*.

Note that only vEvents and vTodos are supported in the xCal specification; vJournal and vFreebusy are not.

The following tables and sections describe the iCalendar components and properties that Oracle Calendar Web services supports. They also describe which properties may be modified, added, or deleted with the Modify SOAP command or required, optional, or not allowed with the Create SOAP command. Oracle Calendar Web services also supports Oracle-specific components and properties whose names begin with "x-oracle".

The current implementation of Web services does not support the retrieval of repeating and recurring meetings as a whole. When a Search is performed, any meeting with instances or recurrence rules stored on the server is expanded to separate each instance into an individual meeting. This helps processing and UI generation.

It is important to note that not all xCal elements and properties nor all Oracle Calendar server attributes are supported in this release.

For more information regarding data types, syntax, and other characteristics of iCalendar components and properties, refer to the following documents:

- *RFC 2445 - Internet Calendaring and Scheduling Core Object Specification (iCalendar)*
- *iCalendar DTD Document (xCal)*

Components

Calendar Web services supports the <vevent> and <vtodo> iCalendar components.

The following tables describe the component properties of <vevent> and <vtodo>. The following are clarifications of some of these table's headings and abbreviations:

- Columns labeled "Min." indicate the minimum number of each component property (or property parameter) that the client must create or generate in order to add the specified component to the Calendar store.
- Columns labeled "Max." indicate the maximum number of each component property (or property parameter) the specified component may contain.

- Columns labeled "Mod." indicate whether the component property can be modified or deleted (with the Modify SOAP method).
- Columns labeled "Add" indicate whether the component property can be added (with the Modify SOAP method).
- Columns labeled "Create" indicate whether the component property is required, supported (optional), not supported, or ignored by the Create SOAP method.

vevent

Describes appointments, daily notes, day events, and holidays.

Table 12–1 Component Properties of VEVENT

Component Property	Min	Max.	Mod.	Add	Create	Data Type
attendee	0	n	no	no	not supported	mailto URI of the attendee
class	0	1	yes	no	required	TEXT
description	0	1	yes	yes	supported	TEXT
dtend (must not appear with duration)	1	1	yes	no	not supported	DATE-TIME (default), DATE
dtstart	1	1	yes	no	required	DATE-TIME (default), DATE
duration (must not appear with dtend)	1	1	no	no	required	DURATION
location	0	1	yes	yes	supported	TEXT
organizer	0	1	no	no	not supported	mailto URI of the attendee
priority	0	1	yes	no	supported	INTEGER
summary	0	1	yes	yes	supported	TEXT
uid	0	1	no	no	supported	TEXT
valarm	0	n	yes	yes	supported	valarm
x-oracle-data-guid	0	1	no	no	not supported	TEXT
x-oracle-event-guid	0	1	no	no	ignored	TEXT
x-oracle-eventinstance-guid	0	1	no	no	ignored	TEXT
x-oracle-eventtype	0	1	yes	yes	required	TEXT
x-oracle-isrtcenabled	0	1	yes	yes	supported	BOOLEAN
x-oracle-loginuser	0	1	no	no	supported	mailto URI of the attendee
x-oracle-rtc-attendee-url	0	1	no	no	supported	URI
x-oracle-rtc-dialininfo	0	1	yes	yes	supported	TEXT
x-oracle-rtc-host-url	0	1	no	no	supported	TEXT
x-oracle-rtc-meetingid	0	1	no	no	supported	TEXT
x-oracle-rtc-password	0	1	yes	yes	supported	TEXT
x-oracle-rtc-securitytype	0	1	yes	yes	supported	TEXT
x-oracle-search-relevance	0	1	no	no	not supported	INTEGER

vtodo

Describes tasks stored in the Oracle Calendar server.

Table 12–2 Component Properties of VTODO

Component Property	Min.	Max.	Mod.	Add	Data Type
class	0	1	no	no	TEXT
completed	0	1	no	no	DATE-TYPE
created	0	1	no	no	DATE-TIME
description	0	1	no	no	TEXT
dtstamp	0	1	no	no	DATE-TIME
dtstart	0	1	no	no	DATE-TIME (default), DATE
due (However, must not appear of duration appears)	1	1	no	no	DATE-TIME (default), DATE
last-modified	0	1	no	no	DATE-TIME
percent	0		no	no	INTEGER
priority	0	1	no	no	INTEGER
summary	0	1	no	no	TEXT
uid	0	1	no	no	TEXT
x-oracle-data-guid	0	1	no	no	TEXT
x-oracle-taskid	0	1	no	no	TEXT

valarm

Describes reminders for Calendar entries. Properties of <valarm> include the type of reminder, such as a popup or an email, and the time before which the <valarm> should notify the user of the Calendar event.

Table 12–3 Component Properties of VFREEBUSY

Component Property	Min.	Max.	Mod.	Add	Data Type	Description
action	1	1	yes	no	TEXT	Describes the type of action to be performed by the alarm. This can be one of the following: ▪ AUDIO ▪ DISPLAY (displays a popup) ▪ EMAIL (emails a reminder) ▪ PROCEDURE ▪ X-ORACLE-ALARM-DEFAULT (uses the user's server's default action) ▪ X-ORACLE-ALARM-NONE (does not trigger an alarm) ▪ X-ORACLE-SMS (triggers an SMS)
trigger	1	1	yes	no	TRIGGER	Time and duration of the alarm

vevent and vtodo Component Properties

The following describes the iCalendar and Oracle-specific component properties of the <vevent> and <vtodo> calendar components.

attendee

Defines an attendee within a Calendar component. This component has the same structure as <x-oracle-loginuser> except it may have additional property parameters. The following are all property parameters of <attendee>:

partstat
The partstat property parameter represents the attendee's participation status. It may have a value of ACCEPTED, DECLINED, or NEEDS-ACTION. Note that DELEGATED and TENTATIVE are not supported.

cn
The common name (cn) attribute provides an X.400 representation of the attendee's name and is typically used for display purposes.

The following is an example of an <attendee> component property:

```
<attendee cn="Germaine Lauzon" partstat="ACCEPTED">
  GERMAINE.LAUZON@LES-BELLES-SOEURS.QC.CA
</attendee>
```

class

Defines the access classification for a Calendar component. It may have one of the following values:

- CONFIDENTIAL
- PRIVATE
- PUBLIC

completed

Has a value of "true" if the value of percent is "100".

created

Specifies the date and time that the Calendar component was created. The date and time is a UTC value.

description

Provides a more complete description of the Calendar component than that provided by the <summary> property.

dtend

Specifies the date and time that a Calendar component ends.

The data type of this property may be DATE-TIME or DATE. Specify the data type with the `value` attribute. The following is an example of this property:

```
<dtend value="DATE">20050119</dtstart>
```

dtstamp

Indicates the date/time that the instance of the Calendar object was created or last modified. The value must be specified in the UTC time format. This property is different than the <created> and <last-modified> properties. These two properties are used to specify when the particular Calendar data in the Calendar store was created and last modified. This is different than when the Calendar object representation of the Calendar data was created or modified by the client.

dtstart

Specifies when the Calendar component begins. Non-standard, value data type, time zone identifier property parameters can be specified on this property.

If `dtend` is present, it will be used to calculate the event duration; the actual end time is not stored. As event times are measured in minutes, the start time and duration will have their 'seconds' component set to zero.

The data type of this property may be DATE-TIME or DATE. Specify the data type with the value attribute. The following is an example of this property:

```
<dtstart value="DATE">20050119</dtstart>
```

due

Represents the task due date and time. The date and time is a UTC string of type DATE-TIME or DATE.

duration

Specifies a positive duration of time.

For example, a duration of 15 days, 5 hours and 20 seconds would be represented as P15DT5H0M20S. A duration of 7 weeks would be represented as P7W

last-modified

Specifies the date and time that the information associated with the Calendar component was last revised in the Calendar store. This is analogous to the modification date and time for a file in the file system. The property value must be specified in the UTC time format.

location

Defines the intended venue for the activity defined by a Calendar component.

organizer

Defines the organizer for a Calendar component. Uses the same structure as <x-oracle-loginuser>.

percent

An integer between 0 and 100 that represents the percent completed of a task.

priority

Defines the relative priority for a Calendar component.

The following table describes the possible values of this property:

Table 12–4

Possible Value of <priority>	Description
1	Highest
2	High

Table 12–4 (Cont.)

Possible Value of <priority>	Description
5	Normal
7	Low
9	Lowest

summary

Defines the title of the event or instance.

uid

Defines the persistent, globally unique identifier for the Calendar component.

If <uid> is not specified when storing data, the Oracle Calendar server will assign a value.

url

Defines a Uniform Resource Locator (URL) associated with the Calendar object.

x-oracle-data-guid

Defines the event's or task's unique identifier so that an entry can be identified when referencing it through OCAS.

x-oracle-event-guid

Uniquely identifies <vevent> components.

x-oracle-eventinstance-guid

Uniquely identifies <vevent> instances.

x-oracle-eventtype

Identifies the type of event that the <vevent> represents. The property can be specified once in the <vevent> component. It can have one of the following values:

- APPOINTMENT: Identifies a regular blocking meeting.
- DAILY NOTE: Identifies a non-blocking note associated with a calendar day.
- DAY EVENT: Identifies a non-blocking all day Calendar event.
- HOLIDAY: Identifies a non-blocking holiday specialization of a day event.

x-oracle-isrtcenabled

Indicates that an instance is Web conference-enabled. This property is generated by the Oracle Calendar server and used by Calendar clients. This property can be specified in the `vevent` Calendar component.

x-oracle-loginuser

Defines the login user.

The value of the <x-oracle-loginuser> the mailto URI of the Calendar user. The property parameters of this property appear as element attributes.

The following is an example of an <x-oracle-loginuser> element:

```
<x-oracle-loginuser cn="GERMAINE LAUZON">
  mailto:GERMAINE.LAUZON@LES-BELLES-SOEURS.QC.CA
</x-oracle-loginuser>
```

The cn property parameter is the common or displayable name associated with the login user.

x-oracle-rtc-attendee-url

This property may be specified once in the `vevent` Calendar component.

x-oracle-rtc-dialininfo

Specifies dial-in information required by attendees to join a teleconference, such as the phone number and conference ID. This property may be specified once in the `vevent` Calendar component.

x-oracle-rtc-host-url

Specifies the URL of the Web page hosting the Web conference associated with this Calendar component instance. This value is set by Web conference. This property may be specified once in the `vevent` Calendar component.

x-oracle-rtc-meetingid

Specifies the Web conference ID associated with the `vevent` Calendar component instance. The value is strictly generated by the Web conference server. This property may be specified once in the `vevent` Calendar component.

x-oracle-rtc-password

The optional key (the password) of a Web conference. The property can be specified in the `vevent` Calendar component.

x-oracle-rtc-securitytype

Indicates the security type of a Web conference enabled instance. Possible types are "restricted", "regular", and "public". The property may be specified once in the `vevent` Calendar component. This property may be specified once in the `vevent` Calendar component.

x-oracle-search-relevance

Represents the relative weight of this instance in the Ultra Search result set. This property contains a value between 0 and 100. Instances with higher values are considered more likely to be relevant to the end user. This is only to be used for Ultra Search.

x-oracle-taskid

Represents the task ID of a <vtodo> generated by the Oracle Calendar server.

Example XML Calendar Data

The following are examples of iCalendar components.

Simple Events

```
<vcalendar>
     <vevent>
          <class>CONFIDENTIAL</class>
          <description>a description</desription>
          <dtend>20021101T120000Z</dtend>
          <dtstart>20021101T110000Z</dtstart>
          <location></location>
          <organizer cn="James Baldwin">
               mailto:james.baldwin@oracle.com
          </organizer>
          <priority>1</priority>
          <status>CONFIRMED</status>
          <summary>a meeting</summary>
          <uid>ORACLE:CALSERV:EVENT:48390483290843290</uid>
          <attendee cn="James Baldwin" partstat="ACCEPTED">
               mailto:james.baldwin@oracle.com
          </attendee>
          <x-oracle-eventtype>APPOINTMENT</x-oracle-eventtype>
          <x-oracle-event-guid>fdjskljfdlkj</x-oracle-event-guid>
          <x-oracle-eventinstance-guid>fdjskljfdlkj
          </x-oracle-eventinstance-guid>
          <x-oracle-data-guid>fdjskljfdlkj</x-oracle-data-guid>
     </vevent>
</vcalendar>
```

Repeating and Recurring Events

If a repeating (or recurring) meeting were pulled directly from the Oracle Calendar server, it would have one vCalendar element with multiple, related vEvents. The main vEvent would contain an rrule element outlining the rule for the meeting followed by instance, exception, and time zone information. All event-guids would be the same through the vCalendar, but the instance-guids would be different.

When a Search is performed, any meeting with instances or recurrence rules stored on the server is expanded to separate each instance into an individual meeting. A meeting and its instances have the same <x-oracle-event-guid>.

The following is an example of a repeating event that repeats daily for two days, from January 21 to January 22:

```
<vcalendar version="2.0" prodid="-//Oracle//Calendaring//OCAS//EN">
  <vevent>
    <uid>E1+12345678+1+123456789@I1+00000000+1+4+111111111</uid>
    <transp>OPAQUE</transp>
    <summary>Repeating meeting</summary>
    <status>CONFIRMED</status>
    <priority>5</priority>
    <organizer cn="Germaine Lauzon">
      mailto:GERMAINE.LAUZON@LES-BELLES-SOEURS.QC.CA
    </organizer>
```

```
    <location>Somewhere</location>
    <description> </description>
    <dtstart>20050122T150000Z</dtstart>
    <dtend>20050122T160000Z</dtend>
    <class>PUBLIC</class>
    <attendee cn="Germaine Lauzon" partstat="ACCEPTED">
      GERMAINE.LAUZON@LES-BELLES-SOEURS.QC.CA
    </attendee>
    <x-oracle-data-guid>
      E1+12345678+1+123456789@I1+00000000+1+4+111111111
    </x-oracle-data-guid>
    <x-oracle-areothersinvited>FALSE</x-oracle-areothersinvited>
    <x-oracle-isextrainstance>FALSE</x-oracle-isextrainstance>
    <x-oracle-isexception>FALSE</x-oracle-isexception>
    <x-oracle-loginuser cn="Germaine Lauzon">
      mailto:GERMAINE.LAUZON@LES-BELLES-SOEURS.QC.CA
    </x-oracle-loginuser>
    <x-oracle-eventinstance-guid>
      I1+00000000+1+4+111111111
    </x-oracle-eventinstance-guid>
    <x-oracle-event-guid>E1+12345678+1+123456789</x-oracle-event-guid>
    <x-oracle-eventtype>APPOINTMENT</x-oracle-eventtype>
    <valarm>
      <trigger>-P0DT0H10M0S</trigger>
      <action>DISPLAY</action>
      <description></description>
    </valarm>
    <valarm>
      <trigger>-P0DT0H10M0S</trigger>
      <action>AUDIO</action>
    </valarm>
  </vevent>
</vcalendar>
<vcalendar version="2.0" prodid="-//Oracle//Calendaring//OCAS//EN">
  <vevent>
    <uid>E1+12345678+1+123456789@I1+00000000+1+5+000000000</uid>
    <transp>OPAQUE</transp>
    <summary>Repeating meeting</summary>
    <status>CONFIRMED</status>
    <priority>5</priority>
    <organizer cn="Germaine Lauzon">
      GERMAINE.LAUZON@LES-BELLES-SOEURS.QC.CA</organizer>
    <location>Somewhere</location>
    <description> </description>
    <dtstart>20050121T150000Z</dtstart>
    <dtend>20050121T160000Z</dtend>
    <class>PUBLIC</class>
    <attendee cn="Germaine Lauzon" partstat="ACCEPTED">
      GERMAINE.LAUZON@LES-BELLES-SOEURS.QC.CA
    </attendee>
    <x-oracle-data-guid>
      E1+12345678+1+123456789@I1+00000000+1+5+000000000
    </x-oracle-data-guid>
    <x-oracle-areothersinvited>FALSE</x-oracle-areothersinvited>
    <x-oracle-isextrainstance>FALSE</x-oracle-isextrainstance>
    <x-oracle-isexception>FALSE</x-oracle-isexception>
    <x-oracle-loginuser cn="Germaine Lauzon">
      GERMAINE.LAUZON@LES-BELLES-SOEURS.QC.CA
    </x-oracle-loginuser>
    <x-oracle-eventinstance-guid>
```

```
      I1+00000000+1+5+000000000
    </x-oracle-eventinstance-guid>
    <x-oracle-event-guid>E1+12345678+1+123456789</x-oracle-event-guid>
    <x-oracle-eventtype>APPOINTMENT</x-oracle-eventtype>
    <valarm>
      <trigger>-P0DT0H10M0S</trigger>
      <action>DISPLAY</action>
      <description></description>
    </valarm>
    <valarm>
      <trigger>-P0DT0H10M0S</trigger>
      <action>AUDIO</action>
    </valarm>
  </vevent>
</vcalendar>
```

Daily Notes

```
<vcalendar>
     <vevent>
          <class>CONFIDENTIAL</class>
          <description>a description</description>
          <dtend value="DATE">20021101</dtend>
          <dtstart value="DATE">20021101</dtstart>
          <organizer cn="Par Lagerkvist">
               mailto:par.lagerkvist@oracle.com
          </organizer>
          <priority>3</priority>
          <status>CONFIRMED</status>
          <summary>a daily note</summary>
          <uid>ORACLE:CALSERV:EVENT:49304932-04932-09</uid>
          <attendee cn="Par Lagerkvist" partstat="ACCEPTED">
               mailto:par.lagerkvist@oracle.com
          </attendee>
          <x-oracle-event-guid>fdjskljfdlkj</x-oracle-event-guid>
          <x-oracle-eventtype>DAILY NOTE</x-oracle-eventtype>
          <x-oracle-eventinstance-guid>fdjskljfdlkj
          </x-oracle-eventinstance-guid>
          <x-oracle-data-guid>fdjskljfdlkj</x-oracle-data-guid>
     </vevent>
</vcalendar>
```

Day Events

```
<vcalendar>
     <vevent>
          <class>CONFIDENTIAL</class>
          <description>a description</description>
          <dtend value="DATE">20021101</dtend>
          <dtstart value="DATE">20021101</dtstart>
          <organizer cn="Par Lagerkvist">
               mailto:par.lagerkvist@oracle.com
          </organizer>
          <priority>3</priority>
          <status>CONFIRMED</status>
          <summary>a day event</summary>
          <uid>ORACLE:CALSERV:EVENT:49304932-04932-09</uid>
          <attendee cn="Par Lagerkvist" partstat="ACCEPTED">
               mailto:par.lagerkvist@oracle.com
```

```
          </attendee>
          <x-oracle-event-guid>fdjskljfdlkj</x-oracle-event-guid>
          <x-oracle-eventtype>DAY EVENT</x-oracle-eventtype>
          <x-oracle-eventinstance-guid>fdjskljfdlkj
          </x-oracle-eventinstance-guid>
          <x-oracle-data-guid>fdjskljfdlkj</x-oracle-data-guid>
     </vevent>
</vcalendar>
```

Holidays

```
<vcalendar>
     <vevent>
          <categories>
               <item>Holiday</item>
          </categories>
          <class>PUBLIC</class>
          <description>a description</description>
          <dtend value="DATE">20021031</dtend>
          <dtstart value="DATE">20021031</dtstart>
          <organizer cn="Par Lagerkvist">
               mailto:par.lagerkvist@oracle.com
          </organizer>
          <priority>3</priority>
          <status>CONFIRMED</status>
          <summary>a holiday</summary>
          <uid>ORACLE:CALSERV:EVENT:49304932-04932-09</uid>
          <attendee cn="Par Lagerkvist" partstat="ACCEPTED">
               mailto:par.lagerkvist@oracle.com
          </attendee>
          <x-oracle-event-guid>fdjskljfdlkj</x-oracle-event-guid>
          <x-oracle-eventtype>HOLIDAY</x-oracle-eventtype>
          <x-oracle-eventinstance-guid>fdjskljfdlkj
          </x-oracle-eventinstance-guid>
          <x-oracle-data-guid>fdjskljfdlkj</x-oracle-data-guid>
     </vevent>
</vcalendar>
```

Tasks

```
<vcalendar version="2.0" prodid="-//Oracle//Calendaring//OCAS//EN">
  <vtodo>
    <uid>ORACLE:CALSERV:TASK:469763206/1</uid>
    <summary>My task</summary>
    <priority>5</priority>
    <percent>50</percent>
    <description></description>
    <dtstart>20060424T040000Z</dtstart>
    <due>20060428T040000Z</due>
    <created>20060417T182532Z</created>
    <class>PUBLIC</class>
    <dtstamp>20060417T193833Z</dtstamp>
    <x-oracle-taskid>469763206/1</x-oracle-taskid>
    <x-oracle-data-guid>ORACLE:CALSERV:TASK:469763206/1</x-oracle-data-guid>
  </vtodo>
</vcalendar>
```

13

Oracle Calendar Web Services Status Codes

Each status code in a SOAP fault is made up of four segments; the first describes the source module, the last describes the error type. (The second and third segments are not generally used at this time.) This chapter lists the Module and Error codes that can be displayed in a SOAP fault.

A sample Code tag might look as follows:

```
0020-00-00-00000017
```

By referring to the tables in this chapter, this code represents the following error:

```
Module 0020 = SYS_MODULE_SOAP
Error 00000017 = SecurityError
```

For more details on working with SOAP faults, see "SOAP Faults and Exceptions" in Chapter 11, "Oracle Calendar Web Services Client-Side Java Implementation".

Module Codes

Each of these 64-bit codes corresponds to the source Module of an error in Oracle Calendar. Each Module name is preceded by "SYS_MODULE_".

Table 13–1 Source Module Codes

Code	Module	Description
0x0000	NONE	not applicable
0x0001	UNIAPI	Calender Server
0x0002	APPLICATION	Calendar Applications
0x0003	MEMORYMGR	Memory Manager
0x0004	CONNECTION	Connection Service
0x0005	DISPATCH	Dispatch Service
0x0006	LINKDB	Link Database Service
0x0007	MESSAGECAT	Message Catalogue Service
0x0008	PREFERENCE	Preference Service
0x0009	REGISTRY	Registry Services
0x000A	SESSIONDB	Session Database Service
0x000B	SYSTEM	System Service
0x000C	DATAACCESS	Data Access Service

Table 13–1 (Cont.) Source Module Codes

Code	Module	Description
0x000D	DATAMANGER	Data Manager Service
0x000E	SYNC	Synchronization Service
0x000F	SYNCML	SyncML Module
0x0010	MALCLIENT	MAL Module
0x0011	NLSSERVICE	NLS Service
0x0012	MALSYSTEM	MAL System Service
0x0013	PLUGINCONFIG	Plug-in (component) Service
0x0014	MOBILE	Mobile Module
0x0015	XMLSERVICE	SOAP Module
0x0016	WINDOWS	Windows Error
0x0017	FCGI	FCGI Toolkit Error
0x0020	SOAP	Web Service Module

Error Codes

The following table lists the error codes that can be generated within the 0020 SOAP module.

Table 13–2 Error Codes

Code	Description
00000000	Error encountered with an invalid error code set
00000001	Displayed in the rare case that the object was created with no error code set
00000002	The SOAP request was received but the content was empty
00000003	The SOAP request was received but there was no SOAPAction line in the HTTP header
00000004	The SOAP request was received in another mechanism other than an HTTP POST
00000005	The SOAP request was not sent in UTF-8
00000006	The requested SOAP method is currently not supported
00000007	The requested SOAP package does not have any HTTP content
00000008	The requested SOAP package failed during the processing of the BasicAuth element
00000009	The requested SOAPAction is not recognized by the server
0000000A	The requested SOAPAction did not match the actual method being invoked in the SOAP envelope body
0000000B	The requested SOAPAction did not contain the proper namespace URI (the CWSL namespace)
0000000C	The SOAP envelope namespace was not the SOAP 1.1 namespace
0000000D	The BasicAuth SOAP header does not have the proper namespace
0000000E	No longer used

Table 13–2 (Cont.) Error Codes

Code	Description
0000000F	An unexpected error occurred while generating the response to Search
00000010	An unexpected error occurred while parsing the SOAP message
00000011	Apache Xerces SAX2 parser fatal error message
00000012	Apache Xerces SAX2 parser error message
00000013	Apache Xerces SAX2 parser warning message
00000014	Apache Xerces SAX2 parser exception message
00000015	Apache Xerces XML exception message
00000016	An unexpected error occurred while generating the response to Ping
00000017	An authentication problem (such as a bad user, bad password, or bad security token). The message is explicitly generic; it does not provide any clue as to why the authentication failed.
00000018	No longer used
00000019	No longer used
0000001A	No longer used
0000001B	No longer used
0000001C	No longer used
0000001D	The SOAP Message EncodeQuietLoginInfo (the Search XML element) did not have the proper Calendaring namespace
0000001E	A CmdId was not specified or was blank in the Search method
0000001F	The data store specified in the Search Query's From element is not recognized or supported
00000020	The SOAP header BasicAuth is missing the Name entry
00000021	The SOAP header BasicAuth is missing the Password entry
00000022	The Query element was not found in the Search element of the SOAP request
00000023	An unexpected error occurred while generating a SOAP response
00000024	An unexpected error occurred during the generation of the Create response
00000025	The Create SOAP method has a namespace that doesn't match the predefined one
00000026	The Create SOAP method did not have a Cmd element in the request
00000027	An unexpected error occurred during the generation of the Modify response
00000028	The Modify SOAP method has a namespace that doesn't match the predefined one
00000029	The Modify SOAP method did not have a Cmd element in the request
0000002A	The Query element was not found in the Modify element of the SOAP request
0000002B	The data store specified in the Modify Query's From element is not recognized or supported
0000002C	The Modify SOAP method did not have the element containing the original values in the request

Table 13–2 (Cont.) Error Codes

Code	Description
0000002D	The Modify SOAP method did not have the element containing the modified values in the request
0000002E	An unexpected error occurred during the generation of the Delete response
0000002F	The Delete SOAP method has a namespace that doesn't match the predefined one
00000030	The Delete SOAP method did not have a Cmd element in the request
00000031	The Query element was not found in the Delete element of the SOAP request
00000032	The data store specified in the Delete Query's From element is not recognized or supported
00000033	The Create method did not have a child element to be created
00000034	The Create method did not have a proper XML child element
00000035	An unexpected error occurred during the generation of the Noop response
00000036	The SOAP header is not supported
00000037	The TrustedAuth namespace did not match the internal value
00000038	The TrustedAuth name element is missing
00000039	The TrustedAuth token element is missing
0000003A	An unexpected error occurred during Trusted Authentication
0000003B	The ProxyAuth namespace did not match the internal value
0000003C	The ProxyAuth name element is missing
0000003D	The ProxyAuth application name element is missing
0000003E	The ProxyAuth application password element is missing
0000003F	An unexpected error occurred during Proxy Authentication
00000040	The component cannot be located within the parsed iCalendar information sent from the client
00000041	More than one Event was found in the message body
00000042	An Event create was requested without an x-oracle-eventtype element
00000043	A component create was requested without a class element
00000044	An Event create was requested without a dtstart element
00000045	An Event create was requested without a duration element
00000046	Unable to convert the xCal event to an internal component. This will cause the SOAP method to fail immediately
00000047	The Event class element did not have a recognized or supported value for the operation being performed
00000048	The Event class element did not have a recognized or supported value for the operation being performed
00000049	An unsupported event type property is included in the xCal event
0000004A	The Modify command found an invalid component
0000004B	A Data GUID property was found in the data

Table 13–2 (Cont.) Error Codes

Code	Description
0000004C	An invalid property was found within the transmitted data
0000004D	No longer used
0000004E	No longer used
0000004F	The Modify command does not support adding or removing the class property. This message is returned if an attempt is made to remove or add the class property.
00000050	The Modify command does not support adding or removing the dtstart property. This message is returned if an attempt is made to remove or add the dtstart property.
00000051	The dtend property cannot be modified
00000052	The duration property cannot be modified
00000053	The eventtype property cannot be modified
00000054	The priority property cannot be modified
00000055	The data guid property cannot be modified
00000056	The uid property cannot be modified
00000057	The dtend property is not supported
00000058	The data-guid property is not supported
00000059	The organizer property is not supported
0000005A	The attendee property is not supported
0000005B	The organizer property cannot be modified
0000005C	The attendee property cannot be modified
0000005D	The search score property is not supported
0000005E	The search score property cannot be modified
0000005F	The timestamp element was missing from the search query
00000060	The search handle element is invalid in the search query
00000061	An internal initialization error caused the SOAP element to not be properly registered. The result will be the inability to parse the incoming SOAP message.
00000062	The SOAP element could not be created; the SOAP transaction will fail.
00000063	Generic XML SAX parser error
00000064	An unexpected error occurred during the generation of the Summary response
00000065	The Summary method did not have the proper namespace
00000066	The Summary method did not have a CmdID element
00000067	The Summary method did not have a cVall element
00000068	The Summary method did not have a StartTime element
00000069	The Summary method did not have an EndTime element
0000006A	The Sumamry method contained an unrecognized vCall element
0000006B	The vAttendee could not be found within the event

Table 13–2 (Cont.) Error Codes

Code	Description
0000006C	The vTodo could not be found within the event
0000006D	The Summary command did not have a Timestamp element
0000006E	The DateTime string is invalid
0000006F	The DateTime string represents a date before the supported minimum date (February 1, 1991)
00000070	The DateTime string represents a date after the supported maximum date (November 30, 2027)
00000071	The DateTime string does not have a time component
00000072	The DateTime string is not in UTC
00000073	The ServiceAuth namespace does not match the internal value
00000074	The ServiceAuth name element is missing
00000075	The ServiceAuth token element is missing
00000076	An unexpected error occurred during Service Authentication
00000077	The Web Conference ID property is not supported for this operation
00000078	The Web Conference host URL property is not supported for this operation
00000079	The Web Conference attendee URL property is not supported for this operation
0000007A	The Web Conference does not have a valid summary
0000007B	The Web Conference type property is invalid
0000007C	The Oracle Calendar server is not configured to support Web Conference meetings
0000007D	The Web Conference type property cannot be modified
0000007E	The Web Conference Attendee URL cannot be modified
0000007F	The Modify command does not support the Web conference host URL property
00000080	The Create command was not successful; check your parameters and try again. This is returned when the server call to create a component throws an error, for example, due to duplicate GUID.
00000081	A Web conference was created without a specified security type
00000082	Invalid alarm. Returned when an invalid or inconsistent alarm is set.
00000083	The Percent Complete value is invalid
00000084	Both a Percent Complete and Date Completed value was specified
00000085	An invalid participation status property was included in an xCal event
00000086	An invalid availability status was included in an xCal event
00000087	The Modify command does not support deleting or adding the partstat property
00000088	The Modify command does not support deleting or adding the showasfree property
00000089	The Create command does not allow setting the participation status
0000008A	An unexpected error occurred while generating the FreeBusy SOAP HTTP response

Table 13–2 (Cont.) Error Codes

Code	Description
0000008B	A calendar user was specified without a calendar user type
0000008C	A calendar user was specified with an invalid calendar user type
0000008D	A calendar user type was specified without a calendar user email or name
0000008E	The operation does not support the specification of a target user
0000008F	The UnconfirmedEvents query cVall did not contain any StartTime elements
00000090	The UnconfirmedEvents call did not have an EndTime element. The UnconfirmedEvents query vCall does not contain any EndTime child elements.
00000091	An event attendee does not support setting the participation status
00000092	An event attendee does not support setting the availability Status
00000093	An event attendee was specified without any identification information
00000094	An event attendee was specified with an invalid email address
00000095	An event attendee was specified with an email or a common name that exceeds the maximum size allowed

A

Oracle Connector for Outlook Components and Properties

The following appendix describes Oracle-specific components and properties supported by Oracle Connector for Outlook.

Components of iCalendar

The following tables describe Oracle-specific properties of the iCalendar components VEVENT, VTODO, and VJOURNAL that are supported by Oracle Connector for Outlook. The following tables also describe the Oracle-specific component X-ORACLE-STICKYNOTE.

The following are clarifications of some of these table's headings and abbreviations:

- Columns labeled "Minimum Occurrences" indicate the minimum number of each component property (or property parameter) that the client must create or generate in order to add the specified component to the Calendar store.
- Columns labeled "Maximum Occurrences" indicate the maximum number of each component property (or property parameter) the specified component may contain.
- n: No limit of the maximum number of the specified property or parameter

VEVENT

Describes appointments, daily notes, day events, and holidays.

Table A–1 Component Properties of VEVENT

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
X-ORACLE-DESCRIPTION-COMPRESSEDRTF	0	1	compressed RTF format
X-ORACLE-OBJECTOWNER	0	1	TEXT

VTODO

The VTODO component describes tasks stored in the Oracle Calendar server.

Table A–2 Component Properties of VTODO

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
X-ORACLE-BILLINGINFO	0	1	TEXT
X-ORACLE-DESCRIPTION-COMPRESSEDRTF	0	1	compressed RTF format
X-ORACLE-ESTIMATEDTIME	0	1	TEXT
X-ORACLE-MILELAGE	0	1	TEXT
X-ORACLE-TIMESPENT	0	1	TEXT

X-ORACLE-STICKYNOTE

An Oracle-specific component that describes personal notes that are appended to a Calendar entry.

Table A–3 Component Properties of X-ORACLE-STICKYNOTE

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
CREATED	0	1	DATE-TIME
DESCRIPTION	0	1	TEXT
DTSTAMP	0	1	DATE-TIME
LAST-MODIFIED	0	1	DATE-TIME
UID	0	1	TEXT
X-ORACLE-COLORID	1	1	INTEGER
X-ORACLE-DESCRIPTION-COMPRESSEDRTF	0	1	compressed RTF format
X-ORACLE-GEOMETRY	1	1	four INTEGER values separated by semicolons
X-ORACLE-STICKYNOTE-GUID	0	1	TEXT

VJOURNAL

Describes records and tracking information for important activities of all types.

Table A–4 Component Properties of VJOURNAL

Component Property	Minimum Occurrences	Maximum Occurrences	Data Type
X-ORACLE-DESCRIPTION-COMPRESSEDRTF	0	1	compressed RTF format
X-ORACLE-ENTRYTYPE	1	1	TEXT

VEVENT, VTODO, VALARM, X-ORACLE-STICKYNOTE Component Properties

The following describes Oracle-specific component properties of the VEVENT, VTODO, VALARM, VJOURNAL, and X-ORACLE-STICKYNOTE calendar components.

X-ORACLE-BILLINGINFO

Describes information related to billing for the task described by the VTODO, such as account information. The property can be specified once in the VTODO calendar component. The following is a list of colors and their corresponding ID:

The following is an example of this property:

```
X-ORACLE-BILLINGINFO:J. Smith & Ass.\, $45.00/hour
```

X-ORACLE-COLORID

Represents the display color of a sticky note. The property can be specified once in the X-ORACLE-STICKYNOTE calendar component.

Table A–5 Sticky Node Colors and Color IDs

Color	Color ID
Blue	0
Green	1
Pink	2
Yellow	3
White	4

The following is an example of this property:

```
X-ORACLE-COLORID:2
```

X-ORACLE-DESCRIPTION-COMPRESSEDRTF

Represents the description in compressed RTF format. If users modify this property, they should take care to ensure that the plain-text version in the DESCRIPTION property is also appropriately modified. The property can be specified in the "VEVENT", "VTODO", "VJOURNAL", and "X-ORACLE-STICKYNOTE" calendar components.

The property is defined by the following notation:

```
x-oracle-description-compressedrtf =
                      "X-ORACLE-DESCRIPTION-COMPRESSEDRTF"
                      x-oracle-description-compressedrtfparam
                      ":" binary CRLF

x-oracle-description-compressedrtfparam = 2*(

                      ; the following are REQUIRED,
                      ; but MUST NOT occur more than once

                      (";" valuetypeparam) /
                      (";" encodingparam ) /

                      ; the following are optional,
                      ; but MUST NOT occur more than once

                      (";" languageparam) /
                      (";" x-oracle-internal-checksumparam ) /

                      ; the following is optional,
```

```
                        ; and MAY occur more than once

                        (";" xparam)

                        )
```

The following is an example of this property:

```
X-ORACLE-DESCRIPTION-COMPRESSEDRTF;VALUE=BINARY;ENCODING=BASE64:
```

X-ORACLE-ENTRYTYPE

Specifies the type of activity or item that the journal item is keeping record of.

X-ORACLE-ESTIMATEDTIME

Represents the number of minutes of work allocated to complete the task described by the VTODO.

The following is an example of this property:

```
X-ORACLE-ESTIMATEDTIME:+PT300M
```

X-ORACLE-GEOMETRY

Represents the size and position on screen of the displayed sticky note. The value is four semicolon separated integer values. Each of these integers represent (in sequence) horizontal coordinate of the top left pixel, vertical coordinate of the top left pixel, width of the note, and height of the note. The property may be specified once in the X-ORACLE-STICKYNOTE component.

The following is an example of this property:

```
X-ORACLE-GEOMETRY:556;121;90;83
```

X-ORACLE-MILELAGE

Specifies mileage accrued while working on the task described by the

VTODO, for billing purposes.

The following is an example of this property:

```
X-ORACLE-MILEAGE:137km
```

X-ORACLE-OBJECTOWNER

Describes the organizer of a meeting. This is the user that created the meeting. The property can be specified once in the "VEVENT" calendar.

The following is the definition of this property:

```
x-oracle-objectowner = "X-ORACLE-OBJECTOWNER" x-oracle-objectownerparam ":"
                       text CRLF

x-oracle-objectownerparam = *(

                            ; the following is optional,
                            ; but MUST NOT occur more than once

                            (";" cnparam) /
```

```
                                ; the following is optional,
                                ; and MAY occur more than once

                                (";" xparam)

                                )
```

The following is an example of this property:

```
X-ORACLE-OBJECTOWNER;CN="John Smith":mailto:jsmith@example.com
```

X-ORACLE-STICKYNOTE-GUID

Uniquely identifies X-ORACLE-STICKYNOTE components.

X-ORACLE-TIMESPENT

Specifies the number of hours spent on the task described by the VTODO. (See X-ORACLE-ESTIMATEDTIME).

Parameters of iCalendar Component Properties

The following describes Oracle-specific parameters (that are supported by Oracle Connector for Outlook) of iCalendar component properties.

X-ORACLE-FILENAME

Specifies the file name of an attachment.

The following is an example of this parameter:

```
ATTACH;X-ORACLE-FILENAME=test.txt;ENCODING=BASE64;VALUE=BINARY:
  MIICajCCAdOgAwIBAgICBEUwDQYJKoZIhvcNAQEEBQAwdzELMAkGA1UEBhMCVV
  MxLDAqBgNVBAoTI05ldHNjYXBlIENvbW11bmljYXRpb25zIENVBAoTI05ldHNX
  <...remainder of "BASE64" encoded binary data...>
```

X-ORACLE-INTERNAL-CHECKSUM

Specifies the checksum of the DESCRIPTION property at the time the X-ORACLE-DESCRIPTION-COMPRESSEDRTF property was last modified.

The following is an example of this parameter:

```
X-ORACLE-DESCRIPTION-COMPRESSEDRTF;X-ORACLE-INTERNAL-CHECKSUM=17262384;
  VALUE=BINARY;ENCODING=BASE64:aMIICajCCAdgAwI
  BAgICBEUxLDABgkasdjflwr84rdnBvcmF==
```

X-ORACLE-LABELID

Integer representing the attendee's chosen display label of the event.

The following is an example of this parameter:

```
ATTENDEE;X-ORACLE-LABELID=3:mailto:john.smith@example.com
```

Components of vCard

The following table describes the Oracle-specific components of vCard.

VCARD

Describes business and personal contacts in a Calendar user's address book.

Table A–6 Component Properties of vCard

Component	Minimum Occurrences	Maximum Occurrences	Data Type
X-ORACLE-ACCOUNTID	0	1	TEXT
X-ORACLE-ANNIVERSARY	0	1	Date value
X-ORACLE-ASSISTANTNAME	0	1	TEXT
X-ORACLE-BILLINGINFO	0	1	TEXT
X-ORACLE-CHILDREN	0	n*	TEXT
X-ORACLE-COMPANYNAME-YOMI	0	1	TEXT
X-ORACLE-COMPUTERNETWORK	0	1	TEXT
X-ORACLE-CONTACT	0	n	TEXT
X-ORACLE-CONTACT-EMAIL-SELECTORS	0	1	Single structured text value, separated by an escaped COMMA character (ASCII decimal 44).
X-ORACLE-CONTACT-EXTRA-SELECTORS	0	1	Single structured text value, separated by an escaped COMMA character (ASCII decimal 44).
X-ORACLE-CONTACT-FILESELECTOR	0	1	Integer value
X-ORACLE-CUSTOMERID	0	1	TEXT
X-ORACLE-DISTRIBUTIONLIST-MEMBER	0	1	TEXT values separated by semicolons
X-ORACLE-FIRSTNAME-YOMI	0	1	TEXT
X-ORACLE-FOLLOWUPINFO	0	1	TEXT values separated by semicolons
X-ORACLE-FTPURL	0	1	TEXT
X-ORACLE-GENDER	0	1	TEXT
X-ORACLE-GOVERNMENTID	0	1	TEXT
X-ORACLE-HOBBIES	0	1	TEXT
X-ORACLE-INSTANTMSGADDRESS	0	1	TEXT
X-ORACLE-LANGUAGEINFO	0	1	TEXT
X-ORACLE-LASTNAME-YOMI	0	1	TEXT
X-ORACLE-MANAGERNAME	0	1	TEXT
X-ORACLE-MILEAGE	0	1	TEXT
X-ORACLE-NOTE-COMPRESSEDRTF	0	1	TEXT (Must have an X-ORACLE-INTERNAL-CHECKSUM parameter)
X-ORACLE-OBJECTTYPE	0 (Minimum one occurrence for a distribution list)	1	TEXT
X-ORACLE-ORGANIZATIONID	0	1	TEXT

Table A–6 (Cont.) Component Properties of vCard

Component	Minimum Occurrences	Maximum Occurrences	Data Type
X-ORACLE-REFERREDBY	0	1	TEXT
X-ORACLE-SENSITIVITY	0	1	TEXT
X-ORACLE-SPOUSE	0	1	TEXT
X-ORACLE-USERFIELD1	0	1	TEXT
X-ORACLE-USERFIELD2	0	1	TEXT
X-ORACLE-USERFIELD3	0	1	TEXT
X-ORACLE-USERFIELD4	0	1	TEXT

vCard Component Properties

The following describes the iCalendar and Oracle-specific component properties of vCard.

X-ORACLE-ACCOUNTID

Specifies an account identifier for the object the vCard represents.

X-ORACLE-ANNIVERSARY

Specify the anniversary of the object the vCard represents.

X-ORACLE-ASSISTANTNAME

Specifies the name of the assistant of the object the vCard represents.

X-ORACLE-BILLINGINFO

Specifies information related to billing for the object the vCard represents.

X-ORACLE-CHILDREN

Specifies the children of the object the vCard represents. The free-form format of this type allows for descriptive elements in addition to a list of names.

The following are examples of this property:

```
X-ORACLE-CHILDREN:Jane, John (step-son)

X-ORACLE-CHILDREN:Smith\, Jane, Smith\, John
```

X-ORACLE-COMPANYNAME-YOMI

Specifies Yomi representation of the company name of the object the vCard represents.

The following is an example of this property:

```
X-ORACLE-COMPANYNAME-YOMI:=83J=83^=83J-83i
```

X-ORACLE-COMPUTERNETWORK

Specifies the computer network name of the object the vCard represents.

X-ORACLE-CONTACT

Specifies a contact of the object the vCard represents.

X-ORACLE-CONTACT-EMAIL-SELECTORS

Specifies additional information that Microsoft Outlook preserves for the object the vCard represents.

The following is an example of this property:

```
X-ORACLE-CONTACT-EMAIL-SELECTORS:32896\,329121\,32928
```

X-ORACLE-CONTACT-EXTRA-SELECTORS

Specifies additional information that Microsoft Outlook preserves for the object the vCard represents.

X-ORACLE-CONTACT-FILESELECTOR

Specifies the format in which the "File As" should be displayed in Microsoft Outlook for this vCard.

X-ORACLE-CUSTOMERID

Specifies the assigned customer ID of the object the vCard represents.

X-ORACLE-DISTRIBUTIONLIST-MEMBER

Specifies a member of a distribution list. Not applicable to conventional vCards representing an individual. This structured value type value corresponds, in sequence, to the Address Type, the Address, and Display Name. The text components are separated by the SEMI-COLON character (ASCII decimal 59).

The following table describes the property parameters of X-ORACLE-DISTRIBUTION-MEMBER:

Table A–7 Parameters of X-ORACLE-DISTRUBUTIONLIST-MEMBER

Parameter	Description
X-ORACLE-ADDRESSTYPE	Text value, specifies the address type of a member of a distribution list
X-ORACLE-MEMBERNAME	Text value, specifies the name of a member of a distribution list

The following is an example of this type:

```
X-ORACLE-DISTRIBUTIONLIST-MEMBER;X-ORACLE-MEMBERNAME=John Smith;
  X-ORACLE-ADDRESSTYPE=OCS:john.smith@example.com
```

X-ORACLE-FIRSTNAME-YOMI

Specifies the Yomi representation of the first name of the object the vCard represents.

The following is an example of this property:

```
X-ORACLE-FIRSTNAME-YOMI:=83J=83^=83J-83i
```

X-ORACLE-FOLLOWUPINFO

Specifies a followup note for the object the vCard represents. This structured type value corresponds, in sequence, to the Followup Date and Followup Note.

The following table describes the property parameters of X-ORACLE-DISTRIBUTION-MEMBER:

Table A–8 Parameters of X-ORACLE-FOLLOWUPINFO

Parameter	Description
X-ORACLE-COMPLETIONTIME	DATE-TIME value (iCalendar format, not vCard), specifies the completion time of a followup note
X-ORACLE-COMPLETED	
X-ORACLE-FOLLOWUPDATE	DATE-TIME value (iCalendar format, not vCard), specifies the date for the followup.

The following is an example of this property:

```
X-ORACLE-FOLLOWUPINFO;X-ORACLE-COMPLETIONTIME=20050516T201900Z;
  X-ORACLE-COMPLETED=1;X-ORACLE-FOLLOWUPDATE=20050516T201900Z:
  Follow up note.
```

X-ORACLE-FTPURL

Specifies the FTP URL of the object the vCard represents.

X-ORACLE-GENDER

Specifies the gender of the object the vCard represents.

X-ORACLE-GOVERNMENTID

Specifies the assigned government ID number of the object the vCard represents.

X-ORACLE-HOBBIES

Specifies a list or description of the hobbies of the object the vCard represents.

X-ORACLE-INSTANTMSGADDRESS

Specifies the instant messaging address of the object the vCard represents.

X-ORACLE-LANGUAGEINFO

Specify language-related information about the object the vCard represents.

X-ORACLE-LASTNAME-YOMI

Specifies the Yomi representation of the last name of the object the vCard represents.

X-ORACLE-MANAGERNAME

Specify the name of the manager of the object the vCard represents.

X-ORACLE-MILEAGE

Specify mileage accrued in activities related to the object the vCard represents, for billing purposes.

X-ORACLE-NOTE-COMPRESSEDRTF

Specifies the NOTE type in compressed RTF format. This property must have an X-ORACLE-INTERNAL-CHECKSUM parameter. If users modify this type, they should ensure that the plain-text version in the NOTE type is also appropriately modified to remain in sync.

The following is an example of this type:

```
X-ORACLE-NOTE-COMPRESSEDRTF;X-ORACLE-INTERNAL-CHECKSUM=23452342;
  ENCODING=BASE64:dGhpcyBpcyBub3QgcnRmCg==
```

X-ORACLE-OBJECTTYPE

Specifies the type of object that the vCard represents. Valid values for this type are "PERSON" and "DISTRIBUTION LIST".

X-ORACLE-ORGANIZATIONID

Specifies the assigned organization ID of the object the vCard represents.

X-ORACLE-REFERREDBY

Specifies the referrer of the object the vCard represents. The referrer could be a person or organization.

X-ORACLE-SENSITIVITY

Specifies the sensitivity of the information in the vCard. Valid values for this property are "NORMAL", "PERSONAL", "PRIVATE", and "CONFIDENTIAL".

X-ORACLE-SPOUSE

Specifies the spouse of the object the vCard represents.

X-ORACLE-USERFIELD1

Specifies the contents of the first user-defined field of the object the vCard represents. The semantics of this type are dependent on the user's field definition, and can vary from user to user.

X-ORACLE-USERFIELD2

Specifies the contents of the second user-defined field of the object the vCard represents. The semantics of this type are dependent on the user's field definition, and can vary from user to user.

X-ORACLE-USERFIELD3

Specifies the contents of the third user-defined field of the object the vCard represents. The semantics of this type are dependent on the user's field definition, and can vary from user to user.

X-ORACLE-USERFIELD4

Specifies the contents of the fourth user-defined field of the object the vCard represents. The semantics of this type are dependent on the user's field definition, and can vary from user to user.

Index

A

B

C

D

E

F

I

M

N

P

R

S

T

U

V

W