# sforce Web Services Developer's Guide

Version 2.5

Last Update: January 26, 2004

# Table of Contents

# CHAPTER 1: Getting Started

This topic describes concepts that you need to understand in order to use the sforce Web services API (version 2.5). It contains the following sections:

- Introducing the sforce Web Services API
- Quick Start
- Sample Code Walkthrough

## INTRODUCING THE SFORCE WEB SERVICES API

The sforce Web services API programmatic access to your organization’s salesforce.com information using a simple, powerful, and secure application programming interface (API).

### Customize, Integrate, and Extend Your salesforce.com Solutions

The sforce platform allows you to customize, integrate, and extend your organization's salesforce.com data using the language and platform of your choice.

- **Customize salesforce.com** with custom fields, layouts, and Web integration links to meet specific business requirements.
- **Integrate salesforce.com** with your organization’s ERP and finance systems, deliver real-time sales and support information to company portals, and populate critical business systems with customer information.
- **Extend salesforce.com** in presentation, business logic, and data services with new functionality that reflects the business requirements of your organization.

For more information about sforce solutions, developer resources, and community resources, go to http://www.sforce.com.

### Supported Operations

Using your favorite Web service enabled development environment, you can construct Web service client applications that use standard Web service protocols to programmatically:

- log into the sforce server (login call)
- query your organization’s information (query, queryMore, and retrieve calls)
- create, update, and delete data (create, update, and delete calls)
- perform various administrative tasks, such as retrieving user information (getUserInfo call), changing passwords (setPassword and resetPassword calls), and getting the server’s system time (getServerTimestamp call).
- obtain and navigate metadata about your organization’s data (describeGlobal and describeSObject calls)

For each operation, client applications submit a synchronous request to the sforce API server, await the server’s response, and process the results. The sforce API server commits any changed data automatically. For detailed information about supported Web service operations, see sforce API Calls on page 24 and sforce Utility API Calls on page 53.

### sforce Objects

The sforce Web services API interacts with your organization’s data via *objects*, which are programmatic representations of your organization’s data entities stored on the salesforce.com server. *Object properties* represent fields in those data entities, and client applications set or retrieve data values via these properties. For example, accounts are represented by an Account object, and an Account object has fields that represent the account name, phone number, shipping address, and so on. This document describes how to perform query, insert, update, and delete operations on salesforce.com data via the sforce objects. For detailed information about sforce objects, see sforce Objects on page 59.

### Supported salesforce.com Editions

To use the sforce Web services API, your organization must use salesforce.com Enterprise Edition. If you are an existing salesforce.com customer and want to upgrade to Enterprise Edition, contact your account representative.

To develop Web service client applications, it is strongly recommended that you use salesforce.com Developer Edition. Developer Edition provides access to *all* of the features available with Enterprise Edition—it is constrained only by the number of users and the amount of storage space. Developer Edition provides a development context that allows you to build and test your solutions without impacting your organization's live data. Developer Edition accounts are available for free at http://www.sforce.com.

### Standards Compliance and Compatible Development Platforms

The sforce Web services API is implemented to comply with SOAP 1.1 (Simple Object Access Protocol), WSDL 1.1 (Web Service Description Language), and WS-I (Web Services Interoperability) Basic Profile specifications. The sforce Web services API works with modern SOAP development environments, including, but not limited to, Visual Studio .NET 2003 and Apache Axis. In this document, we provide Java (Axis) and C# (.NET) examples. To see a complete list of compatible development platforms and more sample code, go to http://www.sforce.com.

*Note*

Development platforms vary in their SOAP implementations. Implementation differences in certain development platforms might prevent access to some or all of the features in the sforce Web services API.

If you are using Visual Studio for .NET development, we recommend that you use Visual Studio 2003 or higher.

### Go to sforce.com For More Information

The sforce.com Web site provides a full suite of developer toolkits, sample code, community-based support, and other resources to help you with your development projects. Be sure to visit http://www.sforce.com and sign up for a free Developer Edition account.

## QUICK START

This topic tells you what you need to start using the sforce API in your development environment. It includes the following steps:

- Step 1: Obtain a salesforce.com Developer Edition Account
- Step 2: Generate or Obtain the sforce Web services WSDL For Your Organization
- Step 3: Import the WSDL File Into Your Development Platform
- Step 4: Walk Through the Sample Code

*Note*

Before you begin building client applications, you need to deploy your development platform according to its product documentation.

## Step 1: Obtain a salesforce.com Developer Edition Account

If you are not already a member of the sforce developer community, you need to go to http://www.sforce.com and following the instructions for signing up for a Developer Edition account. Even if you already have an Enterprise Edition account, it is strongly recommended that you use Developer Edition for developing, staging, and testing your solutions against sample data to avoid impacting your organization's live data. This is especially true for applications that will be inserting, updating, or deleting data (as opposed to simply reading data).

## Step 2: Generate or Obtain the sforce Web services WSDL For Your Organization

To access sforce Web services, you need an sforce Web Service Description Language (WSDL) file. The WSDL file defines the Web services that are available to you. Your development platform uses this WSDL to generate an API to access the sforce Web services it defines. You can either obtain the WSDL file from your organization's salesforce.com administrator or you can generate it yourself if you have access to Administration Setup in the salesforce.com user interface. For more information about WSDL, see http://www.w3.org/TR/wsdl.

### API Options and Associated WSDL Files

There are two options for sforce Web services APIs:

- **sforce Enterprise Web services API** (enterprise.wsdl)—For most enterprise users who are developing client applications for their organization. The enterprise.wsdl file is a strongly typed representation of your organization's data. It provides information about your schema, data types, and fields to your development environment, allowing for a tighter integration between it and the sforce Web service. As such, this WSDL changes if custom fields or custom objects are added to your organization's salesforce.com configuration.
- **sforce Partner Web services API** (partner.wsdl)—For salesforce.com partners who are developing client applications for multiple organizations. As a loosely typed representation of the salesforce.com object model, it can be used to access data within any organization. It is more flexible, although not as easy to use, as its Enterprise counterpart. For more information, see sforce Partner Web Services API on page 104.

### Generating the WSDL File for Your Organization

The WSDL file is dynamically generated based on:

- which type of WSDL file you download
- the permissions defined in the profile associated with the user who generates the WSDL file

The generated WSDL will define only those objects and fields that are accessible to the user who generates the WSDL file.

To generate the WSDL file for your organization:

1. Log in to salesforce.com using an account that has system administration access.
2. Choose **Setup** | **Sforce Application Server**.
3. Click **Sforce WSDL Generator**.
4. Click one of the following links:
    - Download Enterprise WSDL
    - Download Partner WSDL

   Your browser opens the WSDL file.
5. Save the WSDL file to your local file system, if necessary.

*NOTE* For the enterprise.wsdl file, if new custom fields or objects are added to your organization's information, you need to regenerate the WSDL file in order to access them.

## Step 3: Import the WSDL File Into Your Development Platform

Once you have the WSDL file, you need to import it into your development platform so that your development environment can generate the necessary objects for use in building client Web service applications in that environment. This section provides sample instructions for Apache Axis. For instructions about other development platforms, see your platform's product documentation.

### Instructions for Java Environments (Apache Axis)

Java environments access the sforce Web services API through local based Java objects that serve as proxies for their server-side counterparts. Before using the sforce Web services API, you must first generate these objects from your organization's WSDL file.

Each SOAP client has its own tool for this process. For Apache Axis, you use the WSDL2Java utility. For more information about using WSDL2Java, see the following URL:

http://ws.apache.org/axis/java/reference.html

*Note* Before you run WSDL2Java, you must have Axis installed on your system and all of its component JAR files must be referenced in your classpath.

The basic syntax for WSDL2Java is:

```
java -classpath=pathToFirstJAR/FirstJARFilename;pathToSecondJAR/SecondJARFilename
org.apache.axis.wsdl.WSDL2Java pathToWsdl/WsdlFilename
```

For sforce WSDL files, specific switches are recommended to configure WSDL2Java to use SOAP 1.2 encoding and to generate unreferenced object proxies. The following sample command uses these switches:

```
java -classpath=pathToFirstJAR/FirstJARFilename;pathToSecondJAR/SecondJARFilename
org.apache.axis.wsdl.WSDL2Java -a -T 1.2 pathToWsdl/WsdlFilename
```

This command will generate a set of folders and Java source code files in the same directory in which it was run. After these files are compiled, they can be included in your Java programs for use in creating client applications.

For most Java development environments, you can use Wizard-based tools for this process instead of the command line. For more information about using WSDL2Java with sforce, visit the message boards at sforce.com.

## Step 4: Walk Through the Sample Code

Once you have imported your WSDL file, you can begin building client applications that use the sforce Web services API. The fastest way is to learn by example—start by walking through the code example described in Sample Code Walkthrough on page 4.

## Sample Code Walkthrough

This section walks through a sample Java client application for Apache Axis. The purpose of this sample application is to show the required steps for logging into the sforce single sign-on server and to demonstrate the invocation and subsequent handling of several sforce API calls. This sample application performs the following main tasks:

1. Prompts the user for their salesforce.com user name and password.
2. Calls login to log in to the sforce single login server and, if the login succeeds:
   - Sets the returned `sessionID` into the session header, which is required for session authentication on subsequent API calls.
   - Resets the sforce API server endpoint to the returned `serverUrl`, which is the sforce API server that will be the target of subsequent API calls.

   All client applications that access the sforce Web services API *must* complete the tasks in this step before attempting any subsequent API calls.

3. Calls getServerTimestamp to get the current timestamp on the sforce API server.
4. Calls getUserInfo to get information about the currently logged in user.
5. Calls query with the following `queryString`:

   `"select id, Website, Name from Account where Name >= 'United Oil & Gas Corp.'"`

   This query string attempts to retrieve the ID, web site URL, and name from the Account object in which the Name matches the string `"United Oil & Gas Corp."`
6. Changes the web site address and calls update to update the account data in the salesforce.com data.

In the following sample code, sforce API calls and other significant code is identified in a **bold** font. In addition, note the error handling code that follows each API call.

```
package com.salesforce.Docsamples;

import java.rmi.*;
import java.io.*;
import java.net.*;
import javax.xml.rpc.*;
import java.util.*;
import com.sforce.soap.enterprise.*;
import com.sforce.soap.enterprise.sobject.*;
import com.sforce.soap.enterprise.fault.*;

/**
 * <p>Title: sforce Login Sample</p>
 * <p>Description: Console application illustrating login, session management,</p>
 * <p>and server redirection.</p>
 * <p>Copyright: Copyright (c) 2003</p>
 * <p>Company: salesforce.com</p>
 * @author Dave Carroll
 * @version 4.0
 */

public class Samples {

  private SoapBindingStub binding;
  private LoginResult loginResult = null;
  private String un = "";
  private String pw = "";
  private boolean loggedIn = false;
  private GetUserInfoResult userInfo = null;
  private ID accountID = null;
  String loginURL = "http://aspen.salesforce.com/services/Soap/c/2.5";

  static BufferedReader rdr = new BufferedReader(new java.io.InputStreamReader(
      System.in));

  public Samples() {
  }

  public static void main(String[] args) {
    Samples samples1 = new Samples();
    samples1.run();
  }

  String getUserInput(String prompt) {
    System.out.println(prompt);
    try {
      return rdr.readLine();
    }
    catch (IOException ex) {
      return null;
    }
  }

  private boolean login() {

    // Prompt the user to provide salesforce.com login credentials.
    un = getUserInput("Enter user name: ");
    if (un == null) {
      return false;
    }
    pw = getUserInput("Enter password: ");
    if (pw == null) {
      return false;
    }

    System.out.println("Creating the binding to the web service...");

    try {
        binding = (SoapBindingStub)
            new SforceServiceLocator().getSoap(new URL(loginURL));
    }
    catch (MalformedURLException ex4) {
    }
    catch (ServiceException jre) {
      if (jre.getLinkedCause() != null) {
        jre.getLinkedCause().printStackTrace();
      }
      System.out.println("ERROR: creating binding to soap service, error was: \n"
+ jre.getMessage());
      System.out.println("Press the RETURN key to continue...");
      return false;
    }

    // Time out after a minute
    binding.setTimeout(60000);

    // Attempt to log in to the sforce single sign-on server
    // by invoking the login call.
    try {
      System.out.println("LOGGING IN NOW....");
      loginResult = binding.login(un, pw);
    }
    catch (RemoteException ex) {
      if (ex.getMessage().equals("user not valid")){
        System.out.println("Please make sure that you have a valid user id and
password. \nYou can get a user name and password by signing up at \nhttps://
www.sforce.com/us/orderEntry/signup.jsp.");
        getUserInput();
      }
      System.out.println("\nFailed to successfully login, error message was: \n" +
ex.getMessage());
      System.out.println("\nPress the RETURN key to continue...");
      String wex = getUserInput();
      if (wex.equals("v")) { System.out.println("New url: ");
      loginURL = getUserInput(); }
      return false;
    }
    // Display the returned session ID and URL to the sforce API server
    System.out.println("The session id is: " + loginResult.getSessionId());
    System.out.println("The new server url is: " + loginResult.getServerUrl());

      // Change the binding to the new sforce API server endpoint.
      // Required before invoking subsequent sforce API calls.
      // If the url is null, do not reset.
      if (loginResult.getServerUrl() != null) {
      try {
        binding = (SoapBindingStub)
            new SforceServiceLocator().getSoap(new java.net.
            URL(loginResult.getServerUrl()));
      }
      catch (MalformedURLException ex1) {
        System.out.print("\nFailed to reset the service endpoint, error message
was: \n" + ex1.getMessage());
        System.out.print("\nPress the RETURN key to continue...");
        getUserInput();
      }
      catch (javax.xml.rpc.ServiceException jre) {
        if (jre.getLinkedCause() != null) {
          jre.getLinkedCause().printStackTrace();
        }
       System.out.print("\nFailed to re System.out.print(et the service endpoint,
error message was: \n" + jre.getMessage());
        System.out.print("\nPress the RETURN key to continue...");
        getUserInput();
      }
    }

    // Instantiate a session header for this server session
    // and set the returned session ID into the session header.
    // Required for session validation in subsequent API calls.
    _SessionHeader sh = new _SessionHeader();
    sh.setSessionId(loginResult.getSessionId());
    binding.setHeader("SoapService", "SessionHeader", sh);
    System.out.println("Getting server's timestamp...");

    // Invoke the getServerTimestamp call to obtain the current server timestamp.
    try {
      System.out.println("Time stamp on server: " +
binding.getServerTimestamp(null).getTimestamp().getTime().toGMTString());
    }
    catch (RemoteException ex2) {
     System.out.println("ERROR: getting server timestamp.\n" + ex2.getMessage());
    }

    // Invoke the getUserInfo call to obtain information about the logged on user.
    try {
      System.out.println("Getting user info...");
      userInfo = binding.getUserInfo(un);
      System.out.println("User Name: " + userInfo.getUserFullName());
      System.out.println("User Email: " + userInfo.getUserEmail());
      System.out.println("User Language: " + userInfo.getUserLanguage());
      System.out.println("User Organization: " + userInfo.getOrganizationName());
    }
    catch (RemoteException ex3) {
      System.out.println("ERROR: getting user info.\n" + ex3.getMessage());
    }
```

```
      loggedIn = true;
      return true;

    }

    private void querySample() {
      if (!loggedIn) {
        if (!login()) return;
      }

      QueryResult qr = null;

      // Normally not recommend change the batch size,
      // but provided to demonstrate QueryMore
      _QueryOptions qo = new _QueryOptions();
      qo.setBatchSize(new Integer(3));
      binding.setHeader("SoapService", "QueryOptions", qo);

      // Specify a query string and invoke the query call.
      try {
        Account account = null;
        qr = binding.query(
            "select id, Website, Name from Account where Name >= 'United Oil & Gas
Corp.'");
        if (qr != null) {
          account = ((Account)qr.getRecords()[0]);
          System.out.println("Retrieved the account using Name = 'United Oil & Gas
Corp.*', ID = " + account.getId().getValue() + ", website = " +
account.getWebsite());
        }
        System.out.println("\nQuery succesfully executed.");
        System.out.println("\nPress the RETURN key to continue...");

        // Change the Website address and update the Account
        // by invoking the update call.
        account.setWebsite(account.getWebsite() + ",u");
        SaveResult[] saveResults = binding.update(new SObject[] {account});
      }
      catch (RemoteException ex) {
        System.out.println("\nFailed to execute query succesfully, error message
was: \n" + ex.getMessage());
        System.out.println("\nPress the RETURN key to continue...");
        getUserInput();
      }

    }

    private String getUserInput() {
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      try {
        return br.readLine();
      }
      catch (IOException ex) {
        return null;
      }
    }
    private void run() {
      if (login()) {
        this.querySample();
```

```
            }
        }
    }
```

# CHAPTER 2: Concepts

This topic describes concepts that you need to understand in order to use the sforce API. It contains the following sections:

- Basic Concepts
- Primitive Data Types in the sforce API
- Field Types in the sforce API
- Security in the sforce API
- sforce Object Query Language (SOQL)
- Advanced Topics

## BASIC CONCEPTS

This topic describes the basic concepts you need to understand before building client applications that use the sforce Web services API. It includes the following topics:

- sforce API Calls
- sforce API Objects
- Security in the sforce API

### sforce API Calls

The sforce API calls, summarized in the following table, represent specific operations that your client applications can invoke at run-time to perform certain tasks. All sforce API calls are synchronous requests made to the sforce API server. Any changes to your salesforce.com data are committed automatically. For more information, see About sforce API Calls on page 24.

**Table 1: sforce API Calls**

| Task / Call | Description |
|---|---|
| **Login / Client Session** | |
| login | Logs in to the sforce single sign-on server and starts a client session. |
| **Querying Data** | |
| query | Executes a query against the specified object and returns data that matches the specified criteria. |
| queryMore | Retrieves the next batch of objects from a query. |
| retrieve | Retrieves one or more objects based on the specified object IDs. |
| **Modifying Data** | |
| create | Adds one or more new individual objects to your organization's data. |
| update | Updates one or more existing objects in your organization's data. |

Table 1: sforce API Calls (Continued)

| Task / Call | Description |
|---|---|
| delete | Deletes one or more individual objects from your organization's data. |
| **Object Metadata** | |
| describeGlobal | Retrieves a list of available object types for your organization's data. |
| describeSObject | Retrieves the next batch of objects from a query. |
| **Utilities** | For details, see sforce Utility API Calls on page 53. |
| getServerTimestamp | Retrieves the current system timestamp from the sforce API server. |
| getUserInfo | Retrieves personal information for the user associated with the current session. |
| resetPassword | Changes a user's password to a server-generated value. |
| setPassword | Sets the specified user's password to the specified value. |

## sforce API Objects

In the sforce API, *objects* are data entities that represent your organization's information. The sforce Web services API interacts with your organization's data via *objects*, which are programmatic representations of your organization's data entities stored on the salesforce.com server. *Object properties* represent fields in those data entities, and client applications set or retrieve data values via these properties. For more information, see About sforce Objects on page 59.

## Security in the sforce API

Client applications that access your organization's sensitive salesforce.com data are subject to the same security protections that are used in the salesforce.com user interface.

### User Authentication

Client applications must log in using valid credentials for a salesforce.com account. The sforce single sign-on server authenticates these credentials and, if valid, provides the client application with:

- a sessionID that must be set into the session head so that all subsequent API calls are authenticated
- an URL for the sforce API server that will service the client application's Web service requests

### Profile Configuration

Permissions to access data and invoke API calls are governed by the profile associated with the salesforce.com account under which the client application logs in. The organization's salesforce.com administer controls access by configuring the profile settings and assigning users to that profile (**Setup** | **Manage Users** | **Profiles**). Client applications can query or update only those objects and fields to which they have appropriate access.

### Sharing

In the salesforce.com user interface, the concept of *sharing* refers to the act of granting access to a user or group to view and/or edit a record owned by another user, if the default organization access levels do not otherwise permit such access.

### Implicit Restrictions for Objects and Fields

Certain sforce objects can be created or deleted only in the salesforce.com user interface. Other sforce objects are read-only—client applications cannot create, update, or delete such objects. Similarly, certain fields within some sforce objects can be specified on create but not on update. Other fields are read-only—client applications cannot specify field values in create or update calls. For more information, see the object description in sforce Objects on page 59.

## PRIMITIVE DATA TYPES IN THE SFORCE API

The sforce API uses the following primitive data types:

**Table 2: Primitive Data Types Used in the sforce API**

| Value | Description |
|---|---|
| **`xsd:base64Binary`** | Base 64-encoded binary data. |
| **`xsd:boolean`** | Boolean (True / false) values. |
| **`xsd:date`** | Date values. |
| **`xsd:dateTime`** | Date/time values (timestamps). |
| **`xsd:double`** | Double values. |
| **`xsd:int`** | Integer values. |
| **`xsd:string`** | Character strings. |

These data types are used in the SOAP messages that are exchanged between your client application and the sforce API server. When writing your client application, you simply follow the data typing rules defined for your programming language and development environment. Your development tool handles the mapping of typed data in your programming language with these SOAP data types.

The primitive data types are:

- specified in the World Wide Web Consortium's publication *XML Schema Part 2: Datatypes* at the following URL: http://www.w3.org/TR/xmlschema-2/
- enumerated in the `soapType` field of the Field type, which is described in the `fields` property of the DescribeSObjectResult.

Primitive types are used as a standardized way to define, send, receive, and interpret basic data types in the SOAP messages exchanged between client applications and the sforce API server. In addition, primitive data types are interpreted in a salesforce.com-specific way, which is useful for display formatting and for numeric conversion (adding values of different currencies).

For example, salesforce.com chooses to interpret a double value passed via SOAP (as an **`xsd:double`**) in a number of possible ways, depending on the field definition. If the field type for that data is currency, salesforce.com handles the display of the data by prepending it with a currency symbol and inserting a decimal for precision. Similarly, if the field type is percent, salesforce.com handles the display of the data by appending a percent sign (%). Regardless of the field type, however, the value is sent in the SOAP message as a double.

## FIELD TYPES IN THE SFORCE API

The sforce API uses the following field types:

Table 3: Field Types Used in the sforce API

| Field Type | What the Field Contains |
|---|---|
| **`string`** | String values. |
| **`boolean`** | Boolean (True / False) values. |
| **`int`** | Integer (int) values. |
| **`double`** | Double values. |
| **`date`** | Date values. |
| **`datetime`** | Date and time values. |
| **`base64`** | Base64-encoded arbitrary binary data (of type base64Binary). Used for Attachment, Document, and Scontrol objects. |
| **`id`** | Primary key field for the object. |
| **`reference`** | Cross-references to a different sforce object. Analogous to a foreign key field in SQL. |
| **`currency`** | Currency values. |
| **`textarea`** | String that is displayed as a multi-line text field. |
| **`percent`** | Percentage values. |
| **`phone`** | Phone numbers. Values can include alphabetic characters. Client applications are responsible for phone number formatting. |
| **`url`** | URL values. Client applications should commonly display these as hyperlinks. |
| **`email`** | Email addresses. |
| **`combobox`** | Combobox, which include a set of enumerated values and allow the user to specify a value not in the list. |
| **`picklist`** | Picklists, which include a set of enumerated values. |

These field types extend the primitive data types, which are described in Primitive Data Types in the sforce API on page 12. While many of these field types follow common data typing conventions that are made explicit in their metadata, certain field types have unique characteristics that you need to understand before using them in your client application.

These field types apply to both predefined fields and custom fields. They are enumerated in the `type` field of the Field type, which is described in the `fields` property of the DescribeSObjectResult.

*NOTE*

Some numeric fields have precision and scale limits. In addition, certain text fields have length restrictions. These restrictions are enforced when inserting or updating data. However, the sforce API may return data that does not meet these restrictions.

## String Field Type

String fields (string) contain text and some have length restrictions depending on the data being stored. For example, in the Contact object, the **`FirstName`** field is 40 characters, the **`LastName`** field is 80 characters, the **`MailingStreet`** is 255 characters.

## Boolean Field Type

Boolean (boolean) fields have either of two values:

- True (or 1)
- False (or 0)

## Int Field Type

Integer fields (int) are numbers that contain no fractional portion (digits to the right of a decimal place), such as the **NumberOfEmployees** in an Account. For integer fields, the `digits` field specifies the maximum number of digits that an integer can have.

*NOTE* The limits for integer fields are enforced when you create or update objects. However, the sforce API might return data that does not meet these restrictions.

## Double Field Type

Double fields (double) can contain fractional portions (digits to the right of the decimal place), such as **`ConversionRate`** in CurrencyType. In the sforce API, all non-integer values (such as Currency Field Type and Percent Field Type) are of type double. For double fields, the following restrictions might exist:

Table 4: Limitations on Double Fields

| Fields | Description |
|---|---|
| **`scale`** | Maximum number of digits to the right of the decimal place. |
| **`precision`** | Total number of digits, including those to the left and the right of the decimal place |

The maximum number of digits to the left of the decimal place is equal to **`precision`** minus **`scale`**. In the salesforce.com user interface, precision is defined differently—it is the maximum number of digits allowed to the left of the decimal place.

*NOTE* The limits for double fields are enforced when you create or update objects. However, the sforce API might return data that does not meet these restrictions.

## Date Field Type

Date fields (date) contain date values, such as **ActivityDate** in the Event object. Unlike dateTime fields, date fields contain no time value—the time portion of a date field is not relevant and is always set to midnight in the GMT/UTC time zone.

## DateTime Field Type

DateTime (dateTime) fields handle date/time values (timestamps), such as **ActivityDateTime** in the Event object or the **`CreatedDate`**, **`LastModifiedDate`**, or **`SystemModstamp`** in many sforce objects. Regular Date/Time fields are full timestamps with a precision of one second. They are always transferred in the GMT/UTC time zone. In your client application, you might need to translate the timestamp to or from a local time zone.

*NOTE* The Event object has a **`DurationInMinutes`** field that specifies the number of minutes for an event. Even though this is a temporal value, it is an integer type—not a dateTime type.

## Base64 Field Type

Base64 fields contain base64-encoded arbitrary binary data (of type base64Binary). These fields are used for storing binary files in Attachments, Documents, and Scontrol objects. In these objects, the `Body` or `Binary` field contains the (base64 encoded) data, while the **BodyLength** field defines the length of the data in the `Body` or `Binary` field. In the Document object, you can specify an URL to the document instead of storing the document directly in the record.

## Id Field Type

All object types have an `Id` field (of type ID) that uniquely identifies each record in that type. This is analogous to the concept of a primary key in relational databases.

The value in the `Id` field is assigned by the sforce API server when the record is originally created (create call) to ensure that it is globally unique. This value remains unchanged over the entire lifetime of the record. Some API calls, such as retrieve and delete, accept an array of `Id`s as parameters—each array element uniquely identifies the row to retrieve or delete. Similarly, the update call accepts an array of sObjects—each sObject contains an `Id` field that uniquely identifies the sObject.

## Reference Field Type

A reference field contains an `Id` value that points to a unique record (usually the parent record) on another object. This is analogous to the concept of a foreign key in relational database. For example, in the OpportunityCompetitor object, the `OpportunityId` field is a reference field that points to the Opportunity object. It contains an `Id` value that uniquely identifies an Opportunity record.

In some cases, an object can refer to another object of its same type. For example, accounts have a parent link that can point to another account.

You can also describe and query each cross-referenced object. When you query a cross-reference ID field, it returns an object ID of the appropriate type. You can then query that ID to get additional information about the object, using the ID in the `id` field for that query. The cross-reference ID field value is either a valid record in your organization, or an empty value, which indicates an empty reference.

The cross-reference ID field value, if non-empty, is guaranteed to be an object in your organization. However, it is not guaranteed that you can query that object. Users with the “Manage All Data” permission can always query that object. Other users may be restricted from viewing or editing the referenced object.

When specifying a value for a cross-reference ID field in a create or update call, the value must be a valid value of type ID, and the user must have appropriate access to that object. The exact requirements vary from field to field.

## Currency Field Type

Currency fields contain currency values, such as the `ExpectedRevenue` field in a Campaign, and are defined as type double.

For organizations that have the multi-currency option enabled, the `CurrencyISOCode` field is defined for any object that can have currency fields. The `CurrencyISOCode` field and currency fields are linked in a special way. On any specific record, the `CurrencyISOCode` field defines the currency of that record, and thus, the values of all currency fields on that record will be expressed in that currency.

For most cases, clients do not need to consider the linking of the `CurrencyISOCode` field and the currency fields on an object. However, clients may need to consider the following:

- The `CurrencyISOCode` field exists only for those organizations that have enabled multi-currency support.
- When displaying the currency values in a user interface, it is preferred to prepend each currency value with its `CurrencyISOCode` value and a space separator.

- The **CurrencyISOCode** field is a restricted picklist field. It has certain defined values that can vary from organization to organization, and it can only be set to those fields. Attempting to set it to a value that is not defined for an organization causes the operation to be rejected.
- If you update the **CurrencyISOCode** field on an object, it implicitly converts all currency values on that object to the new currency code, using the conversion rates that are defined for that organization in the salesforce.com user interface. If you specify currency values in that same update call, the new currency values you specify are interpreted in the new **CurrencyISOCode** field value, without conversion.

## Textarea Field Type

Textarea fields contain text that can be longer than 4000 bytes. Unlike string fields, textarea fields cannot be specified in the WHERE clause of a `queryString` of a query call. To filter records on this field, you must do so while processing records in the QueryResult. For fields with this restriction, its **filterable** field in the Field type (described in the `fields` property of the DescribeSObjectResult) is False.

## Percent Field Type

Percent fields contain percent values. Percent fields are defined as type double.

## Phone Field Type

Phone fields contain phone numbers, which can include alphabetic characters. Client applications are responsible for phone number formatting.

## URL Field Type

URL fields contain URLs. Client applications are responsible for specifying valid and properly formatted URLs in create and update calls.

## Email Field Type

Email fields contain email addresses. Client applications are responsible for specifying valid and properly formatted email addresses in create and update calls.

## Picklist Field Type

Picklist fields contain a list of one or more items from which a user choose a single item. One of the items can be configured as the default item.

In the Field object associated with the DescribeSObjectResult, the `restrictedPicklist` field defines whether it is a restricted picklist or not (unrestricted picklists are advisory only). The sforce API does not enforce the list of values for advisory (unrestricted) picklist fields on create or update.

In the Field object associated with the DescribeSObjectResult, the `picklistValues` field contains an array of items (PickListEntry objects). Each PickListEntry defines the item's label, value, and whether it is the default item in the picklist (a picklist no more than one default value).

Enumerated fields support localization of the values to the language of the user. For example, for the **ForecastCategory** field on an Opportunity, the value "Omitted" may be translated to various languages. The enumerated field values are fixed and do not change with a user's language. However, each value may have a specified "label" field that provides the localized label for that value. You must always use the value when inserting or updating a field. The query call always returns the value, not the label. The corresponding label for a value in the DescribeSObjectResult should be used when displaying the value to the user in any user interface.

### Combobox Field Type

A combobox is a picklist that also allows users to type a value that is not already specified in the list. A combobox is defined as a string value.

## SFORCE OBJECT QUERY LANGUAGE (SOQL)

You use the sforce Object Query Language (SOQL) to construct simple but powerful query strings for the `queryString` parameter in the query call. Similar to the SELECT command in SQL, SOQL allows you to specify the source object (such as Account), a list of fields to retrieve, and conditions for selecting rows in the source object.

Note that SOQL does not support all advanced features of the SQL SELECT command. For example, you cannot use SOQL to perform join operations, use wildcards in field lists, use calculation expressions, or specify an ORDERBY clause to sort rows in the result set.

### SOQL Syntax

SOQL uses the following syntax:

```
select fieldList from objectType [where conditionExpression]
```

where:

| Syntax | Description |
|---|---|
| *fieldList* | Specifies a list of one or more fields, separated by commas, that you want to retrieve from the specified *object*.<br>You must specify valid field names and must have read-level permissions to each specified field. The *fieldList* defines the ordering of fields in the query results. |
| *objectType* | Specifies the type of sforce object that you want to query.<br>You must specify a valid sforce object and must have read-level permissions to that object. For a list of valid objects, see List of Supported sforce Object Types on page 61. |
| *conditionExpression* | Determines which rows in the specified *object* to retrieve. If unspecified, the query retrieves all rows in the *object*.<br>See conditionExpression Syntax on page 17 for the appropriate syntax. |

### *conditionExpression* Syntax

The *conditionExpression* uses the following syntax:

```
fieldExpression [logicalOperator fieldExpression2][logicalOperator
fieldExpression3]...
```

- You can use parentheses to define the order in which `fieldExpression`s are evaluated. For example, the following expression is True if *fieldExpression1* is True and *either* *fieldExpression2* or *fieldExpression3* are True.

  `fieldExpression1 AND (fieldExpression2 OR fieldExpression3)`

- However, the following expression is True if either *fieldExpression3* is True or both *fieldExpression1* and *fieldExpression2* are True.

  `(fieldExpression1 AND fieldExpression2) OR fieldExpression3`

See fieldExpression Syntax on page 18 for the syntax of *fieldExpression*s. See Logical Operators on page 19 for the valid logical operators.

## *fieldExpression* Syntax

A *`fieldExpression`* uses the following syntax:

*`fieldName comparisonOperator`* *value*

where:

| Syntax | Description |
|---|---|
| *`fieldName`* | The name of a field in the specified *`object`*. Use of single or double quotes around the name will result in an error. You must have at least read-level permissions to the field. It can be any field—it does not need to be a field in the *`fieldList`*. |
| *`comparisonOperator`* | One of the comparison operators listed in Comparison Operators on page 18. |
| *`value`* | A value, enclosed in *single quotes* (double quotes result in an error), used to compare with the value in *`fieldName`*. You must supply a value whose data type matches the field type of the specified field. You must supply a native value—other field names or calculations are not permitted.<br>For date values, use the formatting listed in Date Formats on page 19. |

### Comparison Operators

A *`fieldExpression`* uses the following *`comparisonOperator`*s:

| Operator | Name | Description |
|---|---|---|
| = | Equals | Expression is True if the value in the specified *`fieldName`* equals the specified *`value`* in the expression. |
| != | Not equals | Expression is True if the value in the specified *`fieldName`* does *not* equal the specified *`value`*. |
| < | Less than | Expression is True if the value in the specified *`fieldName`* is less than the specified *`value`*. |
| <= | Less or equal | Expression is True if the value in the specified *`fieldName`* is less than, or equals, the specified *`value`*. |
| > | Greater than | Expression is True if the value in the specified *`fieldName`* is greater than the specified *`value`*. |

| Operator | Name | Description |
|---|---|---|
| **>=** | Greater or equal | Expression is True if the value in the specified *`fieldName`* is greater than, or equal to, the specified *`value`*. |
| **like** | Like | Expression is True if the value in the specified *`fieldName`* matches the characters of the text string in the specified *`value`*.<br>The *`like`* operator in SOQL is similar to the same operator in SQL; it provides a mechanism for matching partial text strings and includes support for wildcards.<br>• The % and _ wildcards are supported for the *`like`* operator.<br>  • The % wildcard matches zero or more characters.<br>  • The _ wildcard matches exactly one character.<br>• The text string in the specified *`value`* must be enclosed in single quotes.<br>• The *`like`* operator is supported for string fields only (see String Field Type on page 13).<br>• The *`like`* operator performs a case-insensitive match, unlike the case-sensitive matching in SQL.<br>• The *`like`* operator in SOQL does not currently support escaping of special characters such as % or _. The \ (backslash) character should not be used.<br>`select AccountId, FirstName, lastname from Contact where lastname like '%appl_%'`<br>matches *Appleton*, *Apple*, *Bapple*, but not *Appl*. |

## Date Formats

A *`fieldExpression`* uses the following date formats (milliseconds and time zone are optional):

| Use | Format Syntax | Example |
|---|---|---|
| Date only | `YYYY-MM-DD` | 1999-01-01 |
| Date and time | `YYYY-MM-DDThh:mm:ss` | 1999-01-01T24:01:01 |
| Date, time, and milliseconds | `YYYY-MM-DDThh:mm:ss.MILLIS` | 1999-01-01T24:01:01.001 |
| Date, time, milliseconds, and time zone offset | • `YYYY-MM-DDThh:mm:ss.MILLISZ+hh:mm`<br>• `YYYY-MM-DDThh:mm:ss.MILLISZ-hh:mm` | • 1999-01-01T24:01:01.001Z+01:00<br>• 1999-01-01T24:01:01.001Z-01:00 |

*NOTE*

For *`fieldExpression`*s that use date formats, the date is not enclosed in single quotes. No quotes should be used around the date. For example:

```
select Id from Account where CreatedDate > 2003-10-29T11:30:00Z
```

## Logical Operators

A *`logicalOperator`* is used to join two or more *`fieldExpression`*s. A *`logicalOperator`* is one of the following values:

| Operator | Syntax | Description |
|---|---|---|
| `and` | *`fieldExpressionX`* `and` *`fieldExpressionY`* | True only if both *`fieldExpressionX`* and *`fieldExpressionY`* are True. |
| `or` | *`fieldExpressionX`* `or` *`fieldExpressionY`* | True if either *`fieldExpressionX`* and *`fieldExpressionY`* is True. |
| `not` | *`fieldExpressionX`* `or` *`fieldExpressionY`* | True if *`fieldExpressionX`* is True *`fieldExpressionY`* is False. |

## ADVANCED TOPICS

This topic describes advanced but optional topics that might be of interest to some readers. It contains the following sections:

- Custom Objects Types and Custom Fields
- Changing the Batch Size in Queries
- Unicode and Character Encoding Support
- XML Compliance
- Compression
- Multiple Instance Support
- HTTP Persistent Connections
- HTTP Chunking

### Custom Objects Types and Custom Fields

In the salesforce.com user interface, organizations can define custom object types. For custom object types, the **`custom`** flag—a boolean field in the DescribeSObjectResult—is True. Custom object types are defined in the salesforce.com user interface only. Client applications cannot create custom object types via the sforce API. However, client applications with sufficient permissions can invoke other API calls on custom objects.

Organizations can also define custom fields for standard or custom object types. For custom fields, the **`custom`** flag—a boolean field in the Field object—is True. Custom fields are defined in the salesforce.com user interface only. Client applications cannot define custom fields via the sforce API. For the most part, client applications do not need to know whether a field is a standard field or a custom field.

For custom entities and fields, your organization's salesforce.com administrator

Custom objects and fields have an associated name field that is defined by your salesforce.com administrator. Custom objects must have unique names within your organization, and custom fields must have unique names within the same object. In your WSDL file, custom object and field names have a **`__c`** suffix, such as **`myCustomObject__c`** and **`myCustomField__c`**.

Note that all numeric custom fields are handled as type double. For more information, see Double Field Type on page 14.

### Changing the Batch Size in Queries

By default, the batch size for the number of records returned in a query or queryMore call is set to 2000. Client applications can change this setting by specifying the batch size in the QueryOptions portion of the SOAP header before invoking the query call.

The following sample Java (Axis) code demonstrates setting the batch size to three (3) records.

```
_QueryOptions qo = new _QueryOptions();
qo.setBatchSize(new Integer(3));
```

```
binding.setHeader("SoapService", "QueryOptions", qo);
```

The following sample C# (.NET) code demonstrates setting the batch size to three (3) records.

```
binding.QueryOptionsValue = new QueryOptions();
binding.QueryOptionsValue.batchSize = 10;
binding.QueryOptionsValue.batchSizeSpecified = true;
```

## Unicode and Character Encoding Support

The salesforce.com server supports either full Unicode characters or ISO-8859-1 characters, depending on the instance. You can determine the encoding ahead of time using the describeGlobal call. The encoding specified by the describeGlobal call is the character set that is supported by that sforce instance.

The response from the server will be in UTF-8 or ISO-8859-1 encoding, depending on the character set supported by the instance. This is usually handled for you by the SOAP client. All servers accept either encoding, but the ISO-8859-1 server cannot support characters outside of the ISO-8859-1 range. Data sent to that server outside of the valid ISO-8859-1 range may either be truncated or cause an error.

The sforce API follows the conventions of XML character encoding, which means that any character encoding specified in the HTTP header is ignored. Usually this is handled for you by the SOAP client. When writing directly to the XML/HTTP layer, you must specify an encoding in the XML header line. The sforce server specifies the encoding of the response in both the XML header and the HTTP header, in case any clients only support the HTTP encoding form.

*NOTE* Refer to the XML specification for information on specifying and interpreting the encoding specifier in an XML document. In particular, see www.w3.org/TR/REC-xml#charencoding.

## XML Compliance

The sforce API is based on XML, which requires all documents to be well formed. Part of that requirement is that certain Unicode characters are not allowed in an XML document, even in an escaped form, and that others must be encoded according to their location. Normally this is handled for you by any standard SOAP or XML client. Clients must be able to parse any normal XML escape sequence, and must not pass up invalid XML characters.

Some characters, as mentioned, are illegal even if they are escaped. The illegal characters include the Unicode surrogate blocks and a few other Unicode characters. All are seldom-used control characters that are usually not important in any data, and tend to cause problems with many programs. Although they are not allowed in XML documents, they are allowed in HTML documents and may be present in sforce data. The illegal characters will be stripped from any API response.

The following characters are illegal:

### Table 5: Illegal XML Characters

| |
|---|
| **0xFFFE** |
| **0xFFFF** |
| **Control characters 0x0 - 0x19**<br>(Not including 0x9, 0xA, 0xD, tab, newline, and carriage return) |
| **0xD800 - 0xDFFF** |

For UTF-8 encoding, force supports only the basic UCS-2 plane and does not support any of the extended UCS-4 characters. UCS-4 support is extremely rare in any system. UCS-2 is the set that Java and Windows NT support. For more information about XML characters and character sets, see *www.w3.org/TR/REC-xml#charsets*.

## Compression

The sforce API allows the use of compression on the request and the response, using the standards defined by the HTTP 1.1 specification. This is automatically supported by some SOAP/WSDL clients, and can be manually added to others. Check the sforce.com site for more information on particular clients.

Compression is not used unless the client specifically indicates that it supports compression. For better performance, we suggest that clients accept and support compression as defined by the HTTP 1.1 specification.

To indicate that the client supports compression, you should include the HTTP header "Accept-Encoding: gzip, deflate" or a similar heading. The server compresses the response if the client properly specifies this header. The response includes the header "Content-Encoding: deflate" or "Content-Encoding: gzip," as appropriate. You can also compress any request by including a "Content-Encoding: deflate" or "gzip" header.

Most clients are partially constrained by their network connection, even on a corporate LAN. The sforce API allows the use of compression to improve performance. Almost all clients can benefit from response compression, and many clients may benefit from compression of requests as well. The sforce server supports deflate and gzip compression according the HTTP 1.1 specification.

### Response Compression

The sforce server can optionally compress responses. Responses are compressed only if the client sends an Accept-Encoding header with either gzip or deflate compression specified. The server is not required to compress the response even if you have specified Accept-Encoding, but it normally does. If the server compresses the response, it also specifies a Content-Encoding header with the name of the compression algorithm used, either gzip or deflate.

### Request Compression

Clients can also compress requests. The sforce server decompresses any requests before processing. The client must send up a Content-Encoding HTTP header with the name of the appropriate compression algorithm. For more information, see:

- Content-Encoding at www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.11
- Accept-Encoding at www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.3
- Content Codings at www.w3.org/Protocols/rfc2616/rfc2616-sec3.html#sec3.5

## Multiple Instance Support

The sforce API provides access to all worldwide sforce servers, including:

- US/North America 0
- US/North America 1
- EMEA/Europe
- AP/Japan

The different sets of servers are referred to as *instances*.

*NOTE* An organization is not guaranteed to be on a particular instance, just because that organization is located in a particular region, although that is generally true. Also, additional instances may be added in the future.

The SOAP implementation of the sforce API also provides a single login server. You can log in to any organization via a single entry point, without having to hard-code the instance for your organization. To access an organization via the sforce API, you must first authenticate the session by sending a login request to the login server at the following URL:

```
https://www.salesforce.com/servlet/servlet.SoapApi
```

The insecure version of the URL is also supported:

**`http://www.salesforce.com/servlet/servlet.SoapApi`**

All subsequent calls to the server during the session should be made to the URL returned in the login response.

## HTTP Persistent Connections

Most clients achieve better performance if they use HTTP 1.1 persistent connection to reuse the socket connection for multiple requests. Persistent connections are normally handled by your SOAP/WSDL client automatically. For more details, see the HTTP 1.1 specification at:

www.w3.org/Protocols/rfc2616/rfc2616-sec8.html#sec8.1

## HTTP Chunking

Clients that use HTTP 1.1 may receive chunked responses. Chunking is normally handled by your SOAP/WSDL client automatically.

# CHAPTER 3: sforce API Calls

This topic describes the sforce Web services API calls that your client applications can invoke to retrieve and change your organization’s information. It contains the following sections:

- About sforce API Calls
- List of Supported sforce API Calls

The rest of this topic describes most of the sforce API calls in detail. Some of the sforce API calls are described in sforce Utility API Calls on page 53.

## ABOUT SFORCE API CALLS

The sforce API calls represent specific operations that your client applications can invoke at run-time to perform certain tasks. For example, you can query your organization’s data; add, update, and delete information; obtain metadata about your organization’s data; and run utilities to perform administration tasks.

### Characteristics of sforce API Calls

All sforce API calls are:

- **Service requests and responses**—Your client application prepares and submits a service request to the sforce API server, the sforce API server processes the request and returns a response, and the client application handles the response as appropriate.
- **Synchronous**—Once the API call is invoked, your client application waits until it receives a response from the service. Asynchronous calls are not supported.
- **Committed automatically**—Every operation that writes to a salesforce.com table is committed automatically. This is analogous to the AUTOCOMMMIT setting in SQL. For create, update, and delete calls that attempt to write to multiple rows in a table, the write operation for *each* row is treated as a *separate transaction*. For example, if a client application attempts to create two new accounts, they’re created using mutually exclusive insert operations that succeed or fail individually, not as a group.

### Factors that Affect Data Access

When using the sforce API, the following factors affect access to your organization’s data:

- Whether a given object exists for your organization.
- Whether the object exists in the generated WSDL. The output of the WSDL depends on the access rights of the user who generated the WSDL. You can access an object only if it is defined in the generated WSDL that you are using. For more information, see Generating the WSDL File for Your Organization on page 3.
- Your client application logs in as a user to the sforce API server. The permissions profile associated with that logged in user determine the level of access to specific objects and fields in your organization’s information. You can access objects and fields only if the security settings in the user’s personal profile permit such access. For more information, see Security in the sforce API on page 11.

### Typical API Call Sequence

For each sforce API call, your client application typically:

- Prepares the request by defining request parameters, if applicable

- Invokes the call, which passes the request with its parameters to the sforce API server for processing
- Receives the response (synchronously) from the sforce API server
- Handles the response, either by processing the returned data (for a successful invocation) or by handling the error (for a failed invocation).

## Start By Logging In to the sforce Single Sign-on Server

Before invoking any other sforce API calls, a client application must first invoke the login call to establish a session with the sforce single logon server, set the returned server URL as the target server for subsequent API requests, and set the returned session ID in the SOAP header to provide server authorization for subsequent API requests. For more information, see login on page 41 and Sample Code Walkthrough on page 4.

## Core Data Objects

Many calls in the sforce API use the following data objects:

- ID
- sObject
- Error
- ExceptionCode
- ApiFault

### ID

Almost all objects in the sforce API have an associated ID, which is a string (18 alphanumeric characters in length) that uniquely identifies an individual object. An ID is analogous to a primary or foreign key field in a database table. When you create a new object, the sforce API server generates an `ID` value for the object, ensuring that it is properly formatted and unique within your organization's data. Thereafter, you can refer to the object by its unique ID in subsequent sforce API calls.

### sObject

An `sObject` represents an sforce object, such as an individual Account or Campaign. For a complete list of sforce objects, see Chapter 5: sforce Objects on page 59.

An `sObject` has the following properties:

| Name | Type | Description |
|---|---|---|
| **fieldsToNull** | string[] | Array of one or more field names whose value you want to explicitly set to `null`. Used only with the update call. Ensures that, for this `sObject`, the value in the specified field will be reset to `null`.<br><br>You can specify only those fields that you can update. For example, specifying an ID field or required field results in a run-time error. |
| **ID** | ID | Unique ID for this individual object. For the create call, this value is null. For all other sforce API calls, this value must be specified. |

### Error

An `Error` contains information about an error that occurred during an sforce API call (create, update, or delete only).

An `Error` has the following properties:

| Name | Type | Description |
|---|---|---|
| **StatusCode** | Exception Code | Status code that characterizes the error. |
| **message** | string | Error message text. |
| **fields** | string[] | Reserved for future use. |

### ExceptionCode

An `ExceptionCode` contains information about an ApiFault that occurred during an sforce API call.

The following list of `ExceptionCode` values is defined in your WSDL file:

| ExceptionCode | Description |
|---|---|
| **UNKNOWN_EXCEPTION** | Server encountered an internal error. You should report this problem to salesforce.com. |
| **API_CURRENTLY_DISABLED** | API functionality is temporarily down due to a server problem. |
| **API_DISABLED_FOR_ORG** | Organization is not enabled for use of the API. A representative from the organization must contact salesforce.com to enable API access. |
| **EXCEEDED_QUOTA** | Organization storage limits have been exceeded during a create call. |
| **EXCEEDED_RATE_LIMIT** | Client sent concurrent API requests and the original request has been terminated. |
| **SERVER_UNAVAILABLE** | A server that is necessary for this call is currently down. Other types of requests might still work. |
| **INVALID_TYPE** | Specified sObject type is invalid. |
| **UNSUPPORTED_API_VERSION** | WSDL file was generated from an older version that is no longer supported. |
| **INVALID_CLIENT** | Client is invalid. |
| **UNSUPPORTED_CLIENT** | This version of the client is no longer supported. |
| **FUNCTIONALITY_NOT_ENABLED** | Functionality has been temporarily disabled. Other calls may continue to work. |
| **INVALID_SESSION_ID** | Specified `sessionID` is invalid or has expired. You should log in again to generate a new session. |
| **TRIAL_EXPIRED** | Organization is a trial organization that has reached its expiration date. A representative from the organization must contact salesforce.com to re-enable the organization. |

| ExceptionCode | Description |
|---|---|
| **INVALID_LOGIN** | Invalid login credentials. |
| **LOGIN_DURING_RESTRICTED_TIME** | User is restricted from logging in during this time period. |
| **LOGIN_DURING_RESTRICTED_DOMAIN** | User is restricted from logging in from this IP address. |
| **PASSWORD_LOCKOUT** | User has attempted multiple invalid logins and has been locked out. The user must contact the organization administrator to re-enable the account. |
| **ORG_LOCKED** | Organization has been locked. You must contact salesforce.com to re-enable the organization. |
| **INVALID_FIELD** | Specified field name is invalid. |
| **INVALID_QUERY_FILTER_OPERATOR** | An operator used in the query filter clause is invalid, at least for that field. |
| **EXCEEDED_ID_LIMIT_ON_RETRIEVE** | Too many IDs were requested in a retrieve call. |
| **INVALID_QUERY_LOCATOR** | Specified `queryLocator` parameter in a queryMore call is invalid. |
| **MALFORMED_QUERY** | Specified query string is not valid. |
| **INVALID_BATCH_SIZE** | Batch size specified in the query options is out of the supported range. |
| **MALFORMED_SEARCH** | Reserved for future use. |
| **INVALID_SEARCH_SCOPE** | Reserved for future use. |

### ApiFault

If an error occurs during the invocation of an API call, the sforce API server throws an exception and returns an `ApiFault` with an associated ExceptionCode and error message text that provide additional information about the error. For more information about `ApiFault`, see List of APIFault Codes on page 27. Fault codes are of type `ApiFault`, which has the following properties.

| Name | Type | Description |
|---|---|---|
| **exceptionCode** | ExceptionCode | Exception code. |
| **exceptionMessage** | string | Error message text. |

## List of APIFault Codes

The following table lists the ApiFault codes that the sforce API Server returns if an error occurs when processing a service request.

**Table 6: Fault Codes for sforce API Calls**

| Fault | Description |
|---|---|
| **LoginFault** | Error occurred during the login call. |

Table 6: Fault Codes for sforce API Calls (Continued)

| Fault | Description |
|---|---|
| `InvalidSObjectFault` | Invalid sObject in a describeSObject, create, update, retrieve, or query call. |
| `InvalidFieldFault` | Invalid field in a retrieve or query call. |
| `MalformedQueryFault` | Problem in the `queryString` passed in a query call. |
| `InvalidQueryLocatorFault` | Problem in the `queryLocator` passed in a queryMore call. |
| `InvalidIdFault` | Specified ID was invalid in a setPassword or resetPassword call. |
| `UnknownErrorFault` | Unknown error. The error is not associated with any other ApiFault. |

## LIST OF SUPPORTED SFORCE API CALLS

Table 7: Supported Calls in the sforce API

| Task / Call | Description |
|---|---|
| create | Adds one or more new individual objects to your organization's data. |
| delete | Deletes one or more individual objects from your organization's data. |
| describeGlobal | Retrieves a list of available object types for your organization's data. |
| describeSObject | Retrieves the next batch of objects from a query. |
| getServerTimestamp | Retrieves the current system timestamp from the sforce API server. |
| getUserInfo | Retrieves personal information for the user associated with the current session. |
| login | Logs in to the sforce single sign-on server and starts a client session. |
| query | Executes a query against the specified object and returns data that matches the specified criteria. |
| queryMore | Retrieves the next batch of objects from a query. |
| resetPassword | Changes a user's password to a server-generated value. |
| retrieve | Retrieves one or more objects based on the specified object IDs. |
| setPassword | Sets the specified user's password to the specified value. |
| update | Updates one or more existing objects in your organization's data. |

### create

Adds one or more new individual objects to your organization's data.

## Syntax

```
SaveResult[] = sfdc.create(sObject[] sObjects);
```

## Usage

Use create to add one or more individual objects, such as an Account or Contact, to your organization’s information. The create call is analogous to the INSERT statement in SQL.

When creating objects, consider the following rules and guidelines:

- Your client application must be logged in with sufficient access rights to create individual objects within the specified object type. For more information, see Factors that Affect Data Access on page 24.
- Certain objects—and certain fields within those objects—require special handling or permissions. For example, you might also need permissions to access this object’s parent object. Before you attempt to create a particular object, be sure to read its description in Chapter 5: sforce Objects on page 59.
- The sforce API server generates unique values for ID fields automatically. For create, you cannot explicitly specify an ID value in the sObject. The SaveResult contains the ID of each object that was successfully created.
- The sforce API server populates certain fields automatically, such as **CreatedDate**, **CreatedById**, **LastModifiedDate**, **LastModifiedById**, and **SystemModstamp**. You cannot explicitly specify these values.
- For some objects, certain fields have a default value, such as **OwnerID**. If you do not specify a value for such fields, the sforce API server populates these fields with the default value. For example, if you do not override the **OwnerID**, then the sforce API server populates this field with the user ID associated with the user under which your client application is logged in.
- For required fields that do not have a preconfigured default value, you must supply a value.
- For all other fields in the object, if you do not explicitly specify a value, then its value is `null`.
- Your client application must conform to the rules of referential integrity. For example, if you are creating an object that is the child of a parent object, you must supply the foreign key information that links the child to the parent. For example, when creating a CaseComment, you must supply the valid caseID for the parent Case, and that parent Case must exist in the database.
- You must supply values that are valid for the field’s data type, such as integers (not alphabetic characters) for integer fields. In your client application, follow the data formatting rules specified for your programming language and development tool (your development tool will handle the appropriate mapping of data types in SOAP messages).

Creating objects involves the following basic steps:

1. Instantiate one or more individual objects within the object type. For each object, you populate its fields with the data that you want to add.
2. Construct an `sObject[]` array and populate that array with the objects that you want to create. All objects *must* be of the same object type.
3. Call create, passing in the `sObject[]` array.
4. Process the results in the `SaveResult[]` object to verify whether the objects have been successfully created.

## Sample Code—Java

```
public void createAccountSample() {

   // Create two account objects
   Account account1 = new Account();
   Account account2 = new Account();
   // Set some fields on the account1 object
   account1.setAccountNumber("002DF99ELK9");
   account1.setBillingCity("Wichita");
   account1.setBillingCountry("US");
   account1.setBillingState("KA");
   account1.setBillingStreet("4322 Haystack Boulevard");
   account1.setBillingPostalCode("87901");

   // Set some fields on the account2 object
   account2.setName("Golden Straw");
   account2.setAccountNumber("003DF99ELK9");
   account2.setBillingCity("Oaklanc");
   account2.setBillingCountry("US");
   account2.setBillingState("CA");
   account2.setBillingStreet("666 Raiders Boulevard");
   account2.setBillingPostalCode("97502");

   // Create an array of SObjects to hold the accounts
   SObject[] sObjects = new SObject[2];

   // Add the accounts to the SObject array
   sObjects[0] = account1;
   sObjects[1] = account2;

   // Invoke the create call
   SaveResult[] saveResults = binding.create(sObjects);

   // Handle the results
   for (int i=0;i<saveResults.length;i++) {
      // Determine whether create succeeded or had errors
      if (saveResults[i].isSuccess()) {
         // No errors, so we will retrieve the id created for this index
         System.out.println(saveResults[i].getId().getValue());
      }
      else {
         // Handle the errors
         ...
      }
   }
}
```

## Sample Code—C#

```
private void createAccount()
{
   // Create an account object to send to the service
   Account account = new Account();

   // Set several properties
   account.Name = "Koka Kola";
   account.Website = "www.kokakola.com";

   // Add the account to an array of SObjects
   sObject[] records = new sObject[] {account};

   // Invoke the create call, passing in the account properties
   // and saving the results in a SaveResult object
   SaveResult[] saveResults = binding.create(records);
```

```
            // Access the new ID
            String newID = saveResults[0].id;
        }
```

### Arguments

| Name | Type | Description |
|---|---|---|
| **`sObjects`** | sObject[] | Array of one or more objects to create. The sforce API server creates these objects in array index order. |

### Response

SaveResult[]

### Fault

`InvalidSObjectFault`

`UnknownErrorFault`

### See Also

*Sample SOAP Messages—create*
*About sforce API Calls on page 24*

### SaveResult

The create call returns an array of `SaveResult` objects. Each element in the `SaveResult` array corresponds to the sObject[] array of passed as the `sObjects` parameter in the create call. For example, the object returned in the first index in the `SaveResult` array matches the object specified in the first index of the sObject[] array. A `SaveResult` object has the following properties:

| Name | Type | Description |
|---|---|---|
| **`id`** | ID | ID of the sObject that you attempted to create. If this field contains a value, then the object was created successfully. If this field is empty, then the object was not created and the sforce API server returned error information instead. |
| **`success`** | boolean | Indicates whether the create call succeeded (True) or not (False) for this object. |
| **`errors`** | Error[] | If an error occurred during the create call, an array of one or more Error objects providing the error code and description. |

## delete

Deletes one or more individual objects from your organization's data.

## Syntax

```
DeleteResult[] = sfdc.delete(ID[] ids);
```

## Usage

Use delete to delete one or more existing objects, such as individual accounts or contacts, in your organization's data. The delete call is analogous to the DELETE statement in SQL.

When deleting objects, consider the following rules and guidelines:

- Your client application must be logged in with sufficient access rights to delete individual objects within the specified object type. For more information, see Factors that Affect Data Access on page 24.
- In addition, you might also need permissions to access this object's parent object. For special access requirements, see the object's description in Chapter 5: sforce Objects on page 59.
- To ensure referential integrity, the delete call supports cascading deletions. If you delete a parent object, you delete its children automatically, as long as each child object can be deleted. For example, if you delete a Case, the sforce API automatically deletes any CaseComment, CaseHistory, and CaseSolution objects associated with that case. However, if a CaseComment is not deletable or is currently being used, then the delete call on the parent Case will fail.

Deleting objects involves the following basic steps:

1. Determine the ID of each object that you want to delete. For example, you might call query to retrieve a set of records that you want to delete based on specific criteria.
2. Construct an `ID[]` array and populate it with the IDs of each object that you want to delete. You can specify the IDs of different object types. For example, you could specify the ID for an individual Account and an individual Contact in the same array.
3. Call delete, passing in the `ID[]` array.
4. Process the results in the `DeleteResult[]` object to verify whether the objects have been successfully deleted.

## Sample Code—Java

```
public void deleteSample() {

   // Create an array of IDs to hold the IDs of the records to delete
   ID[] ids = new ID[2];

   // Add the IDs to the ID array
   ids[0].setValue("001x00000000JerAAE");
   ids[1].setValue("001x00000000JesAAE");

   // Invoke the delete call
   DeleteResult[] deleteResults = binding.delete(tasks);
   // Process the results
   for (int i=0;i<deleteResults.length;i++) {
      DeleteResult deleteResult = deleteResults[i];
      // Determine whether delete succeeded or had errors
      if (deleteResult.isSuccess()) {
         // Get the id of the deleted record
         deleteResult.getId();
         }
      else {
         // Handle the errors
         Error[] errors = deleteResult.getErrors();
      }
   }
}
```

## Sample Code—C#

```
private void deleteAccount()
{
   // Delete call takes an string array of Ids as parameter
   String[] IDs = new Sring[] {""};

   // Invoke the delete call, saving the result in a DeleteResult object
   DeleteResult[] deleteResults = binding.delete(IDs);

   // Determine whether the delete call succeeded or failed
   if (deleteResults[0].success)
   {
      // Delete operation succeeded
      System.Diagnostics.Trace.WriteLine("Deleted: " + deleteResults[0].id);
   }
   else
   {
      // Delete operation failed
      System.Diagnostics.Trace.WriteLine("Couldn't delete because: " +
deleteResults[0].errors[0].message);
   }
}
```

## Arguments

| Name | Type | Description |
|---|---|---|
| **ids** | ID[] | Array of one or more IDs associated with the objects to delete. The sforce API server deletes these objects in array index order. |

## Response

DeleteResult[]

## Fault

InvalidSObjectFault

UnknownErrorFault

## See Also

*Sample SOAP Messages—delete*
*About sforce API Calls on page 24*

## DeleteResult

The delete call returns an array of `DeleteResult` objects. Each element in the `DeleteResult` array corresponds to the ID[] array of passed as the `ids` parameter in the delete call. For example, the object returned in the first index in the `SaveResult` array matches the object specified in the first index of the ID[] array.

A `DeleteResult` object has the following properties:

| Name | Type | Description |
|---|---|---|
| **`id`** | ID | ID of an sObject that you attempted to delete. |
| **`success`** | boolean | Indicates whether the delete call succeeded (True) or not (False) for this object. |
| **`errors`** | Error[] | If an error occurred during the delete call, an array of one or more Error objects providing the error information. |

## describeGlobal

Retrieves a list of available object types for your organization's data.

### Syntax

```
DescribeGlobalResult = sfdc.describeGlobal(null);
```

### Usage

Use describeGlobal to obtain the list of available object types for your organization. You can then iterate through this list and use describeSObject to obtain metadata about individual objects.

Your client application must be logged in with sufficient access rights to retrieve metadata about your organization's data. For more information, see Factors that Affect Data Access on page 24.

### Sample Code—Java

```
public void describeGlobalSample() {

   // Invoke describeGlobal call and save results in DescribeGlobalResult object
   DescribeGlobalResult describeGlobalResult = binding.describeGlobal(null);
   if (! (describeGlobalResult == null)) {
      // Get the array of object names from the result
      String[] types = describeGlobalResult.getTypes();
      if (! (types == null)) {
         for (int i = 0; i < types.length; i++) {
            System.out.println((types[i]));
         }
      }
   }
}
```

### Sample Code—C#

```
private void globalDescribe()
{
   // Invoke describeGlobal call and save results in DescribeGlobalResult object
   DescribeGlobalResult dgr = binding.describeGlobal();

   // Iterate through the results
   for (int i=0;i<dgr.types.Length;i++)
   {
      // The dgr.types[i] object is a string
      System.Diagnostics.Trace.WriteLine(dgr.types[i]);
   }
   binding.describeSObject
}
```

### Arguments

None.

### Response

DescribeGlobalResult

### Fault

`UnknownErrorFault`

### See Also

*describeSObject on page 35*
*Sample SOAP Messages—describeGlobal*
*About sforce API Calls on page 24*
*sforce Partner Web Services API on page 104*

### DescribeGlobalResult

The describeGlobal call returns a `DescribeGlobalResult` object, which has the following properties.

| Name | Type | Description |
|---|---|---|
| **`encoding`** | string | Specifies how an organization's data is encoded, such as `UTF-8` or `ISO8859/1`. |
| **`maxBatchSize`** | int | Maximum number of records allowed in a create, update, or delete call. |
| **`types`** | string[] | List of available object types for your organization. You iterate through this list to retrieve the object type string that you pass to describeSObject. |

## describeSObject

Describes metadata (field list and object properties) for the specified object type.

### Syntax

```
DescribeSObjectResult = sfdc.describeSObject(string sObjectType);
```

### Usage

Use describeSObject to obtain metadata for a given object type. You can first call describeGlobal to retrieve a list of all object types for your organization, then iterate through this list and use describeSObject to obtain metadata about individual objects.

Your client application must be logged in with sufficient access rights to retrieve metadata about your organization's data. For more information, see Factors that Affect Data Access on page 24.

## Sample Code—Java

```
public void describeSample() {

   // Invoke describeSObject and save results in DescribeSObjectResult
   DescribeSObjectResult describeSObjectResult =
binding.describeSObject("account");
   // Determine whether the describeSObject call succeeded
   if (! (describeSObjectResult == null)) {
      // Retrieve fields from the results
      Field[] fields = describeSObjectResult.getFields();
      // Get the name of the object
      String objectName = describeSObjectResult.getName();
      // Get some flags
      boolean isActivateable = describeSObjectResult.isActivateable();
      // Many other values are accessible

      if (! (fields == null)) {
      // Iterate through the fields to get properties for each field
         for (int i = 0; i < fields.length; i++) {
            Field field = fields[i];
            int byteLength = field.getByteLength().intValue();
            int digits = field.getDigits().intValue();
            String label = field.getLabel();
            int length = field.getLength().intValue();
            String name = field.getName();
            PicklistEntry[] picklistValues = field.getPicklistValues();
            int precision = field.getPrecision().intValue();
            String[] referenceTos = field.getReferenceTo();
            int scale = field.getScale().intValue();
            FieldType fieldType = field.getType();
            boolean fieldIsCreateable = field.isCreateable();
            // Determine whether there are picklist values
            if (picklistValues != null) {
               System.out.println("Picklist values = ");
               for (int j = 0; j < picklistValues.length; j++) {
                  if (picklistValues[j].getLabel() != null) {
                     System.out.println("    Item:  " +
picklistValues[j].getLabel());
                  }
               }
            }
            // Determine whether this field refers to another object
            if (referenceTos != null) {
               System.out.println("Field references the following objects:");
               for (int j = 0; j < referenceTos.length; j++) {
                  System.out.println("    " + referenceTos[j]);
               }
            }
         }
      }
   }
}
```

## Sample Code—C#

```
private void sObjectDescribe()
{
   // Invoke describeSObject and save results in DescribeSObjectResult
   DescribeSObjectResult dsr = binding.describeSObject("Account");

   // Get value that indicates whether we can create a record
   bool canCreate = dsr.createable;

   // Get a field and save its name
   String fldName = dsr.fields[0].name;
}
```

## Arguments

| Name | Type | Description |
|---|---|---|
| **sObjectType** | string | Object type. The specified value must be a valid object type for your organization. For a complete list of sforce object types, see List of Supported sforce Object Types. |

## Response

DescribeSObjectResult

## Fault

`InvalidSObjectFault`

`UnknownErrorFault`

## See Also

*describeGlobal on page 34*
*Sample SOAP Messages—describeSObject*
*About sforce API Calls on page 24*
*sforce Partner Web Services API on page 104*

## DescribeSObjectResult

The describeSObject call returns a `DescribeSObjectResult` object, which has the following properties. Note that, while the boolean properties indicate whether certain API calls can be used for an object type, other factors (such as security settings in the user's personal profile) also affect whether such operations can be performed on the object type.

| Name | Type | Description |
|---|---|---|
| **name** | string | Name of the object type. This is the same string that was passed in as the `sObjectType` parameter. |
| **custom** | boolean | Indicates whether the object is a custom object (True) or not (False). |

| Name | Type | Description |
|---|---|---|
| **queryable** | boolean | Indicates whether the object can be queried via the query call (True) or not (False). |
| **createable** | boolean | Indicates whether the object can be created via the create call (True) or not (False). |
| **updateable** | boolean | Indicates whether the object can be updated via the update call (True) or not (False). |
| **deletable** | boolean | Indicates whether the object can be deleted via the delete call (True) or not (False). |
| **undeletable** | boolean | Reserved for future use. |
| **activateable** | boolean | Reserved for future use. |
| **retrieveable** | boolean | Indicates whether the object can be retrieved via the retrieve call (True) or not (False). |
| **searchable** | boolean | Reserved for future use. |
| **replicateable** | boolean | Reserved for future use. |
| **fields** | Field[] | Array of fields associated with the object. The mechanism for retrieving information from this list varies among development tools. |

## Field

In the DescribeSObjectResult, the `fields` property contains an array of fields of type `Field`. Each field represents a field in an sforce API object. The array contains only the fields that the user can view, as defined by the user's field-level security settings.

| Name | Type | Description |
|---|---|---|
| **type** | FieldType | See FieldType for a list of allowable values. |
| **name** | string | Field name used in sforce API calls, such as create, delete, and query. |
| **label** | string | Text label that is displayed next to the field in the salesforce.com user interface. This label can be localized. |
| **soapType** | SOAPType | See SOAPType for a list of allowable values. |
| **custom** | boolean | Indicates whether the field is a custom field (True) or not (False). |
| **nillable** | boolean | Indicates whether the field is nillable (True) or not (False). A nillable field can have empty content. A non-nillable field must have a value in order for the object to be created or saved. |
| **length** | int | For string fields, the maximum size of the field in Unicode characters (not bytes). |
| **byteLength** | int | For variable-length fields (including binary fields), the maximum size of the field, in bytes. |

| Name | Type | Description |
|---|---|---|
| `restrictedPicklist` | boolean | Indicates whether the field is a restricted pick list (True) or not (False). |
| `picklistValues` | PickListEntry[] | Provides the list of valid values for the picklist. Specified only if restrictedPicklist is True. |
| `referenceTo` | string[] | For fields that refer to other objects, this array indicates the object types of the referenced objects. |
| `precision` | int | For fields of type double. Maximum number of digits that can be stored, including all numbers to the left and to the right of the decimal point (but excluding the decimal point character). |
| `scale` | int | For fields of type double. Number of digits to the right of the decimal point. The sforce API server silently truncates any extra digits to the right of the decimal point, but it returns a fault response if the number has too many digits to the left of the decimal point. |
| `digits` | int | For fields of type integer. Maximum number of digits. The sforce API server returns an error if an integer value exceeds the number of digits. |
| `selectable` | boolean | Indicates whether the field is selectable (True) or not (False). If True, then this field can be specified in the list of fields of a query string in a query call. |
| `filterable` | boolean | Indicates whether the field is filterable (True) or not (False). If True, then this field can be specified in the WHERE clause of a query string in a query call. |
| `createable` | boolean | Indicates whether the field can be created (True) or not (False). If True, then this field value can be set in a create call. |
| `updateable` | boolean | Indicates whether the field is updateable (True) or not (False). If True, then this field value can be set in a update call. |

## FieldType

In the Field object associated with the DescribeSObjectResult, the `type` field can contain one of the following strings. For more information about field types, see Field Types in the sforce API on page 12.

| Field Type | What the Field Contains |
|---|---|
| `string` | String values. |
| `boolean` | Boolean (True / False) values. |
| `i4` | Integer (int) values. |
| `double` | Double values. |

| Field Type | What the Field Contains |
|---|---|
| **`date`** | Date values. |
| **`datetime`** | Date and time values. |
| **`base64`** | Base64-encoded arbitrary binary data (of type base64Binary). Used for Attachment, Document, and Scontrol objects. |
| **`id`** | Primary key field for the object. |
| **`reference`** | Cross-references to a different sforce object. Analogous to a foreign key field in SQL. |
| **`currency`** | Currency values. |
| **`textarea`** | String that is displayed as a multi-line text field. |
| **`percent`** | Percentage values. |
| **`phone`** | Phone numbers. Values can include alphabetic characters. Client applications are responsible for phone number formatting. |
| **`url`** | URL values. Client applications should commonly display these as hyperlinks. |
| **`email`** | Email addresses. |
| **`combobox`** | Comboboxes, which provide a set of enumerated values and allow the user to specify a value not in the list. |
| **`picklist`** | Picklists, which provide a set of enumerated values. |

## SOAPType

In the Field property associated with the DescribeSObjectResult, the `SOAPType` field can contain any one of the following string values. All of the values preceded by **`xsd:`** are XML schema primitive data types. For more information about the XML schema primitive data types, see the World Wide Web Consortium's publication *XML Schema Part 2: Datatypes* at the following URL: http://www.w3.org/TR/xmlschema-2/.

| Value | Description |
|---|---|
| **`tns:ID`** | Unique ID associated with an sObject. |
| **`xsd:base64Binary`** | Base 64-encoded binary data. |
| **`xsd:boolean`** | Boolean (True / False) values. |
| **`xsd:dateTime`** | Date/time values. |
| **`xsd:double`** | Double values. |
| **`xsd:int`** | Integer values. |
| **`xsd:string`** | Character strings. |

## PickListEntry

In the Field object associated with the DescribeSObjectResult, the `picklistValues` field contains an array of `PickListEntry` properties. Each `PickListEntry` can contain any one of the following string values. For more information, see Picklist Field Type on page 16.

| Name | Type | Description |
|---|---|---|
| **label** | string | Display name of this item in the picklist. |
| **value** | string | Value of this item in the picklist. |
| **defaultValue** | boolean | Indicates whether this item is the default item (True) in the picklist or not (False). Only one item in a picklist is designated as the default. |
| **active** | boolean | Indicates whether this item must be displayed (True) or not (False) in the drop-down list for the picklist field in the user interface. |

## login

Logs in to the sforce single sign-on server and starts a client session.

### Syntax

```
LoginResult = sfdc.login(string username, string password);
```

### Usage

Use the login call to log in to the sforce single sign-on server and start a client session. A client application *must* log in and obtain a session ID and server URL before making any other sforce API calls.

When a client application invokes the login call, it passes in a user name and password. Upon invocation, the sforce API server authenticates the login and returns the session ID for the session, the user ID associated with the logged in user name, and an URL that points to the sforce API server to use in all subsequent sforce API calls.

After logging in, a client application needs to:

- set the session ID in the SOAP header so that the sforce API server can validate subsequent requests for this session
- specify the server URL as the target server for subsequent service requests

Development tools differ in the way you specify session headers and server URLs. For more information, see the documentation for your particular development tool.

Client applications do not need to explicitly log out to end the session. Sessions expire automatically after a period of inactivity, which can be configured (in the Setup section in the salesforce.com user interface) for your organization to be 30, 60, or 120 minute intervals. If you have a client application that periodically polls the sforce API server at an interval longer than the configured time-out, then that application should log in each time to obtain a new session.

### Sample Code—Java

The following sample Java code shows logging in to the sforce single sign-on server, getting the login result, setting the target server URL to the returned URL, and setting the returned session ID into the session header for Axis.

```
private void login() {

   // Create binding object for sforce
   SoapBindingStub sfdc = (SoapBindingStub) new SforceServiceLocator().getSoap();

   // login
   LoginResult loginResult = sfdc.login("userName", "password");

   // Reset the SOAP endpoint to the returned server URL
   sfdc = (SoapBindingStub) new
   SforceServiceLocator().getSoap(new java.net. URL(loginResult.getServerUrl()));

   // Create a new session header object
   // add the session ID returned from the login
   _SessionHeader sh = new _SessionHeader();
   sh.setSessionId(loginResult.getSessionId());

   // Set the session header for subsequent call authentication
   sfdc.setHeader("SoapService", "SessionHeader", sh);
}
```

## Sample Code—C#

```
private void login()
{
   // Create service object for sforce
   SforceService sfdc = new SforceService();

   // Invoke the login call and save results in LoginResult
   LoginResult lr = sfdc.login("username","password");

   // Reset the SOAP endpoint to the returned server URL
   sfdc.Url = lr.serverUrl;

   // Create a new session header object
   // Add the session ID returned from the login
   sfdc.SessionHeaderValue = new SessionHeader();
   sfdc.SessionHeaderValue.sessionId = lr.sessionId;
}
```

## Arguments

| Name | Type | Description |
|---|---|---|
| **username** | string | Login user name. |
| **password** | string | Login password associated with the specified username. |

## Response

LoginResult

## Fault

LoginFault

UnknownErrorFault

## See Also

*Sample SOAP Messages—login*
*About sforce API Calls on page 24*

### LoginResult

The login call returns a `LoginResult` object, which has the following properties:

| Name | Type | Description |
|---|---|---|
| **serverUrl** | string | URL of the sforce API server that will process subsequent sforce API calls. Your client application needs to define the target server. |
| **sessionID** | string | Unique ID associated with this session. Your client application needs to set this value in the session header. |
| **userID** | ID | ID of the user associated with the specified user name / password. If you want to retrieve information from your personal profile in the User object, you can pass this userID in the retrieve call. Alternatively, you can call getUserInfo to retrieve your personal profile information without this userID. |

## query

Executes a query against the specified object and returns data that matches the specified criteria.

### Syntax

```
QueryResult = sfdc.query(string queryString);
```

### Usage

Use the query call to retrieve data from an sforce API object. When a client application invokes the query call, it passes in a query expression that specifies the object to query, the fields to retrieve, and any conditions that determine whether a given object qualifies.

Upon invocation, the sforce API server executes the query against the specified object, caches the results of the query on the sforce API server, and returns a query response object to the client application. The client application can then use methods on the query response object to iterate through rows in the query response and retrieve information.

Your client application must be logged in with sufficient access rights to query individual objects within the specified object type and to query the fields in the specified field list. For more information, see Factors that Affect Data Access on page 24.

The query response object contains up to 2,000 rows of data. If the query results exceed 2,000 rows, then the client application uses the queryMore call and a server-side cursor to retrieve additional rows in 2000-row chunks. You can customize this option in the QueryOptions header, as described in Changing the Batch Size in Queries on page 20.

When querying for fields of type Base64 (see Base64 Field Type on page 15), the query response object returns only one record at a time. You cannot alter this by changing the batch size of the query call.

### Sample Code—Java

```
public void querySample() {

   QueryResult queryResult = null;
   // Set up query options. Set the max batch size to 3
   // so that we can exercise the queryMore call as well
   _QueryOptions queryOptions = new _QueryOptions();
   queryOptions.setBatchSize(new Integer(3));

   // Add the query options to the SOAP header
   binding.setHeader("SoapService", "QueryOptions", queryOptions);

   // Invoke the query call and save the results
   queryResult = binding.query("select FirstName, LastName from Contact");
   // Determine whether the query returned all the possible records
      if (queryResult.isDone()) {
      // Iterate through the records and process them
         for (int i = 0; i < queryResult.getRecords().length; i++) {
            Contact con = (Contact) queryResult.getRecords(i);
            String firstName = con.getFirstName();
            String lastName = con.getLastName();
            System.out.println("Contact " + (i + 1) + ": " + firstName + " " +
lastName);
         }
      }
      else {
         // Need to use queryMore call after processing
         // the first set of records from the query result
         while (queryResult.getRecords() != null) {
            for (int i = 0; i < queryResult.getRecords().length; i++) {
               Contact con = (Contact) queryResult.getRecords(i);
               String firstName = con.getFirstName();
               String lastName = con.getLastName();
               System.out.println("Contact " + (i + 1) + ": " + firstName + " " +
lastName);
            }
         // Invoke the queryMore call to get the next set of returned rows
         queryResult = binding.queryMore(queryResult.getQueryLocator());
      }
   }
}
```

## Sample Code—C#

```
private void contactQuery()
{
   // Set the query options (Optional; default batch size is 2000)
   binding.QueryOptionsValue = new QueryOptions();
   binding.QueryOptionsValue.batchSize = 10;
   binding.QueryOptionsValue.batchSizeSpecified = true;

   // Invoke the query call and save the result in a QueryResult
   QueryResult qr = binding.query("select FirstName, LastName from contact where
MailingPostalCode = '94062'");

   // Get the returned records
   sObject[] records = qr.records;

   // Determine whether some records where returned
   if (records.Length > 0)
   {
      bool done = false;  // Use this for loop control
      while (done = false)
      {
         for (inti=0;i<records.Length;i++)
         {
            Contact contact = (Contact)records[0];
            System.Diagnostics.Trace.WriteLine(contact.FirstName + " " +
contact.LastName);
         }
         // Update the loop control
         done = qr.done;
         // Determine whether we need to retrieve another batch of result records
         if (done == false)
         { qr = binding.queryMore(qr.queryLocator); }
         else
         { done = qr.done; }
      }
   }
   else
   {
      System.Diagnostics.Trace.WriteLine("no records matched criteria");
   }
}
```

## Arguments

| Name | Type | Description |
|---|---|---|
| **`queryString`** | string | Query string that specifies the object to query, the fields to return, and any conditions for including a specific object in the query. For more information, see sforce Object Query Language (SOQL) on page 17. |

## Response

QueryResult

## Fault

`MalformedQueryFault`

`InvalidSObjectFault`

`InvalidFieldFault`

`UnknownErrorFault`

## See Also

*queryMore on page 46*
*Sample SOAP Messages—query*
*sforce Object Query Language (SOQL) on page 17*
*About sforce API Calls on page 24*
*Changing the Batch Size in Queries on page 20*

## QueryResult

The query call returns a `QueryResult` object, which has the following properties:

| Name | Type | Description |
|---|---|---|
| **`queryLocator`** | QueryLocator | String. Used in queryMore for retrieving subsequent sets of objects from the query results, if applicable.<br><br>Represents a server-side cursor. Note that an salesforce.com account can have up to five (5) query cursors open at a time. |
| **`done`** | boolean | Indicates whether additional rows need to be retrieved from the query results (False) using queryMore, or not (True). Your client application can use this value as a loop condition while iterating through the query results. |
| **`records`** | sObject[] | Array of sObjects representing individual objects of the specified object type and containing data defined in the field list specified in the `queryString`. |
| **`size`** | int | Total number of rows retrieved in the query. Your client application can use this value to determine whether the query retrieved any rows (size > 0) or not (size = 0). |

### QueryLocator

In the QueryResult object returned by the query call, the queryLocator field contains a `QueryLocator` object that you will use in a subsequent queryMore call. Note that:

- You use a given `QueryLocator` only *once*. Each time you pass it in a queryMore call, the server returns a new `QueryLocator` in the QueryResult.
- `QueryLocator` objects expire automatically after 15 minutes of inactivity.

A `QueryLocator` represents a server-side cursor. A salesforce.com account can have up to five (5) query cursors open at a time. If five `QueryLocator` cursors are opened when a client application attempts open a new one, then the oldest of the five cursors is released.

## queryMore

Retrieves the next batch of objects from a query.

### Syntax

```
QueryResult = sfdc.queryMore(QueryLocator QueryLocator);
```

### Usage

You use queryMore to process query calls that retrieve a large number of records (more than 2000) in the result set. The query call retrieves the first 2000 records and creates a server-side cursor that is represented in the queryLocator object. The queryMore call processes subsequent records in up to 2000-record chunks, resets the server-side cursor, and returns a newly generated QueryLocator. To iterate through records in the result set, you generally call queryMore repeatedly until all records in the result set have been processed (the `Done` flag is True).

### Sample Code—Java

See the Sample Code—Java for the query call.

## Sample Code—C#

See the Sample Code—C# for the query call.

## Arguments

| Name | Type | Description |
|---|---|---|
| **`queryLocator`** | QueryLocator | Represents the server-side cursor that tracks the current processing location in the query result set. |

## Response

QueryResult

## Fault

`InvalidQueryLocatorFault`

`UnknownErrorFault`

## See Also

*query on page 43*
*Sample SOAP Messages—queryMore*
*About sforce API Calls on page 24*
*Changing the Batch Size in Queries on page 20*

## QueryResult

The queryMore call returns a `QueryResult` object, which has the following properties:

| Name | Type | Description |
|---|---|---|
| **`queryLocator`** | QueryLocator | String. Used in subsequent queryMore calls for retrieving sets of objects from the query results, if applicable. |
| **`done`** | boolean | Indicates whether additional rows need to be retrieved from the query results (False) using another queryMore call, or not (True). Your client application can use this value as a loop condition while iterating through the query results. |
| **`records`** | sObject[] | Array of sObjects representing individual objects of the specified object type and containing data defined in the field list specified in the `queryString`. |
| **`size`** | int | Total number of rows retrieved in the query. Your client application can use this value to determine whether the query retrieved any rows (size > 0) or not (size = 0). |

## QueryLocator

In the QueryResult object returned by the queryMore call, the queryLocator field contains a `QueryLocator` object that you will use in subsequent queryMore calls. Note that:

- You use a given `QueryLocator` only *once*. Each time you pass it in a queryMore call, the server returns a new `QueryLocator` in the QueryResult.
- `QueryLocator` objects expire automatically after 15 minutes of inactivity.

A `QueryLocator` represents a server-side cursor. A salesforce.com account can have up to five (5) query cursors open at a time. If five `QueryLocator` cursors are opened when a client application attempts open a new one, then the oldest of the five cursors is released.

## retrieve

Retrieves one or more objects based on the specified object IDs.

### Syntax

```
sObject[] result = sfdc.retrieve(string fieldList, string sObjectType, ID ids[]);
```

### Usage

Use the retrieve call to retrieve individual objects from an sforce API object. The client application passes the list of fields to retrieve, the object type, and an array of object IDs to retrieve.

In general, you use retrieve when you know in advance the IDs of the objects to retrieve. Use query instead to obtain objects when you do not know the IDs or when you want to specify other selection criteria.

Your client application must be logged in with sufficient access rights to retrieve individual objects within the specified object type and to retrieve the fields in the specified field list. For more information, see Factors that Affect Data Access on page 24.

### Sample Code—Java

```
private void retrieveSample() {
// Invoke the retrieve call and save results in an array of SObjects
SObject[] sObjects = binding.retrieve("Id, AccountNumber, Name, Website",
"Account", accounts);
// Verify that some objects were returned.
// Even though we began with valid object Ids,
// someone else might have deleted them in the meantime.
if (sObjects != null) {
   // Loop through the array and print out some properties
   for (int i=0;i<sObjects.length;i++) {
      // Cast the SObject into an Account object
      Account retrievedAccount = (Account)sObjects[i];
      System.out.println("Account: " + retrievedAccount.getId().getValue());
      System.out.println("    AccountNumber = " +
retrievedAccount.getAccountNumber());
      System.out.println("    Name          = " + retrievedAccount.getName());
      System.out.println("    Website       = " + retrievedAccount.getWebsite());
      }
   }
}
```

### Sample Code—C#

```
private void retrieve()
{
   // Invoke retrieve call and save results in an array of SObjects
   sObject[] records = binding.retrieve("FirstName, LastName", "Contact", new
String[] {"", ""});

   // Iterate through the results
   for (int i=0;i<records.Length;i++)
   {
      Contact contact = (Contact)records[i];
      // Get the contact properties
      System.Diagnostics.Trace.WriteLine("Name is: " + contact.FirstName + " " +
contact.LastName);
   }
}
```

### Arguments

| Name | Type | Description |
|---|---|---|
| **`fieldList`** | string | List of one or more fields in the specified object type, separated by commas. You must specify valid field names and must have read-level permissions to each specified field. The *`fieldList`* defines the ordering of fields in the `result`. |
| **`from`** | string | Object type from which to retrieve data. The specified value must be a valid object type for your organization. For a complete list of sforce object types, see List of Supported sforce Object Types on page 61. |
| **`ids`** | ID[] | Array of one or more IDs of the objects to retrieve. |

### Response

| Name | Type | Description |
|---|---|---|
| **`result`** | sObject[] | Array of one or more sObjects representing individual objects of the specified object type. |

### Fault

`InvalidSObjectFault`

`InvalidFieldFault`

`UnknownErrorFault`

### See Also

*Sample SOAP Messages—retrieve*
*About sforce API Calls on page 24*

## update

Updates one or more existing objects in your organization's data.

## Syntax

```
SaveResult[] = sfdc.update(sObject[] sObjects);
```

## Usage

Use update to update one or more existing objects, such as individual accounts or contacts, in your organization's data. The update call is analogous to the UPDATE statement in SQL.

When updating objects, consider the following rules and guidelines:

- Your client application must be logged in with sufficient access rights to update individual objects (as well as individual fields inside that object) within the specified object type. For more information, see Factors that Affect Data Access on page 24.
- Certain objects—and certain fields within those objects—require special handling or permissions. For example, you might also need permissions to access this object's parent object. Before you attempt to update a particular object, be sure to read its description in Chapter 5: sforce Objects on page 59.
- You cannot update ID fields.
- The sforce API server updates certain fields automatically, such as **LastModifiedDate**, **LastModifiedById**, and **SystemModstamp**. You cannot explicitly specify these values in your update call.
- To reset a field value to null, you add the field name to the fieldsToNull array in the sObject. You cannot set required fields to null.
- You must supply values that are valid for the field's data type, such as integers (not alphabetic characters) for integer fields. In your client application, follow the data formatting rules specified for your programming language and development tool (your development tool will handle the appropriate mapping of data types in SOAP messages).

Updating objects involves the following basic steps:

1. Determine the ID of each object that you want to update. For example, you might call query to retrieve a set of objects (with their IDs), based on specific criteria, that you would want to update. If you know the ID of the object that you want to update, you can call retrieve instead.
2. For each object, populate its fields with the data that you want to update.
3. Construct an sObject[] array and populate that array with the objects that you want to update. All objects *must* be of the same object type.
4. Call update, passing in the sObject[] array.
5. Process the results in the SaveResult[] object to verify whether the objects have been successfully updated.

## Sample Code—Java

```
public void updateAccountSample() {

   // Create an array of SObjects to send to the update method
   SObject[] updates = new SObject[2];

   // This account could also be from the results of a retrieve or query call
   Account updateAccount = new Account();
   updateAccount.setId(new ID("001x00000000JerAAE"));
   updateAccount.setName("New Account Name from Update Sample");
   updates[0] = updateAccount;

   Account updateAccount2 = new Account();
   updateAccount2 = new Account();
   updateAccount2.setId(new ID("001x00000000JesAAE"));
   updateAccount2.setWebsite("www.website.com");
   updates[1] = updateAccount2;

   // Invoke the update call and save the results
   SaveResult[] saveResults = binding.update(updates);
   print("\nPress the RETURN key to continue...", false);
}
```

## Sample Code—C#

```
private void update()
{
   // You would typically retrieve an SObject, modify its properties,
   // and then send the objects up in an array. For this sample,
   // we create a new contact object to update by setting
   // the id to a valid contact id
   Contact contact = new Contact();
   contact.Id = "";  // This should be a valid ID
   contact.MailingCity = "new city";
   contact.MailingPostalCode = "98776";

  // Invoke the update call, saving the results in SaveResult
   SaveResult[] sr = binding.update(new sObject[]{contact});

   // The SaveResult should never be empty
   for (int i=0;i<sr.Length;i++)
   {
      // Determine whether the row update succeeded
      if (sr[i].success)
      {
         // Get the ID of the updated row
   System.Diagnostics.Trace.WriteLine(sr[i].id);
      }
      else
      {
         // Iterate through the errors
         Error[] errors = sr[i].errors;
         for (int j=0;j<errors.Length;j++)
         {
            System.Diagnostics.Trace.WriteLine(errors[j].message);
         }
      }
   }
}
```

## Arguments

| Name | Type | Description |
|---|---|---|
| **sObjects** | sObject[] | Array of one or more objects to update. The sforce API server updates these objects in array index order. |

## Response

SaveResult[]

## Fault

`InvalidSObjectFault`

`UnknownErrorFault`

## See Also

*Sample SOAP Messages—update*
*About sforce API Calls on page 24*

## SaveResult

The update call returns an array of `SaveResult` objects. Each element in the `SaveResult` array corresponds to the sObject[] array passed as the `sObjects` parameter in the update call. For example, the object returned in the first index in the `SaveResult` array matches the object specified in the first index of the sObject[] array.

A `SaveResult` object has the following properties:

| Name | Type | Description |
|---|---|---|
| **`id`** | ID | ID of an sObject that you attempted to update. |
| **`success`** | boolean | Indicates whether the update call succeeded (True) or not (False) for this object. |
| **`errors`** | Error[] | If an error occurred during the update call, an array of one or more Error objects providing the error code and description. |

# CHAPTER 4: sforce Utility API Calls

This topic describes sforce utility API calls that your client applications can invoke to obtain the server timestamp, user information, and change user passwords. For a complete list of all sforce API calls, see List of Supported sforce API Calls on page 28.

The following table lists the sforce utility API calls described in this topic:

**Table 8: Supported Utility Calls in the sforce API**

| Task / Call | Description |
|---|---|
| getServerTimestamp | Retrieves the current system timestamp from the sforce API server. |
| getUserInfo | Retrieves personal information for the user associated with the current session. |
| resetPassword | Changes a user's password to a server-generated value. |
| setPassword | Sets the specified user's password to the specified value. |

## getServerTimestamp

Retrieves the current system timestamp from the sforce API server.

### Syntax

```
dateTime timestamp = sfdc.getServerTimestamp(null);
```

### Usage

Use getServerTimestamp to obtain the current system timestamp from the sforce API server. You might do this if, for example, you need to use the exact timestamp for timing or data synchronization purposes. When you create or update an object, the sforce API server uses the system timestamp to update the **`CreatedDate`** and **`LastModifiedDate`** fields, respectively, in the object.

### Sample Code—Java

```
public void getServerTimestampSample() {
   // Invoke the getServerTimestamp call and save the results
   GetServerTimestampResult serverTimestampResult =
sfdc.getServerTimestamp(null);
   System.out.println("Server Timestamp: " +
serverTimestampResult.getTimestamp());
}
```

### Sample Code—C#

```
private void getServerTimeStamp()
{
   // Invoke the getServerTimeStamp call and save the results
   GetServerTimestampResult ts = binding.getServerTimestamp();
   // Write the server timestamp to the diagnostics window
   System.Diagnostics.Trace.WriteLine(ts.timestamp.ToUniversalTime);
}
```

### Arguments

None.

### Response

| Name | Type | Description |
|---|---|---|
| **`timestamp`** | dateTime | System timestamp of the sforce API server when the getServerTimestamp call was executed. |

### Fault

`UnknownErrorFault`

### See Also

*Sample SOAP Messages—getServerTimestamp*
*sforce Utility API Calls on page 53*

## getUserInfo

Retrieves personal information for the user associated with the current session.

### Syntax

```
GetUserInfoResult result = sfdc.getUserInfo(null);
```

### Usage

Use getUserInfo to obtain personal information about the currently logged in user. The getUserInfo call is a convenience API call that retrieves and aggregates common profile information that your client application can use for display purposes, performing currency calculations, and so on.

The getUserInfo call applies only to the user name under which your client application has logged in. To retrieve additional personal information not found in the GetUserInfoResult object, you can call retrieve on the User object and pass in the `userID` returned by this call. To retrieve personal information about other users, you could call retrieve (if you know their user ID) or query on the User object.

### Sample Code—Java

```
public void getUserInfoSample() {

   GetUserInfoResult getUserInfoResult = null;

   // Invoke the getUserInfo call
   getUserInfoResult = binding.getUserInfo("fil@fil.com");
   // Display the returned user information
   System.out.println("User's currency symbol: " +
                    getUserInfoResult.getCurrencySymbol());
   System.out.println("User's organization name: " +
                    getUserInfoResult.getOrganizationName());
   System.out.println("User's default currency code: " +
                    getUserInfoResult.getUserDefaultCurrencyIsoCode());
   System.out.println("User's email: " + getUserInfoResult.getUserEmail());
   System.out.println("User's full name: " +
getUserInfoResult.getUserFullName());
   System.out.println("User's user id: " + getUserInfoResult.getUserId());
   System.out.println("User's language: " + getUserInfoResult.getUserLanguage());
   System.out.println("User's locale: " + getUserInfoResult.getUserLocale());
   System.out.println("User's timezone: " + getUserInfoResult.getUserTimeZone());
   System.out.println("User's org is multi currency: " +
                    getUserInfoResult.isOrganizationMultiCurrency());
}
```

## Sample Code—C#

```
private void getUserInfo()
{
   // Invoke getUserInfo call and save the results in getUserInfoResult
   GetUserInfoResult ui = binding.getUserInfo();
   // Get some of the user information
   String orgName = ui.organizationName;
   String userFullName = ui.userFullName;
}
```

## Arguments

None.

## Response

GetUserInfoResult

## Fault

`UnknownErrorFault`

## See Also

*Sample SOAP Messages—getUserInfo*
*sforce Utility API Calls on page 53*

## GetUserInfoResult

The getUserInfo call returns a `GetUserInfoResult` object, which has the following properties:

| Name | Type | Description |
|---|---|---|
| **`userID`** | ID | User ID. |
| **`userFullName`** | string | User's full name. |

| Name | Type | Description |
|---|---|---|
| **userEmail** | string | User's email address. |
| **userTimeZone** | string | User's time zone. |
| **userLocale** | string | User's locale (language and country). |
| **userLanguage** | string | User's language. |
| **userDefaultCurrencyIsoCode** | string | Default currency ISO code. Applicable only when **organizationMultiCurrency** is True.<br>When the logged in user creates any objects that have a currency ISO code, the server uses this currency ISO code if it is not explicitly specified in the create call. |
| **organizationName** | string | Name of the user's organization or company. |
| **organizationMultiCurrency** | boolean | Indicates whether the user's organization uses multiple currencies (True) or not (False). |
| **currencySymbol** | string | Currency symbol to use for displaying currency values. Applicable only when **organizationMultiCurrency** is False. |

## resetPassword

Changes a user's password to a server-generated value.

### Syntax

```
string password = sfdc.resetPassword(ID userID);
```

### Usage

Use resetPassword to request that the sforce API server change a user's password and return the server-generated password string. Use setPassword instead if you want to set the password to a specific value.

Your client application must be logged in with sufficient access rights to change the password for the specified user. For more information, see Factors that Affect Data Access on page 24.

### Sample Code—Java

```
public void resetPasswordSample() {

   // Specify the user ID of the password to reset
   String idToReset = "005x00000001ZPH";

   // Invoke the resetPasswordResult call
   ResetPasswordResult resetPasswordResult = binding.resetPassword(new
ID(idToReset));

   // Display the new server-generated password
   System.out.println(resetPasswordResult.getPassword());
}
```

### Sample Code—C#

```
private void resetPassword()
{
   // Invoke resetPassword call and save results in ResetPasswordResult
   ResetPasswordResult rpr = binding.resetPassword("userID");
   // Get the generated password
   System.Diagnostics.Trace.WriteLine(rpr.password);
}
```

### Arguments

| Name | Type | Description |
|---|---|---|
| **`userID`** | ID | ID of the user whose password you want to reset. |

### Response

| Name | Type | Description |
|---|---|---|
| **`password`** | string | New password generated by the sforce API server. |

### Fault

`InvalidIdFault`

`UnknownErrorFault`

### See Also

*setPassword*
*Sample SOAP Messages—resetPassword*
*sforce Utility API Calls on page 53*

## setPassword

Sets the specified user’s password to the specified value.

### Syntax

```
SetPasswordResult setPasswordResult = sfdc.setPassword(ID userID string
password);
```

### Usage

Use setPassword to change a user’s password to a value that you specify. For example, a client application might prompt a user to specify a different password, and then invoke setPassword to change the user’s login password. Use resetPassword instead if you want to reset the password with an sforce API server-generated value.

Your client application must be logged in with sufficient access rights to change the password for the specified user. For more information, see Factors that Affect Data Access on page 24.

## Sample Code—Java

```
public void setPasswordSample() {

   // Specify the userID and new password
   String idToReset = "005x00020001ZPH";
   String newPassword = "bigsecret";

   // Invoke the setPassword call
   SetPasswordResult setPasswordResult = binding.setPassword(new ID(idToReset),
newPassword);
   // If the call fails, an exception is raised; otherwise, the return is empty.
}
```

## Sample Code—C#

```
private void setPassword()
{
   // Invoke setPassword call; returns nothing if successful
   binding.setPassword("userid", "newpassword");
}
```

## Arguments

| Name | Type | Description |
|---|---|---|
| **userID** | ID | ID of the user whose password you want to reset. |
| **password** | string | New password to use for the specified user. |

## Response

None.

## Fault

InvalidIdFault

UnknownErrorFault

## See Also

*resetPassword on page 56*
*Sample SOAP Messages—setPassword*
*sforce Utility API Calls on page 53*

# CHAPTER 5: sforce Objects

This topic describes all of the possible objects defined in the sforce API. It contains the following sections:

- About sforce Objects
- List of Supported sforce Object Types

## ABOUT SFORCE OBJECTS

In the sforce API, *objects* are data entities that represent your organization's information. For example, the Account object type represents accounts—companies and organizations involved with your business, such as customers, partners, and competitors. An Account object represents a single account.

### Access to Objects

While this topic describes all of the objects available in the sforce API, your applications are able to work with only the objects that you are authorized to access. Programmatic access to objects is determined by the objects that are defined in your Enterprise WSDL file, your organization configuration, and your security access (which is configured by your organization's system administrator in your personal profile). For more information, see Factors that Affect Data Access on page 24.

### Changes to Objects in the sforce API 2.5

This topic describes changes to the sforce objects (previously called *entities*) from the previous version of the sforce API.

#### New Objects

The following objects were added in version 2.5.

**Table 9: New Object Types in the sforce API 2.5**

| Object Type | Description |
|---|---|
| CurrencyType | Represents the currencies used by an organization for which the multi-currency feature is enabled. |
| MailMergeTemplate | Represents a mail merge template (a Microsoft Word document) used for performing mail merges for your organization. |
| OpportunityCompetitor | Represents a competitor on an Opportunity. |
| Scontrol | Represents an sforce control, which is custom content that is hosted by the server but executed by client applications. |

#### Removed Objects

The following objects have changed and are not supported in version 2.5.

- CustomFieldDefinition

- ProfileRecordType
- RecordTypePickList

See Also

*List of Supported sforce Object Types on page 61*

### Changed Objects

The following objects were changed in version 2.5.

- The Profile and Role objects are now read-write.
- The PriceBook object is now called Pricebook.

Other objects have changed as well. This document has been updated to reflect any visible changes.

## Common Fields in sforce Objects

Several fields are commonly found in sforce objects. Not all sforce objects have all of these fields, but most have at least some of them.

**Read-Only Fields.** The following fields are read-only fields commonly found in sforce objects. The sforce API server updates these fields automatically.

**Table 10: Common Fields in sforce Objects**

| Field | Data Type | Description |
|---|---|---|
| `Id` | ID | Globally unique ID of this field. See Id Field Type on page 15. |
| `CreatedById` | ID | ID of the User who created this object. Read-only. |
| `CreatedDate` | dateTime | Date and time when this object was created. Read-only. |
| `LastModifiedById` | ID | ID of the User who last updated this object. Read-only. |
| `LastModifiedDate` | dateTime | Date and time when this object was last modified. Read-only. |
| `SystemModStamp` | dateTime | Date and time when this record was last modified. Read-only. |

**OwnerID Fields.** Most objects have an `ownerID` field that is an object reference field to the user that owns that object. Ownership is an important concept that affects the security model and has other implications throughout the system. Any user can query the owner field for any record they can access. However, the `ownerID` field has limitations when being set:

- For most users and most objects, the `ownerID` field cannot be set directly upon insert. It is implicitly set to the current user when inserting an object.
- Administrators (those with the "Manage All Data" permission) can specify an `ownerID` when inserting or updating any object. The `ownerID` field value must be a valid user in the organization.
- When creating or updating a Case or Lead, an administrator can set the `ownerID` field to any valid User in the organization or to any valid queue of the appropriate type in the organization.
- Updating the `ownerID` field via the API changes only the owner of that record. The change of ownership does not cascade to associated records as it does when you transfer record ownership in the application.
- Updating the `ownerID` field on an account deletes the existing sharing information and reapplies the default sharing model and autoshare rules.

## LIST OF SUPPORTED SFORCE OBJECT TYPES

Table 11: Supported Object Types in the sforce API

| Object Type | Description |
|---|---|
| Account | Represents an individual account, which is an organization involved with your business (such as customers, competitors, and partners). |
| AccountShare | Represents a sharing entry on an Account. |
| AccountTeamMember | Represents a User who is a member of an Account team. |
| Attachment | Represents a file that a User has uploaded and attached to a parent object. |
| BusinessProcess | Represents a business process. |
| Campaign | Represents and tracks a marketing campaign, such as a direct mail promotion, webinar, or trade show. |
| CampaignMember | Represents the association between a Campaign and either a Lead or Contact. |
| Case | Represents a case, which is a customer issue such as a customer’s feedback, problem, or question. |
| CaseComment | Represents a comment that provides additional information about the associated Case. |
| CaseHistory | Represents historical information about changes that have been made to the associated Case. |
| CaseSolution | Represents the association between a particular Case and a particular Solution. |
| Contact | Represents a contact, which is an individual associated with your Accounts. |
| CurrencyType | Represents the currencies used by an organization for which the multi-currency feature is enabled. |
| Document | Represents a file that a user has uploaded. Unlike Attachment objects, Documents are not attached to a parent object. |
| Event | Represents a calendar appointment event. |
| Folder | Represents a repository for a Document, MailMergeTemplate, email template, or report. Only one type of item can be contained in a particular Folder. |
| Group | Represents a set of Users. |
| GroupMember | Represents a User or Group that is a members of a public group. |
| Lead | Represents a lead, which is a prospect or potential Opportunity. |

**Table 11: Supported Object Types in the sforce API (Continued)**

| Object Type | Description |
|---|---|
| MailMergeTemplate | Represents a mail merge template (a Microsoft Word document) used for performing mail merges for your organization. |
| Note | Represents a note, which is text associated with an Attachment, Contact, or Opportunity. |
| Opportunity | Represents an opportunity, which is a sale or pending deal. |
| OpportunityCompetitor | Represents a competitor on an Opportunity. |
| OpportunityContactRole | Represents the association between an Opportunity and a Contact, with a specified Role name applied to the contact. |
| OpportunityLineItem | Represents an opportunity line item, which is a member of the list of Products associated with an Opportunity, along with other information about those products on that opportunity. |
| OpportunityLineItemSchedule | Represents information about the quantity, revenue distribution, and delivery dates for a particular OpportunityLineItem. |
| OpportunityShare | Represents a sharing entry on an Opportunity. |
| OpportunityTeamMember | Represents an individual User on the sales team of a particular Opportunity. |
| Partner | Represents the association between two particular Accounts or between a particular Opportunity and an Account. |
| Pricebook | Represents a price book that contains the list of Products that your organization sells. |
| Product | Represents a product that your organization sells. A product is member of the list of items in a Pricebook. |
| Profile | Represents a profile, which defines a set of permissions to perform different operations, such as querying, adding, updating, or deleting information. |
| RecordType | Represents a record type. |
| Role | Represents a role in your organization. |
| Scontrol | Represents an sforce control, which is custom content that is hosted by the server but executed by client applications. |
| Solution | Represents a solution, which is a detailed description of a customer issue and the resolution of that issue. |
| Task | Represents a task. |
| User | Represents a user in your organization. |
| UserTeamMember | Represents a single User on the default sales team of another user. |

See Also
*About sforce Objects on page 59*

## Account

Represents an individual account, which is an organization involved with your business (such as customers, competitors, and partners).

### Fields

Account fields do not require any special handling. For a list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

### Usage

Use Account objects to query and manage accounts in your organization.

### See Also

*AccountShare on page 63*
*AccountTeamMember on page 65*
*About sforce Objects on page 59*

## AccountShare

Represents a sharing entry on an Account.

### Fields

**Table 12: AccountShare Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. |
| **`AccountId`** | ID | ID of the Account associated with this sharing entry. This field cannot be updated. |
| **`UserOrGroupId`** | ID | ID of the User or Group that has been given access to the Account. This field cannot be updated. |
| **`AccountAccessLevel`** | string | Level of access that the User or Group has to the Account. One of the following values:<br>• **None** - User or Group cannot access the Account.<br>• **Read** - User or Group can only view the Account.<br>• **Edit** - User or Group can view or edit the Account.<br>• **All** - User or Group can view, edit, delete, and share the Account with other Users. This value is not valid for create or update calls.<br>This field must be set to an access level that is at least equal to the organization's default Account access level. In addition, either this field or the **`OpportunityAccessLevel`** field must be set higher than the organization's default access level for Accounts and opportunities. |

Table 12: AccountShare Fields (Continued)

| Field | Data Type | Description |
|---|---|---|
| **OpportunityAccessLevel** | string | Level of access that the User or Group has to opportunities associated with the Account. One of the following values:<br>• **None** - User or Group cannot access the associated opportunities.<br>• **Read** - User or Group can only view the associated opportunities.<br>• **Edit** - User or Group can view or edit the associated opportunities.<br>This field must be set to an access level that is at least equal to the organization's default opportunityAccessLevel. In addition, either this field or the **AccountAccessLevel** field must be set higher than the organization's default access level for accounts and opportunities.<br>This field cannot be updated via the API if the **AccountAccessLevel** field is set to "All."<br>Using the sforce API, you cannot update this field for the associated Account owner. You must update the Account owner's opportunityAccessLevel via the salesforce.com user interface. |
| **RowCause** | string | Reason that this sharing entry exists. Read-only. One of the following values:<br>• **Owner**—The User is the owner of the Account or is in a Role above the Account owner in the role hierarchy.<br>• **Manual**—The User or Group has access because a User with "All" access manually shared the Account with them.<br>• **Rule**—The User or Group has access via an Account sharing rule.<br>• **ImplicitParent**—The User or Group has separate access to an Opportunity associated with this Account, and so they are automatically given "Read" access to the Account.<br>• **Team**—The User or Group has team access (is an AccountTeamMember). |
| **LastModifiedById** | ID | ID of the User who last updated this AccountShare entry. |
| **LastModifiedDate** | dateTime | Date and time when this AccountShare entry was last modified. |

## Usage

The AccountShare object allows you to determine which users and groups can view and/or edit Accounts owned by other users. For more information, see Sharing on page 11.

If you attempt to insert an AccountShare that matches an existing AccountShare record, the create call updates any modified fields and returns the existing record.

### See Also

*Account on page 63*
*Group on page 77*
*OpportunityShare on page 88*
*About sforce Objects on page 59*

## AccountTeamMember

Represents a User who is a member of an Account team.

### Fields

**Table 13: AccountTeamMember Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. |
| **`AccountId`** | ID | ID of the Account to which this user is a team member. Must be a valid Account ID. Required. |
| **`UserId`** | ID | ID of the User who is a member of this account team. Must be a valid User ID. Required. |
| **`TeamMemberRole`** | string | Role associated with this team member. One of the valid team member roles defined for your organization. Required. |
| **`SystemModStamp`** | dateTime | Date and time when this record was last modified. Read-only. |

### Usage

Use the AccountTeamMember object to manage members of a particular Account and to specify roles for those users on that account. This object is available only for Enterprise Edition users who have enabled the account team preference.

### See Also

*About sforce Objects on page 59*

## Attachment

Represents a file that a User has uploaded and attached to a parent object.

### Fields

**Table 14: Attachment Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Name`** | string | Name of the attached file. |

**Table 14: Attachment Fields (Continued)**

| Field | Data Type | Description |
|---|---|---|
| **ParentId** | ID | ID of the parent object of the Attachment. The following object types are supported as parents of Attachments:<br>• Account<br>• Campaign<br>• Case<br>• Contact<br>• Opportunity<br>• Solution |
| **Body** | base64Binary | Encoded file data. |
| **BodyLength** | int | Size of the file (in bytes). |
| **Private** | boolean | Indicates whether the Attachment is viewable only by the owner and administrators. If the private field is set to True for an Attachment, the Attachment can be viewed only by the Attachment owner and administrators. During a create or update call, it is possible to mark an Attachment as "private" even if you are not the Attachment owner. This can result in a situation in which you can no longer access the Attachment that you just inserted or updated. |
| **OwnerId** | string | ID of the User who owns the Attachment. |
| **CreatedById** | ID | ID of the User who created this Attachment. Read-only. |
| **CreatedDate** | dateTime | Date and time when this Attachment was created. Read-only. |
| **LastModifiedById** | ID | ID of the User who last updated this Attachment. Read-only. |
| **LastModifiedDate** | dateTime | Date and time when this Attachment was last modified. Read-only. |

All of the Attachment fields are accessible in the describeSObject and query calls. Using the create call, you can insert the **Name**, **ParentId**, **Body**, **Private**, and **OwnerId** fields. For modifying Attachments, the update call gives you access to change the **Name**, **ParentId**, **Private**, and **OwnerId** fields.

You can access all of the Attachment fields in a query call. However, if you query the **Body** field, the response returns only the first Attachment from the result list. You cannot receive the **Body** field for multiple Attachments in a single query call.

## Usage

Your client application can invoke the describeSObject, create, update, query, and delete calls on Attachment objects. The sforce API sends and receives the binary file attachment data encoded as a base64Binary data type. Prior to create, clients must encode the binary attachment data as base64. Upon receiving an API response, clients must decode the base64 data to binary (this conversion is usually handled for you by the SOAP client).

The create call restricts Attachments to a maximum size of 5MB. For a file attached to a Solution, the limit is 1.5MB. The maximum email attachment size is also 1.5MB.

Note that the API supports attachments on emails in create, update, and delete calls. The query call does not return attachments parented by emails, unless the user performing the query has the "Modify All Data" permission.

### See Also

*About sforce Objects on page 59*

## BusinessProcess

Represents a business process.

### Fields

**Table 15: Business Process Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. |
| **`Name`** | string | Name of this business process. Up to 80 characters. |
| **`Description`** | string | Description of this business process. Up to 255 characters. |
| **`IsActive`** | boolean | Indicates whether this BusinessProcess can be presented to users in the salesforce.com user interface when creating a new record type or changing the business process of an existing record type. |
| **`CreatedById`** | ID | ID of the User who created this BusinessProcess. Read-only. |
| **`CreatedDate`** | dateTime | Date and time when this BusinessProcess was created. Read-only. |
| **`LastModifiedById`** | ID | ID of the User who last updated this BusinessProcess. Read-only. |
| **`LastModifiedDate`** | dateTime | Date and time when this BusinessProcess was last modified. Read-only. |
| **`SystemModstamp`** | dateTime | Date and time when this record was last modified. Read-only. |

### Usage

Use the BusinessProcess object to offer different subsets of picklist values to different users for the **`Lead Status`**, **`Case Status`**, and **`Opportunity Stage`** fields. Similar to a RecordType, a BusinessProcess identifies the type of a row in a Case, Lead, or Opportunity and implies a subset of picklist values for these three fields. The values for the remaining picklist fields are driven off of RecordType.

Your client application can invoke the describeSObject and query calls on BusinessProcess objects. Business processes are read-only in the sforce API.

### See Also

*About sforce Objects on page 59*

## Campaign

Represents and tracks a marketing campaign, such as a direct mail promotion, webinar, or trade show.

### Fields

The Campaign statistics fields are read-only (as in the salesforce.com user interface). You cannot update the statistics via the sforce API.

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

### Usage

Using the sforce API, you can create, update, delete, and query standard and custom fields for a Campaign.

The Campaign object is defined only for those organizations that have the Marketing feature enabled and valid Marketing licenses. In addition, it is accessible only to those users that are enabled as Marketing Users. If the organization does not have the Marketing feature or valid Marketing licenses, the Campaign object type does not appear in the describeGlobal call, and you cannot use describeSObject or query with the Campaign object.

*NOTE* The main constituents of campaigns are CampaignMember. You will commonly need to update campaigns with CampaignMember. See CampaignMember on page 68.

### See Also

*About sforce Objects on page 59*

## CampaignMember

Represents the association between a Campaign and either a Lead or Contact.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

The **`Status`** field is a picklist that directly controls the **`Responded`** flag on a CampaignMember. You cannot directly set the **`Responded`** flag, as it is read-only, but you can set it indirectly by setting the **`Status`** field. Each predefined **`Status`** field value implies a **`Responded`** flag value. Each time you update the **`Status`** field, you implicitly update the **`Responded`** flag on the CampaignMember.

In the salesforce.com user interface, Marketing Users can define the valid CampaignMember status values for the **`Status`** picklist. They can choose one status as the “default” CampaignMember status. For each **`Status`** field value, they can also select which values should be counted as “Responded,” meaning that the **`Responded`** flag will be set to True for those **`Status`** values.

### Usage

Each CampaignMember record has a unique ID. Each individual CampaignMember record must contain either a contactId or a leadId, but cannot contain both. Any attempt to create a single CampaignMember with both a contactId and a leadId results in an error. However, you can create separate CampaignMember records on a Campaign, one for the Lead and one for the Contact.

The CampaignMember object is defined only for those organizations that have the Marketing feature and valid Marketing licenses. In addition, the object is accessible only to those users that are enabled as Marketing Users. If the organization does not have the Marketing feature or valid Marketing licenses, the CampaignMember object type does not appear in the describeGlobal call, and you cannot use describeSObject or query with the CampaignMember object.

#### Inserting, Updating, and Deleting Campaign Members

You can indirectly update CampaignMembers by sending a create request. A create call for CampaignMembers is interpreted as an auto-insert-or-update call. The sforce API automatically determines whether a CampaignMember exists with the specified Campaign Id and Contact or Lead Id. If the CampaignMember does not exist for the given contact or lead Id, then a create is performed. If the CampaignMember already exists, the call is interpreted as an update and the **`Status`** field and **`Responded`** flag on the existing record are updated. Thus, you cannot create duplicate CampaignMember records, since any attempt to create a duplicate record simply updates the existing record.

During a create or update call, the sforce API verifies whether the **`Status`** field value specified in the call is a valid CampaignMember status for the given Campaign. If the specified **`Status`** value is a valid CampaignMember status, the API assigns that value to the CampaignMember **`Status`** field and updates the **`Responded`** flag with the associated value. If the specified **`Status`** value is not a valid CampaignMember status, the API assigns the default CampaignMember status to the **`Status`** field and updates the **`Responded`** flag with the associated value. However, if the given Campaign does not have a default CampaignMember status, the API assigns the value specified in the call to the **`Status`** field, and the **`Responded`** flag is set to False.

### See Also

*About sforce Objects on page 59*

## Case

Represents a case, which is a customer issue such as a customer’s feedback, problem, or question.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

#### Case Number Field

The **`CaseNumber`** field is assigned automatically when each Case is inserted. It cannot be set directly, and it cannot be modified after the Case is created.

#### Create-Only Fields

Cases have several fields that can only be set when calling create. They cannot be updated after the Case has been created. Those fields are **`Name`**, **`Email`**, **`Phone`**, and **`Company`**.

#### Status Field and Closed Flag

The **`Status`** field is a picklist that also directly controls the **`Closed`** flag. You cannot directly set the **`Closed`** flag, but you can set it indirectly by setting the **`Status`** field. Each predefined **`Status`** field value implies a **`Closed`** flag value.

#### Escalated Flag

Cases can have a special status called “escalated.” (See the salesforce.com online help documentation for more information on Case escalation.) A Case's escalated state does not affect how you can use a Case, or whether you can query, update, or delete it. However, you cannot set the **Escalated** flag via the sforce API.

### Description Field

The Case **Description** field is a special text field. The field description indicates a maximum length of 4000, but it applies in a special way to the Case **Description** field. Normally, a field length is expressed in characters, but the actual field length may be longer when measured in bytes, depending on which character encoding is used. However, the Case **Description** field length of 4000 indicates the maximum length of the field in bytes. The character encoding is indicated by the **Encoding** field in the DescribeGlobalResult.

### Usage

Use the Case object to manage cases for your organization.

### See Also

*About sforce Objects on page 59*

## CaseComment

Represents a comment that provides additional information about the associated Case.

### Fields

**Table 16: CaseComment Fields**

| Field | Data Type | Description |
|---|---|---|
| **Id** | ID | Unique ID of this record. Read-only. |
| **ParentId** | ID | ID of the parent Case of the CaseComment. |
| **Published** | boolean | Indicates whether the CaseComment is visible to customers in the Self-Service portal.<br>This is the only field that can be updated via the sforce API. All of the other case CaseComment cannot be updated. |
| **CommentBody** | string | Text of the CaseComment. The maximum size of the comment body is 4000 bytes. |
| **CreatedById** | ID | ID of the User who created this CaseComment. Read-only. |
| **CreatedDate** | dateTime | Date and time when this CaseComment was created. Read-only. |

### Usage

In the salesforce.com user interface, comments are generally entered by users working on a particular Case. Client applications can invoke the describeSObject, create, query, update, and retrieve calls on the CaseComment object.

All users have access to create and view CaseComments in the salesforce.com user interface and when using the sforce API. In both the salesforce.com user interface and via the sforce API, CaseComments cannot be modified after insertion, except to update the **Published** field. You cannot delete CaseComments by any means.

### See Also

*About sforce Objects on page 59*

## CaseHistory

Represents historical information about changes that have been made to the associated Case.

### Fields

**Table 17: CaseHistory Fields**

| Field | Data Type | Description |
|---|---|---|
| `Id` | ID | Unique ID of this record. Read-only. |
| `CaseId` | ID | ID of the Case associated with the CaseHistory entry. |
| `Field` | string | Name of the case field that was modified, or a special value to indicate some other modification to the case. The possible values, in addition to the case field names, are:<br>• **ownerAssignment** - The owner of the case was changed.<br>• **ownerAccepted** - A User took ownership of a case from a queue.<br>• **ownerEscalated** - The owner of the case was changed due to case escalation.<br>• **external** - A User made the case visible to customers in the Customer Self-Service Portal. |
| `OldValue` | string | Previous value of the modified case field. |
| `NewValue` | string | New value of the modified case field. |
| `CreatedById` | ID | ID of the User who created this CaseHistory. Read-only. |
| `CreatedDate` | dateTime | Date and time when this CaseHistory was created. Read-only. |

### Usage

Your client application can invoke the describeSObject and query calls on the CaseHistory object. The CaseHistory object is always read-only in salesforce.com. Case history entries are indirectly created by modifying a case via the salesforce.com user interface or the sforce API.

### See Also

*About sforce Objects on page 59*

## CaseSolution

Represents the association between a particular Case and a particular Solution.

### Fields

**Table 18: CaseSolution Fields**

| Field | Data Type | Description |
|---|---|---|
| `Id` | ID | Unique ID of this record. Read-only. |
| `CaseId` | ID | ID of the Case associated with the Solution. |
| `SolutionId` | ID | ID of the Solution associated with the case. |
| `CreatedById` | ID | ID of the User who created this CaseSolution. Read-only. |
| `CreatedDate` | string | Date and time when this CaseSolution record was created. Read-only. |

### Usage

Your client application can invoke the describeSObject, create, delete, and query calls on CaseSolution objects. You cannot update CaseSolutions via the sforce API.

If you attempt to insert a CaseSolution that matches an existing CaseSolution record, the create call simply returns the existing record.

### See Also

*About sforce Objects on page 59*

## Contact

Represents a contact, which is an individual associated with your Accounts.

### Fields

Contact fields do not require any special handling. For a list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

### Usage

Use the Contact object to manage individuals who are associated with Accounts in your organization.

### See Also

*About sforce Objects on page 59*

## CurrencyType

Represents the currencies used by an organization for which the multi-currency feature is enabled.

## Fields

Table 19: CurrencyType Fields

| Field | Data Type | Description |
|---|---|---|
| `Id` | ID | Unique ID of this record. Read-only. |
| `IsoCode` | string | ISO code of the currency. Required. Must be one of the valid alphabetic, three-letter currency ISO code defined by the ISO 4217 standard, such as USD, GBP, or JPY. Must be unique within your organization. |
| `ConversionRate` | double | Conversion rate of this currency type against the corporate currency. |
| `DecimalPlaces` | int | For this currency, specifies the number of digits to the right of the decimal point, such as zero (0) for JPY or 2 for USD. Required. |
| `IsActive` | boolean | Indicates whether this currency type is active (True) or not (False). Inactive currency types do not appear in picklists in the salesforce.com user interface. |
| `IsCorporate` | boolean | Indicates whether this currency type is the corporate currency (True) or not (False). Required. All other currency conversion rates are applied against this corporate currency. If a currency is already defined as the corporate currency in the salesforce.com user interface, it cannot be unset.<br><br>When a non-corporate currency is set to a corporate currency, the system will reconfigure all conversion rates based on the new corporate currency. |
| `CreatedById` | ID | ID of the User who created this CurrencyType. Read-only. |
| `CreatedDate` | dateTime | Date and time when this CurrencyType was created. Read-only. |
| `LastModifiedById` | ID | ID of the User who last updated this CurrencyType. Read-only. |
| `LastModifiedDate` | dateTime | Date and time when this CurrencyType was last modified. Read-only. |
| `SystemModStamp` | dateTime | Date and time when this record was last modified. Read-only. |

## Usage

For multi-currency organizations only, use the CurrencyType object to define the currencies that your organization uses. This object is not available in single-currency organizations.

Your client application cannot delete a CurrencyType object. In addition, you need "Customize salesforce.com" permission to edit a CurrencyType.

### See Also

*About sforce Objects on page 59*

## Document

Represents a file that a user has uploaded. Unlike Attachment objects, Documents are not attached to a parent object.

### Fields

**Table 20: Document Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. |
| **`FolderId`** | ID | ID of the Folder that contains the Document. See Folder on page 76. |
| **`Name`** | string | Name of the Document. |
| **`Type`** | string | File type of the Document. In general, the values match the file extension for the type of Document, e.g., "pdf" or "jpg." |
| **`Body`** | base64Binary | Encoded file data. |
| **`BodyLength`** | int | Size of the file (in bytes). |
| **`URL`** | string | URL reference to the file (instead of storing it in the database). |
| **`Description`** | string | Text description of the Document. |
| **`Author`** | string | ID of the User who is responsible for the Document. |
| **`CreatedById`** | ID | ID of the User who first uploaded this Document. Read-only. |
| **`CreatedDate`** | dateTime | Date and time when this Document was first uploaded. Read-only. |
| **`LastModifiedById`** | ID | ID of the User who last updated this Document. Read-only. |
| **`LastModifiedDate`** | dateTime | Date and time when this Document was last modified. Read-only. |

### Usage

Your client application can invoke the describeSObject, create, update, query, and delete calls on Document objects. You must have the "Edit Documents" permission and the appropriate access to the Folder that contains a document in order to insert or update a Document in that Folder.

#### Encoded Data

The sforce API sends and receives the binary file data encoded as a base64Binary data type. Prior to create, clients must encode the binary file data as base64. Upon receiving an API response, clients must decode the base64 data to binary (this conversion is usually handled for you by the SOAP client).

#### Maximum Document Size

The create and update calls restrict documents to a maximum size of 5MB.

### See Also

*About sforce Objects on page 59*

## Event

Represents a calendar appointment event.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

#### whoId and whatId Fields

The Event object has **`WhoId`** and **`WhatId`** cross-reference ID fields. These are cross-reference fields that can each point to one of several other objects. The **`WhoId`** field can point to a Contact or Lead, and the **`WhatId`** field can point to an Account, Opportunity, Campaign, or Case. In addition, if the **`WhoId`** field refers to a Lead, then the **`WhatId`** field must be empty.

#### ActivityDateTime and ActivityDate Fields

The salesforce.com user interface has a single **`Due Date`** field for Events. However in the sforce API, the due date information is contained in either the **`ActivityDateTime`** field or the **`ActivityDate`** field, depending on the value of the Event **`IsAllDayEvent`** flag.

- **`ActivityDateTime`** - If the Event **`IsAllDayEvent`** flag is set to False (indicating that it is not an all day Event), then the Event due date information is contained in the **`ActivityDateTime`** field. This field is a regular Date/Time field with a relevant time portion. The time portion is always transferred in the GMT/UTC time zone. You need to translate the time portion to or from a local time zone for the user or the application, as appropriate. For more information, see DateTime Field Type on page 14.
- **`ActivityDate`** - If the Event **`IsAllDayEvent`** flag is set to True (indicating that it is an all day Event), then the Event due date information is contained in the **`ActivityDate`** field. This field is a date field with a timestamp that is always set to midnight in the GMT/UTC time zone. The timestamp is not relevant, and you should not attempt to alter it to account for any time zone differences. For more information, see Date Field Type on page 14.

When querying for events with a specific due date, you must filter on both the **`ActivityDateTime`** and **`ActivityDate`** fields. For example to find all events with a due date of February 14, 2003, you need two filters:

- one filter with the **`ActivityDate`** field equal to midnight GMT on February 14, 2003
- one filter with the **`ActivityDateTime`** field greater than or equal to midnight on February 14, 2003 in the user’s local time zone AND less than or equal to midnight on February 15, 2003 in the user’s local time zone

### Usage

Use Events to manage calendar appointments.

### Archived Activities

Sforce archives older events and Tasks according to the criteria listed below. In the salesforce.com user interface, users can view archived activities only in the **Printable View** or by clicking **View All** on the Activity History related list or by doing an advanced search. However in the sforce API, archived activities are not accessible.

Sforce archives activities according to the following criteria.

- Events with an **`ActivityDateTime`** or **`ActivityDate`** value greater than or equal to 365 days old
- Tasks with a **`Closed`** flag value of True and an **`ActivityDate`** value greater than or equal to 365 days old
- Tasks with a **`Closed`** flag value of True, a blank **`ActivityDate`** field, and a create date greater than or equal to 365 days ago

If you use the sforce API to insert activities that meet these criteria, the activities will be archived during the next run of the archival background process.

## See Also

*About sforce Objects on page 59*

## Folder

Represents a repository for a Document, MailMergeTemplate, email template, or report. Only one type of item can be contained in a particular Folder.

## Fields

**Table 21: Folder Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. |
| **`Name`** | string | Name of the Folder. |
| **`Type`** | string | Type of objects contained in the Folder. The **`Type`** field cannot be updated. One of the following values:<br>• **Document**<br>• **Email template**<br>• **Report** |
| **`AccessType`** | string | Indicates who can access the Folder. One of the following values:<br>• **Shared** - Folder is accessible only by Users in a particular Group or Role. The API does not allow you to view, insert, or update which group or Role the Folder is shared with.<br>• **Public** - Folder is accessible by all users.<br>• **Hidden** - Folder is hidden from everyone. |
| **`ReadOnly`** | boolean | Indicates whether you can add data to this Folder. |
| **`CreatedById`** | ID | ID of the User who created this Folder. Read-only. |
| **`CreatedDate`** | dateTime | Date and time when this Folder was created. Read-only. |
| **`LastModifiedById`** | ID | ID of the User who last updated this Folder. Read-only. |

**Table 21: Folder Fields (Continued)**

| Field | Data Type | Description |
|---|---|---|
| **LastModifiedDate** | dateTime | Date and time when this Folder was last modified. Read-only. |

## Usage

Your client application can invoke the describeSObject, create, update, query, and delete calls on Folder objects. You must have the "Modify All Data" permission to create, update, and delete document folders, email template folders, or report folders. To query Folders, no special permissions are needed.

## See Also

*About sforce Objects on page 59*

## Group

Represents a set of Users.

## Fields

**Table 22: Group Fields**

| Field | Data Type | Description |
|---|---|---|
| **Id** | ID | Unique ID of this record. Read-only. |
| **Name** | string | Name of the Group. |
| **RelatedId** | ID | For Groups of type "Role," the ID of the associated Role. Read-only. |
| **Type** | string | Type of the Group. One of the following values:<br>• **Regular** - A standard public Group. When you insert a Group, its type must be "Regular."<br>• **Role** - A public Group that includes all of the Users in a particular Role.<br>• **RoleAndSubordinates** - A public Group that includes all of the Users in a particular Role and all of the Users in Roles below that Role.<br>• **Organization** - A public Group that includes all of the Users in the organization. This Group is read-only. |
| **CreatedById** | ID | ID of the User who created this Group. Read-only. |
| **CreatedDate** | dateTime | Date and time when this Group was created. Read-only. |
| **LastModifiedById** | ID | ID of the User who last updated this Group. Read-only. |
| **LastModifiedDate** | dateTime | Date and time when this Group was last modified. Read-only. |

### Usage

Your client application can invoke the describeSObject, create, update, and query calls on Group objects. Groups cannot be deleted. Any User can access the Group object—no special permissions are needed.

Only public Groups are accessible via the sforce API. Personal Groups are not available.

### See Also

*GroupMember on page 78*
*About sforce Objects on page 59*

## GroupMember

Represents a User or Group that is a members of a public group.

### Fields

**Table 23: GroupMember Fields**

| Field | Data Type | Description |
|---|---|---|
| `Id` | ID | Unique ID of this record. Read-only. |
| `GroupId` | ID | ID of the Group. |
| `UserOrGroupId` | ID | ID of the User or Group that is a direct member of the group. |

### Usage

Your client application can invoke the describeSObject, create, delete, and query calls on GroupMember objects. GroupMembers cannot be updated. Any user can access the GroupMember object—no special permissions are needed.

A GroupMember record exists for every User or Group who is a direct member of a public group whose `Type` field is set to "Regular." Users who are indirect members of "Regular" public groups are not listed as group members. A User can be an indirect member of a group if he or she is in a Role above the direct group member in the hierarchy, or if he or she is a member of a group that is included as a subgroup in that group.

If you attempt to insert a GroupMember that matches an existing GroupMember record, the create call simply returns the existing record.

### See Also

*Group on page 77*
*About sforce Objects on page 59*

## Lead

Represents a lead, which is a prospect or potential Opportunity.

## Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

### Converted Leads

Leads have a special state to indicate that they have been converted into an Account, Contact, and Opportunity. (See the salesforce.com online help documentation for additional information on converting Leads.) You can convert Leads only through the salesforce.com user interface. Once a Lead has been converted, it is read-only. You cannot update or delete a converted Lead. However, you can query converted Leads using the query call.

Leads have several fields that indicate their converted status. These special fields are read-only via the sforce API. You cannot set these fields directly; they are set when converting the Lead in the salesforce.com user interface. The fields are:

**Table 24: Fields on Converted Leads**

| Field | Data Type | Description |
|---|---|---|
| **`Converted`** | boolean | Indicates whether the Lead has been converted. |
| **`ConvertedAccountId`** | ID | Object reference ID that points to the Account into which the Lead has been converted. |
| **`ConvertedContactId`** | ID | Object reference ID that points to the Contact into which the Lead has been converted. |
| **`ConvertedOpportunityId`** | ID | Object reference ID that points to the Opportunity into which the Lead has been converted. |

### Unread Leads

Leads have a special state to indicate that they have not been viewed or edited by the lead owner. In the salesforce.com user interface, this is helpful for users to know which leads have been assigned to them but which they have not touched yet. The **`IsUnreadByOwner`** field is True if the lead owner has not yet viewed or edited the lead, and False if the lead owner has viewed or edited the lead at least once.

### Lead Status Picklist

Each **`Lead Status`** picklist value corresponds to either a converted or unconverted status, as defined in the salesforce.com user interface. You cannot change a lead to have a Status that is marked as converted through the API.

## Usage

To update a Lead, you must have "Edit Leads" permission.

## See Also

*About sforce Objects on page 59*

## MailMergeTemplate

Represents a mail merge template (a Microsoft Word document) used for performing mail merges for your organization.

### Fields

**Table 25: MailMergeTemplate Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. |
| **`Name`** | string | Name of this mail merge template. |
| **`Description`** | string | Text description of this mail merge template. Up to 255 characters. |
| **`Filename`** | string | Filename of the Microsoft Word document that was uploaded as a mail merge template. Up to 255 characters in length. |
| **`BodyLength`** | int | Length of the Microsoft Word document. |
| **`Body`** | binary | Microsoft Word document to use as a mail merge template. Up to 5MB. Due to limitations with Microsoft Word mail merge templates, your client application can specify the Body field in the create call but not in the update call. |
| **`LastUsedDate`** | dateTime | Date and time when this MailMergeTemplate was last used. |
| **`CreatedById`** | ID | ID of the User who created this MailMergeTemplate. Read-only. |
| **`CreatedDate`** | dateTime | Date and time when this MailMergeTemplate was created. Read-only. |
| **`LastModifiedById`** | ID | ID of the User who last updated this MailMergeTemplate. Read-only. |
| **`LastModifiedDate`** | dateTime | Date and time when this MailMergeTemplate was last modified. Read-only. |
| **`SystemModStamp`** | dateTime | Date and time when this record was last modified. Read-only. |

### Usage

Use the MailMergeTemplate object to manage mail merge templates for your organization. All users can view a MailMergeTemplate, but you need “Customize salesforce.com” permissions to modify it.

### See Also

*About sforce Objects on page 59*

## Note

Represents a note, which is text associated with an Attachment, Contact, or Opportunity.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

The **Private** flag on a note is a field that influences the access rights for that note. If the note is marked private, only the note owner or a User with the "Modify All Data" permission can view the note or query it via the sforce API. Because of this, you can create an unusual situation for a regular user that does not have the "Modify All Data" permission. If a regular user sets the **Private** flag to True on a note that they do not own, then they can no longer query, update, or delete that note.

### Usage

Use the Note object to manage notes for an Attachment, Contact, or Opportunity.

### See Also

*About sforce Objects on page 59*

## Opportunity

Represents an opportunity, which is a sale or pending deal.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

#### StageName Field

The **StageName** field controls several other fields on an Opportunity. Each of the fields can be directly set or implied by changing the **StageName** field. In addition, the **StageName** field is a picklist, so it has additional members in the describeSObject response to indicate how it affects the other fields.

#### ForecastCategory

**ForecastCategory** is a restricted picklist field. It is implied, but not directly controlled, by the **StageName** field. You can override this field to a different value than is implied by the **StageName**.

The values of the **ForecastCategory** field are fixed enumerated values. The field labels are localized to the language of the user performing the operation, if localized versions of those labels are available for that language in the salesforce.com user interface.

#### IsClosed and IsWon Flags

The **IsClosed** and **IsWon** flags are directly controlled by the **StageName**. You can query and filter on these fields, but you cannot directly set them in a create or update request. Instead, you must set the **StageName** to a value that has the appropriate **IsClosed** and **IsWon** flags.

#### Probability Field

The Opportunity **Probability** field is implied, but not directly controlled, by the **StageName** field. You can override this field to a different value than what is implied by the **StageName**.

#### ExpectedRevenue Field

The **ExpectedRevenue** field is a read-only field that is equal to the product of the Opportunity **Amount** field and the **Probability**. You cannot directly set the **ExpectedRevenue** field, but you can indirectly set it by setting the **Amount** or **Probability** fields.

### Amount Field

The Opportunity **`Amount`** field is normally a regular field, but it becomes implicitly read-only if the Opportunity has any line items. Any attempt to update the **`Amount`** of an Opportunity that has line items will be ignored. The update call will not be rejected, and other fields will be updated as specified, but the **`Amount`** will be unchanged.

### Campaign Field

The Opportunity **`CampaignId`** field is a cross-reference field that points to a Campaign object. The **`CampaignId`** field is defined only for those organizations that have Campaigns enabled as a feature. The User must have read access rights to the cross-referenced Campaign object in order to create or update that campaign into the **`CampaignId`** field on the Opportunity.

### HasOpportunityLineItem Field

The Opportunity **`HasOpportunityLineItem`** field is a read-only field that indicates whether the Opportunity has associated line items. A value of True means that Opportunity line items have been created for the Opportunity.

### PricebookId Field

The Opportunity **`PricebookId`** field is a cross-reference field that points to a Pricebook object. The **`PricebookId`** field indicates which Pricebook applies to the specific Opportunity. The **`PricebookId`** field is defined only for those organizations that have Products enabled as a feature.

An Opportunity can only have Opportunity line items if the Opportunity has a Pricebook. The Opportunity line items must correspond to Products that are listed in the Opportunity's Pricebook. However, you can insert Opportunity line items on an Opportunity that does not have an associated Pricebook. For the first Opportunity line item that you insert on an Opportunity without a Pricebook, the API automatically sets the **`PricebookId`** field, if the Opportunity line item corresponds to a Product in an active Pricebook that has a **`CurrencyISOCode`** field that matches the **`CurrencyISOCode`** field of the Opportunity. If the Pricebook is not active or the **`CurrencyISOCode`** fields do not match, the API returns an error.

You cannot update the **`PricebookId`** field if Opportunity line items exist on the Opportunity. You must delete the line items before attempting to update the **`PricebookId`** field.

### Currency Field

The **`CurrencyISOCode`** field exists only for multi-currency organizations. If the organization does not have the multi-currency feature enabled, the **`CurrencyISOCode`** field is not accessible.

If the organization is multi-currency and a Pricebook is not specified on the Opportunity (i.e., the **`PricebookId`** field is blank), the **`CurrencyISOCode`** field can have any currency allowed by the organization.

If the organization is multi-currency and a Pricebook is specified on the Opportunity (i.e., the **`PricebookId`** field is not blank), then the currency value of the **`CurrencyISOCode`** field must match the currency of the Pricebook. If the Pricebook changes, the Opportunity currency must also be changed to be the same as the currency of the Pricebook.

## Usage

Use the Opportunity object to manage information about a sale or pending deal. To update an Opportunity, your client application needs “Edit Opportunities” permission.

## See Also

*OpportunityCompetitor on page 83*
*OpportunityContactRole on page 83*
*OpportunityLineItem on page 84*
*OpportunityLineItemSchedule on page 85*

*About sforce Objects on page 59*

## OpportunityCompetitor

Represents a competitor on an Opportunity.

### Fields

**Table 26: OpportunityCompetitor Fields**

| Field | Data Type | Description |
|---|---|---|
| **`OpportunityID`** | ID | ID of the associated Opportunity. |
| **`CompetitorName`** | string | Name of the competitor. |
| **`Strengths`** | string | Description of the competitor's strengths. |
| **`Weaknesses`** | string | Description of the competitor's weaknesses. |
| **`CreatedById`** | ID | ID of the User who created this OpportunityCompetitor. Read-only. |
| **`CreatedDate`** | dateTime | Date and time when this OpportunityCompetitor was created. Read-only. |
| **`LastModifiedById`** | ID | ID of the User who last updated this OpportunityCompetitor. Read-only. |
| **`LastModifiedDate`** | dateTime | Date and time when this OpportunityCompetitor was last modified. Read-only. |
| **`SystemModstamp`** | dateTime | Date and time when this record was last modified. Read-only. |

### Usage

Use the OpportunityCompetitor object to manage competitors on an Opportunity, associating multiple competitors on a opportunity and specifying the strengths and weaknesses of each competitor.

### See Also

*Opportunity on page 81*
*About sforce Objects on page 59*

## OpportunityContactRole

Represents the association between an Opportunity and a Contact, with a specified Role name applied to the contact.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

#### OpportunityId Field

The **`OpportunityId`** field of an OpportunityContactRole is non-nullable, and it cannot be updated. You must provide a value for the **`OpportunityId`** on create. You cannot change it after it has been inserted.

#### ContactId Field

The sforce API applies user access rights to the associated Opportunity for an OpportunityContactRole, but not to the associated Contact. As such, the API may return rows from an OpportunityContactRole query that include **`ContactId`** values for contacts that the user does not have sufficient rights to access. It may also return **`ContactId`** values for contacts that have been deleted. In either case, the client must perform a query on the contact table for that **`ContactId`** value to determine whether the contact is accessible to the user and has not been deleted.

### Usage

OpportunityContactRoles appear in the salesforce.com user interface on the Opportunity detail page. Like most other sforce objects, OpportunityContactRole records have their own unique ID that you use when updating or deleting an OpportunityContactRole.

You can create multiple relationships between the same Opportunity and a Contact. This action is not recommended, but the application does not prohibit it.

### See Also

*About sforce Objects on page 59*

## OpportunityLineItem

Represents an opportunity line item, which is a member of the list of Products associated with an Opportunity, along with other information about those products on that opportunity.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

#### Total Price Field

The **`TotalPrice`** field specified in a create or update call is the total price of the OpportunityLineItem. The unit price of an OpportunityLineItem is not accessible via the API. The salesforce.com user interface calculates the unit price as the total price of the OpportunityLineItem divided by the quantity listed for that line item. To insert the **`TotalPrice`** for an OpportunityLineItem via the API (given only a unit price and the quantity), calculate the **`TotalPrice`** field as the unit price multiplied by the quantity.

The **`TotalPrice`** field is read-only if the OpportunityLineItem has a revenue schedule. If the OpportunityLineItem does not have a schedule or only has quantity schedule, the **`TotalPrice`** field can be updated.

#### Quantity Field

The **`Quantity`** field on the OpportunityLineItem object is read-only if the OpportunityLineItem has a quantity schedule, a revenue schedule, or both a quantity and a revenue schedule.

#### Has Revenue Schedule and Has Quantity Schedule Fields

The **`HasRevenueSchedule`** field on the OpportunityLineItem object is read-only, and is True if a revenue schedule has been created for the OpportunityLineItem.

The **HasQuantitySchedule** field on the OpportunityLineItem object is read-only, and is True if a quantity schedule has been created for the OpportunityLineItem.

If the OpportunityLineItem has a revenue schedule, the **Quantity** and **TotalPrice** fields cannot be updated. In addition, the **Quantity** field cannot be updated if the OpportunityLineItem has a quantity schedule. The sforce API ignores any attempt to update these fields. The update call will not be rejected; the updated values will simply be ignored.

## Usage

The Opportunity can only have OpportunityLineItems if the opportunity has a Pricebook. An OpportunityLineItem must correspond to a Product that is listed in the opportunity's Pricebook. For information about inserting OpportunityLineItems for an Opportunity that does not have an associated Pricebook or any existing line items, see Effects on Opportunities on page 85.

The OpportunityLineItem object is defined only for those organizations that have Products enabled as a feature. If the organization does not have the Products feature, the OpportunityLineItem object type does not appear in the describeGlobal call, and you cannot use describeSObject or query with the OpportunityLineItem object.

### Permissions

The user must have edit access rights on the Opportunity in order to create or update opportunity line items on that opportunity.

### Effects on Opportunities

Opportunities that have associated OpportunityLineItems are affected in the following ways:

- Inserting an OpportunityLineItem increments the Opportunity **Amount** value by the **TotalPrice** of the OpportunityLineItem. Additionally, inserting an OpportunityLineItem increments the **Expected Amount** on the opportunity by the **TotalPrice** times the opportunity **Probability**.
- The opportunity **Amount** becomes a read-only field when the opportunity has line items. The API ignores any attempt to update this field on an opportunity with line items. The update call will not be rejected; the updated value will simply be ignored.
- You cannot update the **PricebookId** field or the **CurrencyISOCode** field on the opportunity if line items exist. The API rejects any attempt to update these fields on an opportunity with line items.
- When you create or update an OpportunityLineItem, the API verifies that the line item corresponds to a Product in the Pricebook that is associated with the opportunity. If the opportunity does not have an associated Pricebook, the API automatically sets the **PricebookId** field on the opportunity if the line item corresponds to a Product in an active Pricebook that has a **CurrencyISOCode** field that matches the **CurrencyISOCode** field of the opportunity. If the Pricebook is not active or the **CurrencyISOCode** fields do not match, the API returns an error.
- The opportunity **HasLineItem** field is set to True when an OpportunityLineItem is inserted for that opportunity.

## See Also

*OpportunityLineItemSchedule on page 85*
*About sforce Objects on page 59*

# OpportunityLineItemSchedule

Represents information about the quantity, revenue distribution, and delivery dates for a particular OpportunityLineItem.

## Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

### Id Field

The **`Id`** field uniquely identifies the OpportunityLineItemSchedule. You can specify the **`Id`** field for query, update and delete calls.

### Type Field

The **`Type`** field specifies the type of the schedule.You must specify a **`Type`** value when inserting an OpportunityLineItemSchedule. Valid values are "Quantity," "Revenue," or "Both." However, the allowable values for a particular OpportunityLineItemSchedule depend on the product-level schedule preferences and whether the line item has any existing schedules. The following criteria must be met:

- The Product on which the OpportunityLineItem is based must have the appropriate **`RevenueScheduleEnabled`** and/or **`QuantityScheduleEnabled`** flags set to True.
- When you create a schedule for a line item that does not have any existing schedules, you can specify any valid value.
- If you create a schedule for a line item that already has existing schedules, the new schedule must be consistent with the existing schedules. The following matrix outlines the allowable values:

**Table 27: Allowable type Values for Opportunity Line Item Schedules**

| Value of `HasRevenueSchedule` on line item | Value of `HasQuantitySchedule` on line item | Allowable `Type` Values |
|---|---|---|
| false | false | Revenue, Quantity, Both |
| false | true | Quantity |
| true | false | Revenue |
| true | true | Both |

### Quantity and Revenue Fields

The **`Quantity`** field specifies the total number of units to be scheduled in a quantity schedule. The **`Revenue`** field specifies the total price to be scheduled in a revenue schedule.

The allowable **`Quantity`** and **`Revenue`** field values depend on the value of the **`Type`** field:

**Table 28: Allowable quantity and revenue Values for Line Item Schedules**

| `Type` Value | Allowable `Quantity` Value | Allowable `Revenue` Value |
|---|---|---|
| Revenue | Null | Non-null |
| Quantity | Non-null | Null |
| Both | Non-null | Non-null |

The **`Quantity`** and **`Revenue`** fields have the following restrictions in the update call.

- For a schedule of **`Type`** "Quantity," you cannot update a null **`Revenue`** value to non-null. Likewise for a schedule of **`Type`** "Revenue," you cannot update a null **`Quantity`** value to non-null.
- You cannot null out the **`Quantity`** field for a schedule of **`Type`** "Quantity." Likewise you cannot null out the **`Revenue`** field for a schedule of **`Type`** "Revenue."
- You cannot null out either the **`Revenue`** or **`Quantity`** fields for a schedule of type "Both."

## Usage

Sforce supports two types of OpportunityLineItemSchedules:

- quantity schedules
- revenue schedules

The user must have edit access rights on the Opportunity in order to create or update line item schedules on that opportunity.

### Applies Only If Products or Annuities Features Are Enabled

The OpportunityLineItemSchedule object is defined only for those organizations that have the Products and Annuities features enabled. If the organization does not have the Products and Annuities features, the OpportunityLineItemSchedule object type does not appear in the describeGlobal call, and you cannot use describeSObject or query with the OpportunityLineItemSchedule object.

### Effects on Opportunities and Opportunity Line Items

OpportunityLineItemSchedules affect opportunities and opportunity line items in the following ways:

- Inserting an OpportunityLineItemSchedule of **`Type`** "Revenue" or "Quantity" increments the **`TotalPrice`** field on the OpportunityLineItem by the line item schedule **`Revenue`** amount. Inserting an OpportunityLineItemSchedule of **`Type`** "Quantity" or "Both" increments the **`Quantity`** field on the OpportunityLineItem by the line item schedule **`Quantity`** amount.
- The create call also affects the original opportunity as follows: 1) the Opportunity **`Amount`** is incremented the by OpportunityLineItemSchedule revenue amount; and 2) the Opportunity **`Expected Amount`** is incremented by the line item schedule amount multiplied by the Opportunity **`Probability`**.
- Deleting an OpportunityLineItemSchedule has a similar effect on the related OpportunityLineItem and Opportunity. Deleting a schedule decrements the OpportunityLineItem **`TotalPrice`** by the deleted line item schedule amount. The opportunity **`Amount`** is also decremented by the line item schedule amount, and the Opportunity **`Expected Amount`** is reduced by the line item schedule amount multiplied by the Opportunity **`Probability`**.

### Deleting an Opportunity Line Item Schedule

To delete an OpportunityLineItemSchedule, you must specify the **`Id`** in the delete call. Deleting the last remaining schedule will set the corresponding **`HasQuantitySchedule`** and/or **`HasRevenueSchedule`** flags to False on the parent line item.

## See Also

*OpportunityLineItem on page 84*
*Product on page 92*
*delete on page 31*
*About sforce Objects on page 59*

## OpportunityShare

Represents a sharing entry on an Opportunity.

### Fields

Table 29: OpportunityShare Fields

| Field | Data Type | Description |
|---|---|---|
| `Id` | ID | Unique ID of this record. Read-only. |
| `OpportunityId` | ID | ID of the Opportunity associated with this sharing entry. This field cannot be updated. |
| `UserOrGroupId` | ID | ID of the User or Group that has been given access to the Opportunity. This field cannot be updated. |
| `OpportunityAccessLevel` | string | Level of access that the User or Group has to the Opportunity. One of the following values:<br>• **None** - User or Group cannot access the Opportunity.<br>• **Read** - User or Group can only view the Opportunity.<br>• **Edit** - User or Group can view or edit the Opportunity.<br>• **All** - User or Group can view, edit, delete, and share the Opportunity with other users. This value is not valid for create and update calls.<br>This field must be set to an access level that is higher than the organization's default access level for opportunities. |
| `RowCause` | string | Reason that this sharing entry exists. Read-only. One of the following values:<br>• **Owner** - The User is the owner of the Opportunity or is in a Role above the Opportunity owner in the role hierarchy.<br>• **Manual** - The User or Group has access because a User with "All" access manually shared the Opportunity with them.<br>• **Rule** - The User or Group has access via an Opportunity sharing rule.<br>• **ImplicitChild** - The User or Group has access to the Opportunity on the Account associated with this Opportunity.<br>• **Team** - The User has access to the Opportunity because she or he is on the sales team for the Opportunity. The OpportunityTeamMember object for this Opportunity sets the access level. See OpportunityTeamMember on page 89 for more information. |
| `LastModifiedById` | ID | ID of the User who last updated this OpportunityShare entry. |
| `LastModifiedDate` | dateTime | Date and time when this OpportunityShare entry was last modified. |

### Usage

The OpportunityShare object allows you to determine which users and groups can view and/or edit opportunities owned by other users. For more information, see Sharing on page 11.

Your client application can invoke the describeSObject, create, update, query, and delete calls on OpportunityShare objects. If you attempt to create an OpportunityShare that matches an existing OpportunityShare record, the create call updates any modified fields and returns the existing record.

### See Also

*Group on page 77*
*About sforce Objects*

## OpportunityTeamMember

Represents an individual User on the sales team of a particular Opportunity.

### Fields

**Table 30: OpportunityTeamMember Fields**

| Field | Data Type | Description |
|---|---|---|
| `Id` | ID | Unique ID of this record. Read-only. |
| `OpportunityId` | ID | ID of the Opportunity associated with this sales team. This field cannot be updated. |
| `UserId` | ID | ID of the User who is a member of the Opportunity's sales team. This field cannot be updated. |
| `TeamMemberRole` | string | Role that the team member has on the Opportunity. The valid values are set by the organization's administrator in the Sales Team Roles picklist. |
| `LastModifiedById` | ID | ID of the User who last updated this OpportunityTeamMember. |
| `LastModifiedDate` | dateTime | Date and time when this OpportunityTeamMember was last modified. |

### Usage

Your client application can invoke the describeSObject, create, update, query, and delete calls on OpportunityTeamMember objects. If you attempt to insert an OpportunityTeamMember that matches an existing OpportunityTeamMember record, the create call updates any modified fields and returns the existing record.

In the salesforce.com user interface, users can set up a sales team for the opportunities they own. The sales team includes other users that are working on the Opportunity with them. The OpportunityTeamMember object is available only in organizations that have enabled the team selling functionality.

### See Also

*UserTeamMember on page 98*
*About sforce Objects on page 59*

## Partner

Represents the association between two particular Accounts or between a particular Opportunity and an Account.

### Fields

**Table 31: Partner Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. |
| **`OpportunityId`** | ID | ID of the Opportunity in a partner relationship between an Account and an Opportunity.<br>Specifying this field when calling create creates an Opportunity Partner. If you specify the **`AccountFromId`** field, you cannot specify this field as well. |
| **`AccountFromId`** | ID | ID of the main Account in a partner relationship between two accounts.<br>Specifying this field when calling create creates an Account Partner. If you specify the **`OpportunityId`** field, you cannot specify this field as well. |
| **`AccountToId`** | ID | ID of the Partner Account related to either an Opportunity or an Account.<br>You must specify this field when creating an Opportunity Partner or an Account Partner. |
| **`Primary`** | boolean | Valid for Opportunity Partners only.<br>Indicates that the Account is the primary Partner for the Opportunity.<br>Only one Account can be marked as "primary" for an Opportunity. If you set the **`Primary`** flag to '1' upon insert of a new Opportunity Partner, any other existing primary partners for that Opportunity will automatically have their **`Primary`** flag set to False. |
| **`Role`** | string | Role that the Account has towards the related Opportunity or Account (e.g., "Consultant" or "Distributor"). |
| **`CreatedById`** | ID | ID of the User who created this Partner. Read-only. |
| **`CreatedDate`** | dateTime | Date and time when this Partner was created. Read-only. |
| **`LastModifiedById`** | ID | ID of the User who last updated this Partner. Read-only. |
| **`LastModifiedDate`** | dateTime | Date and time when this Partner was last modified. Read-only. |

### Usage

Your client application can invoke the describeSObject, create, query, and delete calls on Partner objects. All of the Partner fields are accessible in the describeSObject and query calls; see Fields on page 90. You must have the "View All Data" permission to access Partners via the API.

Each Account in the relationship is assigned a Role (e.g., "Consultant" or "Distributor") designating that Account's Role towards the related Account or Opportunity. A relationship between two Accounts is referred to as an Account Partner, and a relationship between an Opportunity and an Account is referred to as an Opportunity Partner.

Using the create call, you can insert the **OpportunityId** or **AccountFromId**, **AccountToId**, **Primary**, and **Role** fields. You cannot update Partners via the sforce API.

Using the create call, you can insert the **OpportunityId** or **AccountFromId**, **AccountToId**, **Primary**, and **Role** fields. When creating a Partner object, you must specify either the **OpportunityId** field or the **AccountFromId** field. Specifying the **OpportunityId** field creates an Opportunity Partner, and the **AccountFromId** field creates an Account Partner. You must always specify a value for the **AccountToId** field.

When you create an Account Partner, i.e., a relationship between two Accounts, the sforce API automatically creates a reverse partner relationship between those two Accounts. For example, if you create an Account Partner with "Acme, Inc." as the **AccountFromId** and "Acme Consulting" as the **AccountToId**, the API automatically creates a reverse partner with "Acme Consulting" as the **AccountFromId** and "Acme, Inc." as the **AccountToId**. In the reverse partner, the value of the **Role** field is set to the designated reverse Role value associated with the value of the **Role** field in the original Account Partner. In the salesforce.com user interface, system administrators can set up the valid Role values and their corresponding reverse Role values.

If you set the **Primary** flag to '1' upon insert of a new Opportunity Partner, any other existing primary partners for that Opportunity will automatically have their **Primary** flag set to False.

### See Also

*About sforce Objects on page 59*

## Pricebook

Represents a price book that contains the list of Products that your organization sells.

### Fields

Pricebook fields do not require any special handling. For a list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

### Usage

Use the Pricebook object to query information about Pricebooks that have been configured for an organization. The purpose of the Pricebook object is to allow you to obtain valid Pricebook object IDs for use when querying or modifying Products through the API. See Product on page 92 for more information. Your client application can query Pricebooks but it cannot create, update, or delete Pricebooks.

The Pricebook object is defined only for those organizations that have Products enabled as a feature. If the organization does not have the Products feature, the Pricebook object type does not appear in the describeGlobal call, and you cannot use describeSObject or query with the Pricebook object.

Products are the main constituents of Pricebooks. "Setting up a price book" via the sforce API usually means loading products into those Pricebooks. The usual way to configure Pricebooks via the API is:

1. Manually create the Pricebooks using the salesforce.com user interface.

2. Query the Pricebook object to obtain the Pricebook IDs.
3. To load products into those Pricebooks, call create or update using the sforce API.

### See Also

*About sforce Objects on page 59*

## Product

Represents a product that your organization sells. A product is member of the list of items in a Pricebook.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

#### priceBookId Field

The **PriceBookId** field indicates the Pricebook on which the Product is configured. When inserting Products, the **PricebookId** field must be set, and its value must be valid and non-blank. You cannot update the **PricebookId** field during an update call. You can query the Pricebook object to obtain valid Pricebook object IDs for your organization after they have been created using the salesforce.com user interface.

#### Active Flag

The **Active** flag indicates whether a Product is active. Although you can never delete Products, you can set a Product's **Active** flag to False. Inactive Products are hidden in many areas in the salesforce.com user interface. You can change a Product's **Active** flag as often as necessary.

#### Schedule Fields

The Product object has several fields that are only used for schedules (a.k.a, annuities). Sforce supports two types of schedules on Products - quantity schedules and revenue schedules. Schedules are available only for those organizations that have the Products and Annuities features enabled. If the organization does not have the Annuities feature, the schedule fields do not appear in the DescribeSObjectResult, and you cannot query, create, or update the fields.

**Schedule Enabled Flags**

When enabling the Annuities feature, organizations can decide whether to enable quantity schedules, revenue schedules, or both. In addition, you can use the sforce API to control quantity and revenue scheduling at the product level via the **RevenueScheduleEnabled** and **QuantityScheduleEnabled** flags. A value of True for either flag indicates that the Product and any OpportunityLineItems can have a schedule of that type. These flags can be set via a create or update call.

**Default Schedule Fields**

The remaining Product schedule fields define default schedules for the object. Sforce uses the default schedule values to create an OpportunityLineItemSchedule when an OpportunityLineItem is created for the Product.

The following table lists the default schedule fields and their valid values.

**Table 32: Default Schedule Fields on Products**

| Field | Valid Values |
|---|---|
| **RevenueScheduleType** | None, Divide, Repeat |
| **RevenueInstallmentPeriod** | None, Daily, Weekly, Monthly, Quarterly, Yearly |

**Table 32: Default Schedule Fields on Products (Continued)**

| Field | Valid Values |
|---|---|
| `NumberOfRevenueInstallments` | An integer from 1 to 100 |
| `QuantityScheduleType` | None, Divide, Repeat |
| `QuantityInstallmentPeriod` | None, Daily, Weekly, Monthly, Quarterly, Yearly |
| `NumberOfQuantityInstallments` | An integer from 1 to 100 |

When you attempt to set the schedule fields via a create or update call, the sforce API applies cross-field integrity checks. The integrity requirements are:

- If the schedule type is set to "None," the installment period and number of installments must be null.
- If the schedule type is set to any value other than "None," the installment period and number of installments must be non-null.

Inserts or updates that fail these integrity checks are rejected with an error.

The **`RevenueScheduleType`**, **`RevenueInstallmentPeriod`**, **`QuantityScheduleType`**, and **`QuantityInstallmentPeriod`** fields are restricted picklist fields and are available only if the organization has the Annuities feature enabled.

### Usage

The Product object allows you to query, create, and update Products on Pricebooks. These operations constitute the main configuration necessary for Pricebooks (see Pricebook on page 91). Products can be queried, inserted, and updated via the sforce API, but they cannot be deleted through the API or any other means. See Active Flag on page 92 for more information. Because Products can never be deleted, please exercise caution when creating them.

The Product object is defined only for those organizations that have Products enabled as a feature. If the organization does not have the Products feature, the Product object type does not appear in the describeGlobal call, and you cannot use describeSObject or query with the Product object.

### See Also

*OpportunityLineItem on page 84*
*OpportunityLineItemSchedule on page 85*
*About sforce Objects on page 59*

## Profile

Represents a profile, which defines a set of permissions to perform different operations, such as querying, adding, updating, or deleting information.

### Fields

Profile fields do not require any special handling. For a list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

### Usage

Use the Profile object to query the set of currently configured Profiles in the organization. Your client application can use Profile objects to obtain valid Profile IDs for use when querying or modifying Users through the API. Your client application can query, create, update, and delete Profiles.

### See Also

*About sforce Objects on page 59*

## RecordType

Represents a record type.

### Fields

**Table 33: RecordType Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. This field is read-only. |
| **`IsActive`** | boolean | Indicates whether the RecordType is active or not. Only active RecordTypes can be used. |
| **`Name`** | string | Name of the RecordType. |
| **`TableEnumOrIdName`** | string | Object to which this RecordType applies. Valid values include "lead," "contact," "account," and other object types that support RecordTypes. A particular RecordType can apply to only one type of object. |
| **`CreatedById`** | ID | ID of the User who created this RecordType. Read-only. |
| **`CreatedDate`** | dateTime | Date and time when this RecordType was created. Read-only. |
| **`LastModifiedById`** | ID | ID of the User who last updated this RecordType. Read-only. |
| **`LastModifiedDate`** | dateTime | Date and time when this RecordType was last modified. Read-only. |

### Usage

Use the RecordType object to offer different BusinessProcesses and subsets of picklists values to different users based on their particular Profile. Your client application can invoke the describeSObject and query calls on RecordType objects. Record types are read-only in the sforce API.

### See Also

*About sforce Objects on page 59*

## Role

Represents a role in your organization.

### Fields

Role fields do not require any special handling. For a list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

### Usage

Use the Role object to query the set of currently configured roles in your organization. Use it in your client application to obtain valid Role IDs to use when querying or modifying a User.

All Users have access to invoke query or describeSObject with the Role object. If your client application logs in with "Modify All Data" access, it can query, create, update, and delete Roles.

### See Also

*About sforce Objects on page 59*

## Scontrol

Represents an sforce control, which is custom content that is hosted by the server but executed by client applications.

### Fields

**Table 34: Scontrol Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. |
| **`Name`** | string | Name of this Scontrol. |
| **`Description`** | string | Description of this Scontrol. |
| **`HtmlWrapper`** | string | HTML page that will be delivered when the user views this Scontrol. This HTML page can be the entire content of the Scontrol, or it can reference the binary. Up to 1048576 characters. |
| **`Binary`** | binary | Binary content of this Scontrol, such as an ActiveX control or a Java archive. Up to 5MB. Can be specified when your client application calls create but not update. |
| **`CreatedById`** | ID | ID of the User who created this Scontrol. Read-only. |
| **`CreatedDate`** | dateTime | Date and time when this Scontrol was created. Read-only. |
| **`LastModifiedById`** | ID | ID of the User who last updated this Scontrol. Read-only. |
| **`LastModifiedDate`** | dateTime | Date and time when this Scontrol was last modified. Read-only. |
| **`SystemModStamp`** | dateTime | Date and time when this record was last modified. Read-only. |

### Usage

Use Scontrol objects to manage custom content on the sforce API server that is executed by client applications. All users can view Scontrol objects, but "Customize salesforce.com" permission is required to create or update Scontrol objects. Your organization must be using Enterprise Edition and be configured with sforce controls enabled.

### See Also

*About sforce Objects on page 59*

## Solution

Represents a solution, which is a detailed description of a customer issue and the resolution of that issue.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

#### SolutionNumber Field

The **`SolutionNumber`** field is assigned automatically when a Solution is inserted. It cannot be set directly, and it cannot be modified after the Solution is created.

#### IsPublished Flag

Solutions can have a special status called "published." (See the salesforce.com online help documentation for more information on publishing Solutions.) A Solution's published state does not affect how you can use a Solution, or whether you can query, update, or delete it.

#### Status Field and IsReviewed Flag

The **`Status`** field is a picklist field that also directly controls the **`IsReviewed`** flag. You cannot directly set the **`IsReviewed`** flag, but you can set it indirectly by setting the **`Status`** field. Each predefined **`Status`** field value implies an **`IsReviewed`** flag value.

### Usage

Use Solution objects to manage your organization's solution knowledge base.

### See Also

*About sforce Objects on page 59*

## Task

Represents a task.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

#### whoId and whatId Fields

The task object has **`WhoId`** and **`WhatId`** fields that function like the same fields on the Event object. See Event on page 75 for more information.

#### Status Field and Closed Flag

The **`Status`** field is a picklist that directly controls the **`Closed`** flag. You cannot directly set the **`Closed`** flag, but you can set it indirectly by setting the **`Status`** field. Each predefined **`Status`** field value implies a **`Closed`** flag value.

#### ActivityDate Field

The due date information for the task object is contained in the **`ActivityDate`** field. This field is a date field with a timestamp that is always set to midnight in the GMT/UTC time zone. The timestamp is not relevant, and you should not attempt to alter it to account for any time zone differences. For more information, see Date Field Type on page 14.

### Usage

Archived Task objects are not accessible via the sforce API. See Archived Activities on page 76.

### See Also

*About sforce Objects on page 59*

## User

Represents a user in your organization.

### Fields

For a complete list of fields in this object, see the Enterprise WSDL file for you organization and the salesforce.com online help.

#### Username Field

The **`Username`** field contains the name that a User enters to log into the sforce API or the salesforce.com user interface. The **`Username`** must be in the form of an email address. It must also be unique across all sforce instances. If you try to create or update a User with a duplicate **`Username`**, the operation is rejected and fault code 1229 “duplicate username” is returned.

Each inserted User also counts as a license in sforce. Every organization has a maximum number of licenses. If you attempt to exceed the maximum number of licenses by inserting Users, the create call is rejected and fault code 1230 “license limit exceeded” is returned.

#### Active Flag

The **`Active`** flag on the User object determines whether the User has access to log in. You can modify a User's active status using the salesforce.com user interface or via the sforce API.

#### Timezone Field

The **`Timezone`** field is a restricted picklist field. A User's time zone affects the offset used when displaying or entering times in the user interface. However, the sforce API does not use a User's time zone when querying or setting values.

The **`Timezone`** field values are named using region and key city, according to ISO standards. It can often be more convenient to manually set a User's time zone in the user interface, and then use that value for inserting or updating other Users via the API.

### Locale Field

The **`Locale`** field is a restricted picklist field. The value of the **`Locale`** field affects formatting and parsing of values, especially numeric values, in the user interface. It does not affect the operation of the sforce API.

The **`Locale`** field values are named according to the language, and country if necessary, using two-letter ISO codes. The set of names is based on the ISO standard. It can often be more convenient to manually set a User's **`Locale`** in the user interface, and then use that value for inserting or updating other Users via the sforce API.

## Usage

Use the User object to query information about users and to provision and modify users in your organization. Unlike with other objects, the records in the User table represent actual users—not data owned by users.

All Users have access to use query or describeSObject with User objects. To create or update a User object, you must log in with "Manage Users" permission.

### Disabling Users

You cannot delete Users in the salesforce.com user interface or the sforce API. To disable a User, deactivate that User in the salesforce.com user interface. Because Users can never be deleted, we recommend that you exercise caution when creating them.

### Passwords

For security reasons, you cannot query Users' passwords via the API or the salesforce.com user interface. However, the sforce API allows you to set and "reset" Users' passwords using the setPassword and resetPassword calls.

The password lockout status and the ability to reset a User's locked-out status is not available via the API. You must check and reset a User's password lockout status using the salesforce.com user interface.

## See Also

*getUserInfo on page 54*
*About sforce Objects on page 59*

# UserTeamMember

Represents a single User on the default sales team of another user.

## Fields

**Table 35: UserTeamMember Fields**

| Field | Data Type | Description |
|---|---|---|
| **`Id`** | ID | Unique ID of this record. Read-only. |
| **`OwnerId`** | ID | ID of the User who owns the default sales team. This field cannot be updated. |
| **`UserId`** | ID | ID of the User who is a member of the default sales team. This field cannot be updated. |

**Table 35: UserTeamMember Fields (Continued)**

| Field | Data Type | Description |
|---|---|---|
| `OpportunityAccessLevel` | string | Level of access that the team member has to opportunities for which the User has added his or her default sales team. One of the following values:<br>• **None** - User cannot access the Opportunity.<br>• **Read** - User can only view the Opportunity.<br>• **Edit** - User can view or edit the Opportunity.<br>• **All** - User can view, edit, delete, and share the Opportunity with other Users. This value is not valid for create and update calls.<br>This field must be set to an access level that is higher than the organization's default access level for opportunities. |
| `TeamMemberRole` | string | Role that the team member has on opportunities for which the User has added his or her default sales team. The valid values are set by the organization's administrator in the Sales Team Roles picklist. |
| `LastModifiedById` | ID | ID of the User who last updated this UserTeamMember. |
| `LastModifiedDate` | dateTime | Date and time when this UserTeamMember was last modified. |

## Usage

Your client application can invoke the describeSObject, create, update, query, and delete calls on UserTeamMember objects. If you attempt to insert a UserTeamMember that matches an existing UserTeamMember record, the create call updates any modified fields and returns the existing record.

Users can set up their default sales team to include the other Users that typically work with them on opportunities. The UserTeamMember object is available only in organizations that have enabled the team selling functionality.

## See Also

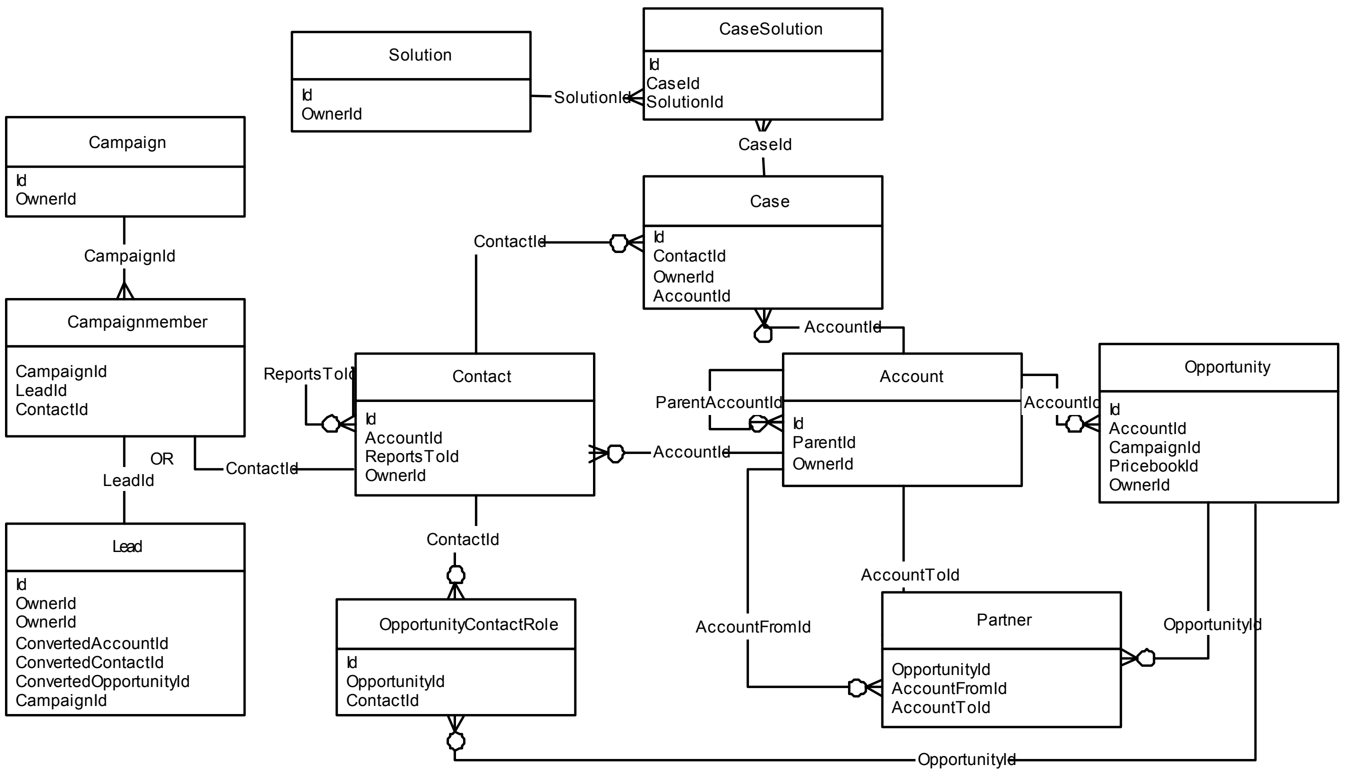
*OpportunityTeamMember on page 89*
*About sforce Objects on page 59*

# CHAPTER 6: Entity Relationship Diagrams

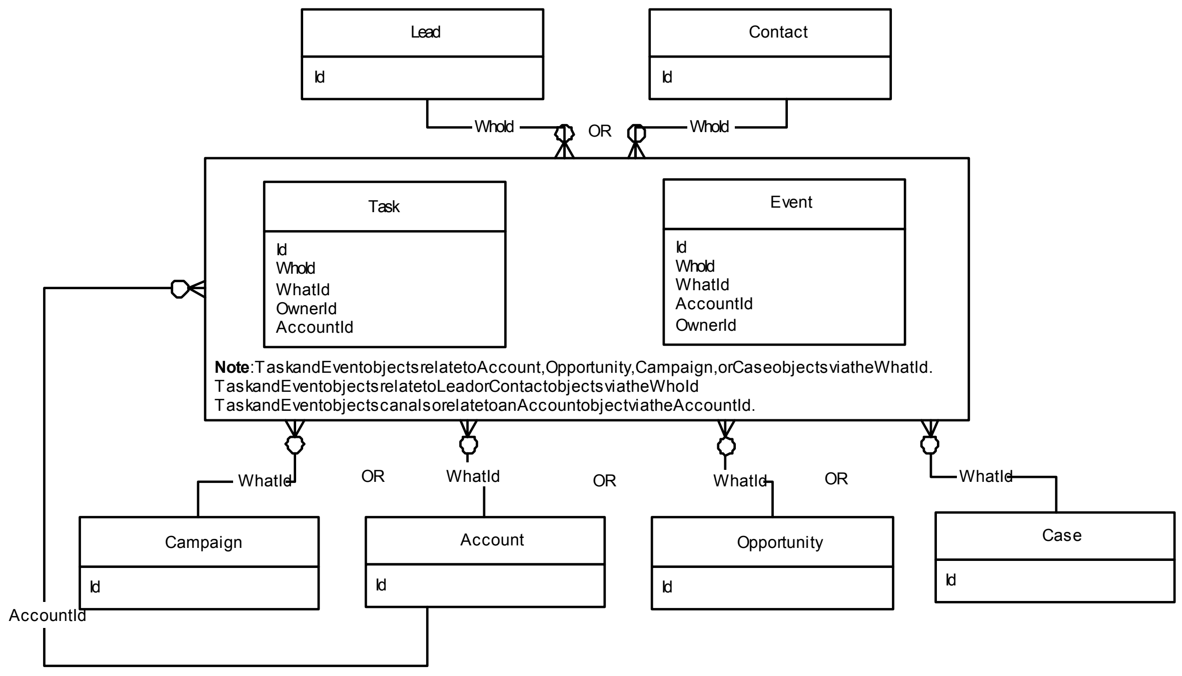
This topic describes the entity relationship diagrams (ERDs) for sforce objects. It includes the following sections:

- Major Objects
- Task and Event Objects
- Support Objects
- Document, Note, and Attachment Objects
- User, Profile, and Record Type Objects
- Product and Schedule Objects
- Sharing and Team Selling Objects

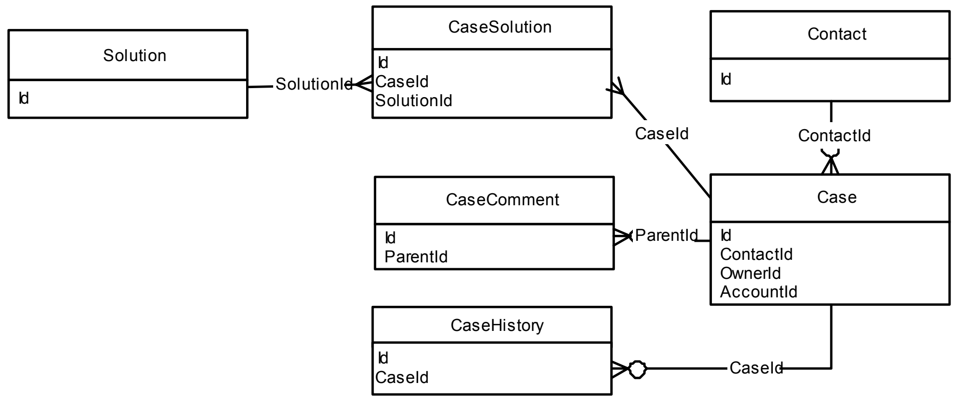
## MAJOR OBJECTS



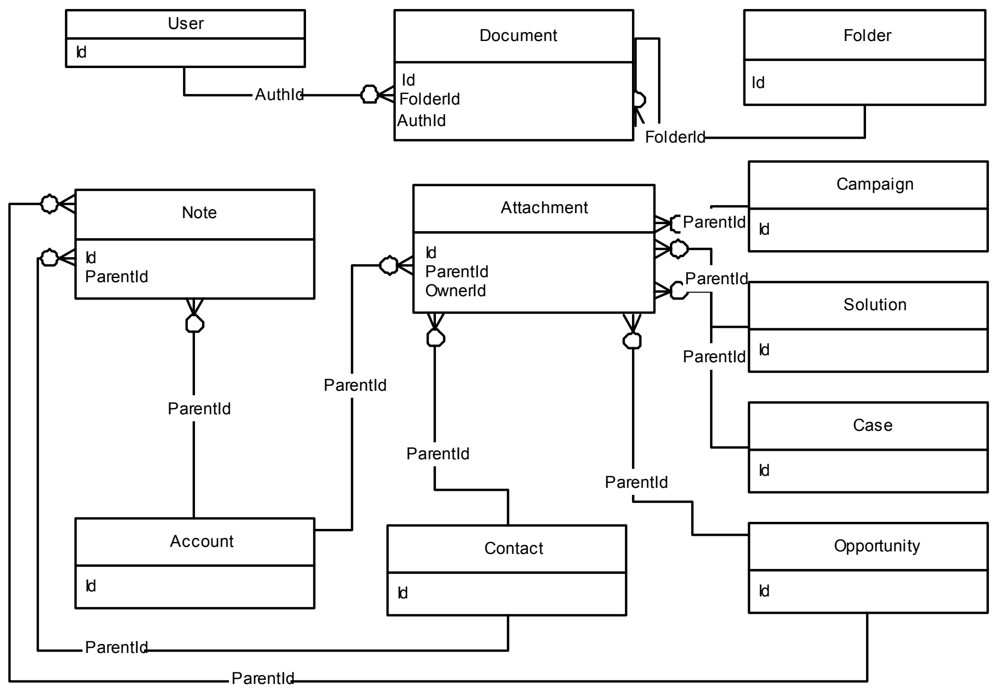
## TASK AND EVENT OBJECTS


Lead
Id
Contact
Id
WhoId
OR
WhoId
Task
Id
WhoId
WhatId
OwnerId
AccountId
Event
Id
WhoId
WhatId
AccountId
OwnerId
Note:TaskandEventobjectsrelatetoAccount,Opportunity,Campaign,orCaseobjectsviatheWhatId.
TaskandEventobjectsrelatetoLeadorContactobjectsviatheWhoId
TaskandEventobjectscanalsorelatetoanAccountobjectviatheAccountId.
WhatId
OR
WhatId
OR
WhatId
OR
WhatId
Campaign
Id
Account
Id
Opportunity
Id
Case
Id
AccountId

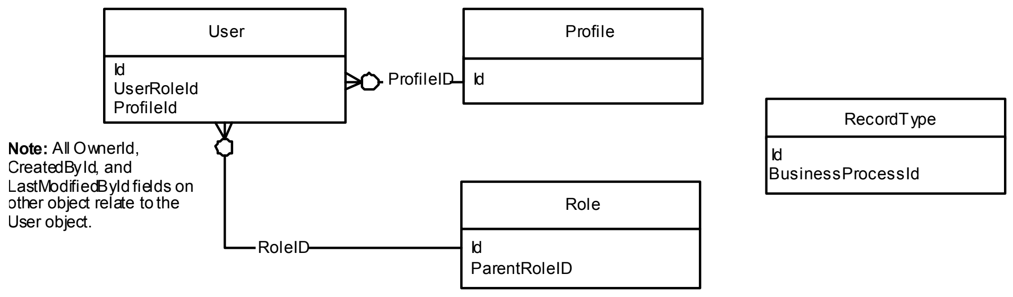
## SUPPORT OBJECTS



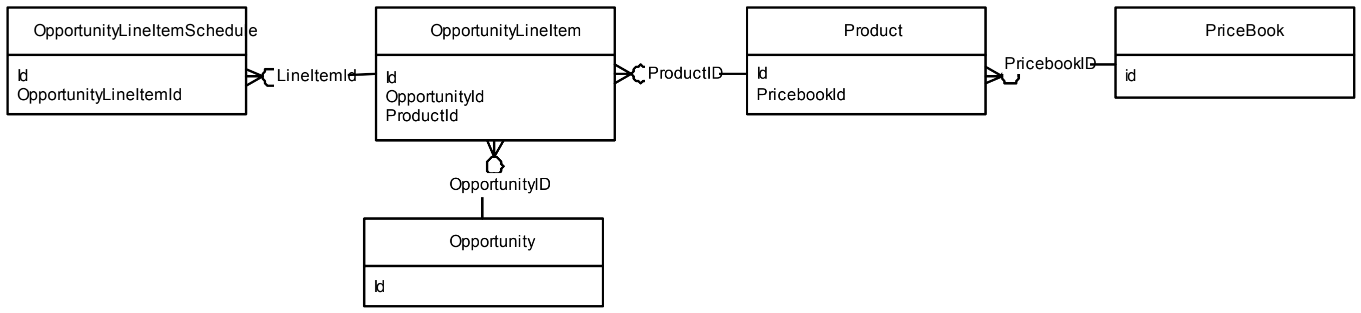
## DOCUMENT, NOTE, AND ATTACHMENT OBJECTS



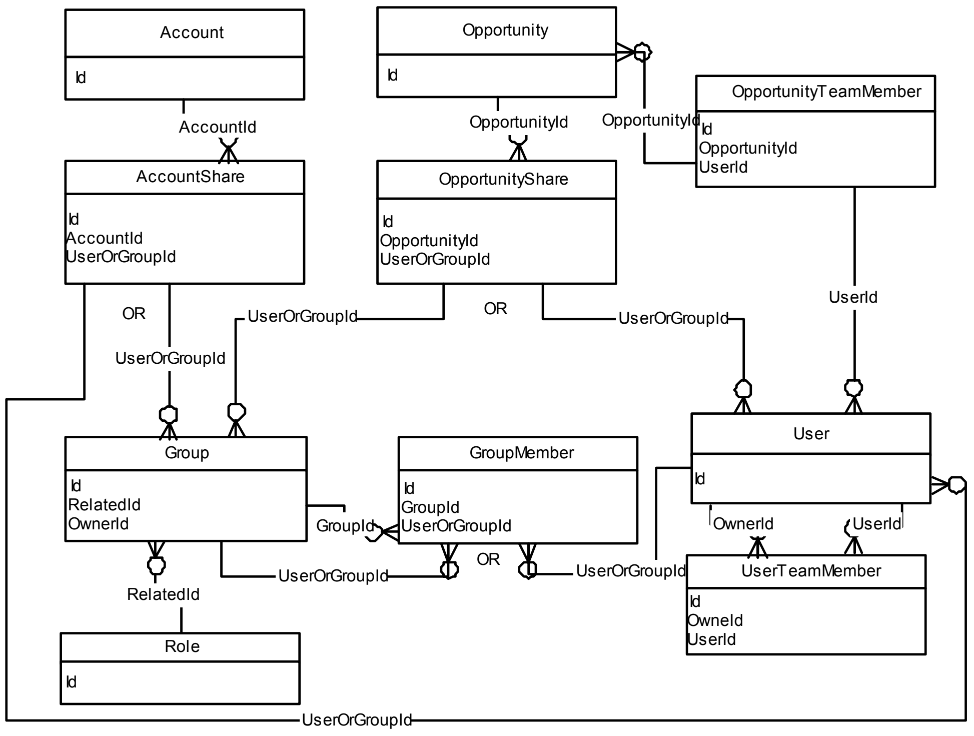
## USER, PROFILE, AND RECORD TYPE OBJECTS


User
Id
UserRoleId
ProfileId
ProfileID
Profile
Id
RecordType
Id
BusinessProcessId
Note: All OwnerId, CreatedById, and LastModifiedById fields on other object relate to the User object.
RoleID
Role
Id
ParentRoleID

## PRODUCT AND SCHEDULE OBJECTS



## SHARING AND TEAM SELLING OBJECTS



# CHAPTER 7: sforce Partner Web Services API

This topic describes the sforce Partner Web services API. It contains the following sections:

## INTRODUCING THE SFORCE PARTNER WEB SERVICES API

The sforce Web services API comes in two variations:

- **sforce Enterprise Web services API**—Used by enterprise developers to build client applications for a *single* salesforce.com organization.
- **sforce Partner Web services API**—Used for client applications that are metadata driven and dynamic in nature. It is particularly—but not exclusively—useful to salesforce.com partners who are building client applications for multiple organizations. As a loosely typed representation of the salesforce.com object model, it can be used to access data within any organization. It is more flexible, although not as easy to use, as its Enterprise counterpart.

This topic introduces the sforce Partner Web services API and describes how it differs from the sforce Enterprise Web services API. In general, the sforce Enterprise Web services API is more straightforward to use, while the sforce Partner Web services API is more flexible and dynamically adaptable to different organizations, allowing you to write a single application that can be used for multiple users and multiple organizations.

### WSDL Files

To use the sforce Partner Web services API, you need a partner.wsdl file, which you can either:

- obtain from your organization's salesforce.com administrator, or
- generate in the **Setup** | **Sforce Application Server** area of the salesforce.com user interface according to the instructions in Step 2: Generate or Obtain the sforce Web services WSDL For Your Organization on page 3.

While the enterprise.wsdl file (used with the sforce Enterprise Web services API) needs to be generated if custom fields or custom objects are added to an organization's salesforce.com information, the partner.wsdl file remains the same regardless of underlying changes in the organization's salesforce.com data.

### API Calls in the sforce Partner Web Services API

The partner.wsdl file defines exactly the same API calls found in the enterprise.wsdl file. A client application using the Partner Web services API will likely use the following API calls to determine an organization's metadata:

**Table 36: Object Metadata Calls in the sforce API**

| Task / Call | Description |
|---|---|
| describeGlobal | Retrieves a list of available object types for your organization's data. |

**Table 36: Object Metadata Calls in the sforce API (Continued)**

| Task / Call | Description |
|---|---|
| describeSObject | Describes metadata (field list and object properties) for the specified object type. |

To explore an organization's metadata, a client application can:

1. Call describeGlobal to obtain a list of available object types.
2. In the returned DescribeGlobalResult, retrieve an array of sObjects (`types` field).
3. Iterate through each sObject in the array, calling describeSObject to retrieve a list of fields and other properties for the sObject in the returned DescribeSObjectResult.

## OBJECTS, FIELDS, AND FIELD DATA IN THE SFORCE PARTNER WEB SERVICES API

While the enterprise.wsdl file defines all of the specific object types (such as Account, Contact, and other objects described in sforce Objects on page 59) in a salesforce.com organization, the partner.wsdl file defines a single, generic object (sObject) that represents all of the object types. For a particular object, its type is defined in the `name` field in the returned DescribeSObjectResult.

In the sforce Partner Web services API, your client application code handles fields as arrays of name-value pairs that represent the field metadata. When referring to the names of individual fields, use the value in its `name` field of the Field type in the DescribeSObjectResult.

Languages vary in the way they handle name-value pairs and map typed values to the primitive XML data types defined in SOAP messages. In the sforce Enterprise Web services API, the mapping is handled implicitly. In the sforce Partner Web services API, however, you need to be more attentive to values and data types when building client applications. When specifying the value of a particular field, be sure to use a value that is valid for the field (range, format, and data type). Make sure that you understand the mapping between data types in your programming language with XML primitive data types (one of the values in the `SOAPType` field of the Field type in the DescribeSObjectResult).

## QUERIES IN THE SFORCE PARTNER WEB SERVICES API

When using the query call in the sforce Partner Web services API, consider the following guidelines:

- The `queryString` parameter is case-insensitive. The sforce API server will accept field names in the *`fieldList`* using any combination of uppercase and lowercase letters. However, in the QueryResult, the case of field names (both predefined and custom fields) will match exactly the value in the `name` field of the Field type in the DescribeSObjectResult. It is recommended that you use the proper case when specifying fields in the *`fieldList`*.
- The ordering of fields in the QueryResult is determined by the field order in the WSDL file, not by the field order in the *`fieldList`*.
- The *`fieldList`* cannot contain duplicate field names. For example:
  - Invalid: `"select firstname, lastname, firstname from User"`
  - Valid: `"select firstname, lastname from User"`
- The QueryResult always contains all of the fields specified in the *`fieldList`*, even if some of the fields contain no data (null). Although SOAP allows you to omit fields that contain no values in the result set, the sforce Web services API always returns an array containing all fields.

## NAMESPACES IN THE SFORCE PARTNER WEB SERVICES API

In XML, every tag has a defined namespace. In the enterprise.wsdl, namespaces are handled implicitly. When using API calls in the sforce Partner Web services API, however, you need to explicitly specify the correct namespaces for sforce API calls, objects, and fields, and faults. This rule applies to predefined and custom objects and fields.

**Table 37: Namespaces for the sforce Partner Web Services API**

| For | Namespace |
|---|---|
| API Calls | `partner.soap.sforce.com` |
| sObjects | `sobject.partner.soap.sforce.com` |
| Fields | `sobject.partner.soap.sforce.com` |
| Faults | `fault.partner.soap.sforce.com` |

# CHAPTER 8: Sample SOAP Messages

This topic provides sample input and output SOAP messages for the sforce API calls. It includes the following topics:

- Sample SOAP Messages—create
- Sample SOAP Messages—delete
- Sample SOAP Messages—describeGlobal
- Sample SOAP Messages—describeSObject
- Sample SOAP Messages—getServerTimestamp
- Sample SOAP Messages—getUserInfo
- Sample SOAP Messages—login
- Sample SOAP Messages—query
- Sample SOAP Messages—queryMore
- Sample SOAP Messages—resetPassword
- Sample SOAP Messages—retrieve
- Sample SOAP Messages—setPassword
- Sample SOAP Messages—update

For detailed information about the sforce API calls, see the following topics:

- sforce API Calls on page 24
- sforce Utility API Calls on page 53

## Sample SOAP Messages—create

This topic provides the following sample SOAP messages for the create call:

- Sample Request Message
- Sample Response Message

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
Content-Length: 1417

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
```

```
            <ns2:sessionId xmlns:ns2="urn:enterprise.soap.sforce.com">0XfofujfT7
</ns2:sessionId>
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <create xmlns="urn:enterprise.soap.sforce.com">
            <sObjects xsi:type="ns3:Account"
xmlns:ns3="urn:sobject.enterprise.soap.sforce.com">
               <ns3:Name>Golden Straw</ns3:Name>
               <ns3:BillingStreet>4322 Haystack Boulevard</ns3:BillingStreet>
               <ns3:BillingCity>Wichita</ns3:BillingCity>
               <ns3:BillingState>KA</ns3:BillingState>
               <ns3:BillingPostalCode>87901</ns3:BillingPostalCode>
               <ns3:BillingCountry>US</ns3:BillingCountry>
               <ns3:Phone>666.666.6666</ns3:Phone>
               <ns3:Fax>555.555.5555</ns3:Fax>
               <ns3:AccountNumber>0000000</ns3:AccountNumber>
               <ns3:Website>www.oz.com</ns3:Website>
               <ns3:Industry>Farming</ns3:Industry>
               <ns3:NumberOfEmployees>40</ns3:NumberOfEmployees>
               <ns3:Ownership>Privately Held</ns3:Ownership>
               <ns3:Description>World class hay makers.</ns3:Description>
            </sObjects>
         </create>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample Response Message

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 20:21:14 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <createResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <errors xsi:nil="true"/>
               <id>001x00000000KQYAA2</id>
               <success>true</success>
            </result>
         </createResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—delete

This topic provides the following sample SOAP messages for the delete call:

- Sample Request Message
- Sample Response Message—No Errors
- Sample Response Message—With Errors

## Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
Content-Length: 702

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId xmlns:ns2="urn:enterprise.soap.sforce.com">b2pHCw.u8DD
</ns2:sessionId>
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <delete xmlns="urn:enterprise.soap.sforce.com">
            <ids>001x00000000KQeAAM</ids>
            <ids>001x00000000KQfAAM</ids>
         </delete>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample Response Message—No Errors

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 20:31:04 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <deleteResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <errors xsi:nil="true"/>
               <id>001x00000000KQeAAM</id>
               <success>true</success>
            </result>
            <result>
               <errors xsi:nil="true"/>
               <id>001x00000000KQfAAM</id>
               <success>true</success>
            </result>
         </deleteResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

### Sample Response Message—With Errors

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 20:33:09 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <deleteResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <errors xsi:nil="true"/>
               <id>001x00000000KQeAAM</id>
               <success>true</success>
            </result>
            <result>
               <errors>
                  <code>1202</code>
                  <fields xsi:nil="true"/>
                  <message>invalid cross reference id</message>
               </errors>
               <id xsi:nil="true"/>
               <success>false</success>
            </result>
         </deleteResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—describeGlobal

This topic provides the following sample SOAP messages for the describeGlobal call:

- Sample Request Message
- Sample Response Message

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
Content-Length: 688

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId
xmlns:ns2="urn:enterprise.soap.sforce.com">VOzM_Ku3Jm07t0_jXlNKOSU</
ns2:sessionId>
```

```
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <describeGlobal xmlns="urn:enterprise.soap.sforce.com">
            <describeGlobal xsi:nil="true"/>
         </describeGlobal>
      </soapenv:Body>
   </soapenv:Envelope>
```

### Sample Response Message

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 20:48:23 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <describeGlobalResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <encoding>UTF-8</encoding>
               <maxBatchSize>500</maxBatchSize>
               <types>Account</types>
               <types>AccountShare</types>
               <types>AccountShareDefault</types>
               <types>AccountTeamMember</types>
               <types>User</types>
    …
    …
    …
               <types>UserRole</types>
               <types>UserTeamMember</types>
               <types>WebIntegrationLink</types>
            </result>
         </describeGlobalResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—describeSObject

This topic provides the following sample SOAP messages for the describeSObject call:

- Sample Request Message
- Sample Response Message—No Errors
- Sample Response Message—With Errors

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
```

```
SOAPAction: ""
Content-Length: 692


<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId
xmlns:ns2="urn:enterprise.soap.sforce.com">V.rt9WQxuzgj_.rjbjXlNKOSU</
ns2:sessionId>
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <describeSObject xmlns="urn:enterprise.soap.sforce.com">
            <sObjectType>Account</sObjectType>
         </describeSObject>
      </soapenv:Body>
   </soapenv:Envelope
```

## Sample Response Message—No Errors

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 20:52:28 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <describeSObjectResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <activateable>false</activateable>
               <createable>true</createable>
               <custom>false</custom>
               <deletable>true</deletable>
               <fields>
                  <byteLength>18</byteLength>
                  <createable>false</createable>
                  <custom>false</custom>
                  <digits xsi:nil="true"/>
                  <filterable>true</filterable>
                  <label>Account ID</label>
                  <length>18</length>
                  <name>Id</name>
                  <nillable>false</nillable>
                  <picklistValues xsi:nil="true"/>
                  <precision xsi:nil="true"/>
                  <referenceTo xsi:nil="true"/>
                  <required>true</required>
                  <restrictedPicklist>false</restrictedPicklist>
                  <scale xsi:nil="true"/>
                  <selectable>true</selectable>
                  <soapType>tns:ID</soapType>
                  <type>id</type>
                  <updateable>false</updateable>
```

```
                                        </fields>
                                        <fields>
                                           <byteLength>120</byteLength>
                                           <createable>true</createable>
                                           <custom>false</custom>
                                           <digits xsi:nil="true"/>
                                           <filterable>true</filterable>
                                           <label>Account Type</label>
                                           <length>40</length>
                                           <name>Type</name>
                                           <nillable>false</nillable>
                                           <picklistValues>
                                              <active xsi:nil="true"/>
                                              <conversionRate xsi:nil="true"/>
                                              <corporate xsi:nil="true"/>
                                              <defaultValue xsi:nil="true"/>
                                              <label xsi:nil="true"/>
                                              <scale xsi:nil="true"/>
                                              <value>Analyst</value>
                                           </picklistValues>
                                           <picklistValues>
                                              <active xsi:nil="true"/>
                                              <conversionRate xsi:nil="true"/>
                                              <corporate xsi:nil="true"/>
                                              <defaultValue xsi:nil="true"/>
                                              <label xsi:nil="true"/>
                                              <scale xsi:nil="true"/>
                                              <value>Competitor</value>
                                           </picklistValues>
                         …
                         …
                         …
                                           <precision xsi:nil="true"/>
                                           <referenceTo xsi:nil="true"/>
                                           <required>false</required>
                                           <restrictedPicklist>false</restrictedPicklist>
                                           <scale xsi:nil="true"/>
                                           <selectable>true</selectable>
                                           <soapType>xsd:string</soapType>
                                           <type>picklist</type>
                                           <updateable>true</updateable>
                                        </fields>
                                        <fields>
                                           <byteLength>18</byteLength>
                                           <createable>true</createable>
                                           <custom>false</custom>
                                           <digits xsi:nil="true"/>
                                           <filterable>true</filterable>
                                           <label>Parent Account ID</label>
                                           <length>18</length>
                                           <name>ParentId</name>
                                           <nillable>false</nillable>
                                           <picklistValues xsi:nil="true"/>
                                           <precision xsi:nil="true"/>
                                           <referenceTo>Account</referenceTo>
                                           <required>false</required>
                                           <restrictedPicklist>false</restrictedPicklist>
                                           <scale xsi:nil="true"/>
                                           <selectable>true</selectable>
                                           <soapType>tns:ID</soapType>
```

```
                    <type>reference</type>
                    <updateable>true</updateable>
                 </fields>
…
…
…
                 <name>Account</name>
                 <queryable>true</queryable>
                 <replicateable>true</replicateable>
                 <retrieveable>true</retrieveable>
                 <searchable>true</searchable>
                 <undeletable>false</undeletable>
                 <updateable>true</updateable>
              </result>
           </describeSObjectResponse>
        </soapenv:Body>
   </soapenv:Envelope>
```

### Sample Response Message—With Errors

```
HTTP/1.0 500 Internal Server Error
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 20:56:57 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <soapenv:Fault>
            <faultcode>soapenv:Server</faultcode>
            <faultstring>No such entity: accounts</faultstring>
            <detail>
               <sf:fault xsi:type="sf:InvalidSObjectFault"
xmlns:sf="urn:fault.enterprise.soap.sforce.com">
                  <sf:code>1111</sf:code>
                  <sf:message>No such entity: accounts</sf:message>
               </sf:fault>
            </detail>
         </soapenv:Fault>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—getServerTimestamp

This topic provides the following sample SOAP messages for the getServerTimestamp call:

- Sample Request Message
- Sample Response Message

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
```

```
Pragma: no-cache
SOAPAction: ""
Content-Length: 700

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId
xmlns:ns2="urn:enterprise.soap.sforce.com">dmzgf2aFxVyrw8lt0_jXlNKOSU</
ns2:sessionId>
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <getServerTimestamp xmlns="urn:enterprise.soap.sforce.com">
            <getServerTimestamp xsi:nil="true"/>
         </getServerTimestamp>
      </soapenv:Body>
   </soapenv:Envelope>
```

### Sample Response Message

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 20:57:46 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <getServerTimestampResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <timestamp>2003-11-01T20:57:46.657Z</timestamp>
            </result>
         </getServerTimestampResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—getUserInfo

This topic provides the following sample SOAP messages for the getUserInfo call:

- Sample Request Message
- Sample Response Message

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
```

```
Content-Length: 717

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId
xmlns:ns2="urn:enterprise.soap.sforce.com">_ZnWl8zuUbtuhBXlNKOSU</ns2:sessionId>
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <getUserInfo xmlns="urn:enterprise.soap.sforce.com">
            <getUserInfo xsi:type="xsd:string">dcarroll1@usee.com</getUserInfo>
         </getUserInfo>
      </soapenv:Body>
   </soapenv:Envelope>
```

### Sample Response Message

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sun, 02 Nov 2003 19:26:13 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <getUserInfoResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <currencySymbol>$</currencySymbol>
               <organizationMultiCurrency>false</organizationMultiCurrency>
               <organizationName>Best Products, Inc.</organizationName>
               <userDefaultCurrencyIsoCode xsi:nil="true"/>
               <userEmail>dcroll@sce.com</userEmail>
               <userFullName>Dang Cal</userFullName>
               <userId>005x003440001ZPEAA2</userId>
               <userLanguage>en_US</userLanguage>
               <userLocale>en_US</userLocale>
               <userTimeZone>America/Los_Angeles</userTimeZone>
            </result>
         </getUserInfoResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—login

This topic provides the following sample SOAP messages for the login call:

- Sample Request Message
- Sample Response Message

### Sample Request Message

```
<SOAP-ENV:Envelope xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://
www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<SOAP-ENV:Body>
<ns:login xmlns:ns="urn:soap.sforce.com">
<ns:username>username@some.com</ns:username>
<ns:password>123456</ns:password>
</ns:login>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

### Sample Response Message

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
<soapenv:Body>
<loginResponse xmlns="urn:soap.sforce.com">
<result>
<serverUrl xsi:nil="true"></serverUrl>
<sessionId>POzjLnJ9bpYuylz.oUh2S6lNKOSU</sessionId>
<userId>005x00000001ZjeAA2</userId>
</result>
</loginResponse>
</soapenv:Body>
</soapenv:Envelope>
```

## Sample SOAP Messages—query

This topic provides the following sample SOAP messages for the query call:

- Sample Request Message
- Sample Response Message—No Errors
- Sample Response Message—With Errors

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
Content-Length: 913

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId
xmlns:ns2="urn:enterprise.soap.sforce.com">dhiRHxhelLChd0_jXlNKOSU</
ns2:sessionId>
```

```
         </ns1:SessionHeader>
         <ns3:QueryOptions soapenv:mustUnderstand="0" xmlns:ns3="SoapService">
            <ns4:batchSize xmlns:ns4="urn:enterprise.soap.sforce.com">3</
ns4:batchSize>
         </ns3:QueryOptions>
      </soapenv:Header>
      <soapenv:Body>
         <query xmlns="urn:enterprise.soap.sforce.com">
            <queryString>select id, Website, Name from Account where Name =
&apos;Golden Straw&apos;</queryString>
         </query>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample Response Message—No Errors

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 21:08:21 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <queryResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <done>false</done>
               <queryLocator>01gx000000000JXAAY-3</queryLocator>
               <records xsi:type="sf:Account"
xmlns:sf="urn:sobject.enterprise.soap.sforce.com">
                  <sf:Website>www.oz.com</sf:Website>
                  <sf:Name>Golden Straw</sf:Name>
                  <sf:id>001x00000000GVRAA2</sf:id>
               </records>
               <records xsi:type="sf:Account"
xmlns:sf="urn:sobject.enterprise.soap.sforce.com">
                  <sf:Website>www.oz.com</sf:Website>
                  <sf:Name>Golden Straw</sf:Name>
                  <sf:id>001x00000000GWsAAM</sf:id>
               </records>
               <records xsi:type="sf:Account"
xmlns:sf="urn:sobject.enterprise.soap.sforce.com">
                  <sf:Website>www.oz.com</sf:Website>
                  <sf:Name>Golden Straw</sf:Name>
                  <sf:id>001x00000000JZpAAM</sf:id>
               </records>
               <size>34</size>
            </result>
         </queryResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample Response Message—With Errors

```
HTTP/1.0 500 Internal Server Error
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
```

```
Date: Sat, 01 Nov 2003 21:10:25 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" 
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <soapenv:Fault>
            <faultcode>soapenv:Server</faultcode>
            <faultstring>No such standard developer name: Finger in Account</
faultstring>
            <detail>
               <sf:fault xsi:type="sf:InvalidFieldFault" 
xmlns:sf="urn:fault.enterprise.soap.sforce.com">
                  <sf:code>1218</sf:code>
                 <sf:message>No such standard developer name: Finger in Account</
sf:message>
               </sf:fault>
            </detail>
         </soapenv:Fault>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—queryMore

This topic provides the following sample SOAP messages for the queryMore call:

- Sample Request Message
- Sample Response Message—No Errors

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
Content-Length: 868

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" 
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
        <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId 
xmlns:ns2="urn:enterprise.soap.sforce.com">cQWiUyJGMN_0_jXlNKOSU</ns2:sessionId>
         </ns1:SessionHeader>
         <ns3:QueryOptions soapenv:mustUnderstand="0" xmlns:ns3="SoapService">
            <ns4:batchSize xmlns:ns4="urn:enterprise.soap.sforce.com">3</
ns4:batchSize>
         </ns3:QueryOptions>
      </soapenv:Header>
      <soapenv:Body>
         <queryMore xmlns="urn:enterprise.soap.sforce.com">
            <queryLocator>01gx000000000JgAAI-3</queryLocator>
         </queryMore>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample Response Message—No Errors

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 21:37:18 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <queryMoreResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <done>false</done>
               <queryLocator>01gx000000000JgAAI-6</queryLocator>
               <records xsi:type="sf:Contact"
xmlns:sf="urn:sobject.enterprise.soap.sforce.com">
                  <sf:FirstName>Joe</sf:FirstName>
                  <sf:LastName>Blow</sf:LastName>
               </records>
               <records xsi:type="sf:Contact"
xmlns:sf="urn:sobject.enterprise.soap.sforce.com">
                  <sf:FirstName>Joe</sf:FirstName>
                  <sf:LastName>Blow</sf:LastName>
               </records>
               <records xsi:type="sf:Contact"
xmlns:sf="urn:sobject.enterprise.soap.sforce.com">
                  <sf:FirstName>Joe</sf:FirstName>
                  <sf:LastName>Blow</sf:LastName>
               </records>
               <size>27</size>
            </result>
         </queryMoreResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample Response Message—With Errors

```
HTTP/1.0 500 Internal Server Error
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 21:15:56 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <soapenv:Fault>
            <faultcode>soapenv:Server</faultcode>
            <faultstring>invalid user id</faultstring>
            <detail>
               <sf:fault xsi:type="sf:InvalidIdFault"
xmlns:sf="urn:fault.enterprise.soap.sforce.com">
```

```
                  <sf:code>1201</sf:code>
                  <sf:message>invalid user id</sf:message>
               </sf:fault>
            </detail>
         </soapenv:Fault>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—resetPassword

This topic provides the following sample SOAP messages for the resetPassword call:

- Sample Request Message
- Sample Response Message

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
Content-Length: 686

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId
xmlns:ns2="urn:enterprise.soap.sforce.com">uHYv33ze9a90_jXlNKOSU</ns2:sessionId>
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <resetPassword xmlns="urn:enterprise.soap.sforce.com">
            <UserId>005x00000001ZPH</UserId>
         </resetPassword>
      </soapenv:Body>
   </soapenv:Envelope>
```

### Sample Response Message

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 21:14:22 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <resetPasswordResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <password>aHaJbh</password>
            </result>
         </resetPasswordResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—retrieve

This topic provides the following sample SOAP message for the retrieve call:

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
Content-Length: 803

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId
xmlns:ns2="urn:enterprise.soap.sforce.com">3akXj4aVzdR0Fpt0_jXlNKOSU</
ns2:sessionId>
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <retrieve xmlns="urn:enterprise.soap.sforce.com">
            <fieldList>Id, AccountNumber, Name, Website</fieldList>
            <sObjectType>Account</sObjectType>
            <ids>001x00000000JedAAE</ids>
            <ids>001x00000000JeeAAE</ids>
         </retrieve>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—setPassword

This topic provides the following sample SOAP messages for the setPassword call:

- Sample Request Message
- Sample Response Message—No Errors
- Sample Response Message—With Errors

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
```

```
Content-Length: 716

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId
xmlns:ns2="urn:enterprise.soap.sforce.com">L80wGrkd_jXlNKOSU</ns2:sessionId>
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <setPassword xmlns="urn:enterprise.soap.sforce.com">
            <UserId>005x00000001ZPH</UserId>
            <Password>bigsecret</Password>
         </setPassword>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample Response Message—No Errors

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 21:19:03 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <setPasswordResponse xmlns="urn:enterprise.soap.sforce.com">
            <result/>
         </setPasswordResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample Response Message—With Errors

```
HTTP/1.0 500 Internal Server Error
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 21:20:28 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <soapenv:Fault>
            <faultcode>soapenv:Server</faultcode>
            <faultstring>invalid user id</faultstring>
            <detail>
               <sf:fault xsi:type="sf:InvalidIdFault"
xmlns:sf="urn:fault.enterprise.soap.sforce.com">
                  <sf:code>1201</sf:code>
                  <sf:message>invalid user id</sf:message>
               </sf:fault>
            </detail>
         </soapenv:Fault>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample SOAP Messages—update

This topic provides the following sample SOAP messages for the update call:

- Sample Request Message
- Sample Response Message—No Errors
- Sample Response Message—With Errors

### Sample Request Message

```
POST /services/Soap/c/2.0 HTTP/1.0
Content-Type: text/xml; charset=utf-8
Accept: application/soap+xml, application/dime, multipart/related, text/*
User-Agent: Axis/1.1
Host: aspen.salesforce.com
Cache-Control: no-cache
Pragma: no-cache
SOAPAction: ""
Content-Length: 1030

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" 
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Header>
         <ns1:SessionHeader soapenv:mustUnderstand="0" xmlns:ns1="SoapService">
            <ns2:sessionId 
xmlns:ns2="urn:enterprise.soap.sforce.com">MkhiFia_uMCmh90_jXlNKOSU</
ns2:sessionId>
         </ns1:SessionHeader>
      </soapenv:Header>
      <soapenv:Body>
         <update xmlns="urn:enterprise.soap.sforce.com">
            <sObjects xsi:type="ns3:Account" 
xmlns:ns3="urn:sobject.enterprise.soap.sforce.com">
               <ns3:Id>001x00000000KQpAAM</ns3:Id>
               <ns3:Name>New Account Name from Update Sample</ns3:Name>
            </sObjects>
            <sObjects xsi:type="ns4:Account" 
xmlns:ns4="urn:sobject.enterprise.soap.sforce.com">
               <ns4:Id>001x00000000KQqAAM</ns4:Id>
               <ns4:Website>www.website.com</ns4:Website>
            </sObjects>
         </update>
      </soapenv:Body>
   </soapenv:Envelope>
```

### Sample Response Message—No Errors

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 20:42:55 GMT
```

```
<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <updateResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <errors xsi:nil="true"/>
               <id>001x00000000KQpAAM</id>
               <success>true</success>
            </result>
            <result>
               <errors xsi:nil="true"/>
               <id>001x00000000KQqAAM</id>
               <success>true</success>
            </result>
         </updateResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

## Sample Response Message—With Errors

```
HTTP/1.0 200 OK
Server: Resin/2.1.9
Content-Type: text/xml; charset=utf-8
Date: Sat, 01 Nov 2003 20:46:53 GMT

<?xml version="1.0" encoding="UTF-8"?>
   <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
      <soapenv:Body>
         <updateResponse xmlns="urn:enterprise.soap.sforce.com">
            <result>
               <errors xsi:nil="true"/>
               <id>001x00000000KQrAAM</id>
               <success>true</success>
            </result>
            <result>
               <errors>
                  <code>1253</code>
                  <fields xsi:nil="true"/>
                  <message>malformed id 654654644664645452</message>
               </errors>
               <id xsi:nil="true"/>
               <success>false</success>
            </result>
         </updateResponse>
      </soapenv:Body>
   </soapenv:Envelope>
```

# Index