



---

# Place Order REST API Developer Guide

Version 39.0, Spring '17





# CONTENTS

Place Order REST API Developer Guide .....	1
Requirements and Limitations .....	1
Understanding Authentication .....	2
Defining Connected Apps .....	3
Understanding OAuth Endpoints .....	3
Understanding the Web Server OAuth Authentication Flow .....	4
Understanding the User-Agent OAuth Authentication Flow .....	9
Understanding the Username-Password OAuth Authentication Flow .....	12
Understanding the OAuth Refresh Token Process .....	15
Finding Additional Resources .....	16
Understanding Place Order REST API Resources .....	17
Add a Contract and Orders to an Existing Account .....	17
Add an Order to an Existing Account .....	19
Add Orders to an Existing Contract .....	20
Add Order Products to an Existing Order .....	23
Get Details About a Contract .....	24
Get Details About an Order .....	26
Filter Details About a Contract .....	27
Filter Details About an Order .....	28
Place Order REST API Resource Reference .....	29
Create Contract-based Orders .....	29
Contract-based Orders .....	32
Create Order .....	35
Order .....	37
Index .....	41



# PLACE ORDER REST API DEVELOPER GUIDE

Access your organization's order and contract data programmatically with the Salesforce Place Order REST API.

The Salesforce Place Order REST API is a composite API that gives programmatic access to contract, order, and order product data, as well as child custom object data of contracts and orders in Salesforce. With this composite API, you can create contract, order, order product, and custom object records in a single call. Any organization that has orders and API enabled can use the Place Order REST API.

Use the Place Order REST API, a REST-based composite application programming interface, to:

- Add orders to a new or existing contract, and add order products to those orders.
- Add order products to a new or existing order.
- Add custom objects to a new or existing contract or order.
- Retrieve order records under a given contract, plus those orders' custom objects and order products.
- Retrieve order product records under a given order, plus custom object records under the order and its order products.
- Retrieve a filtered list of orders under a given contract or order products under a given order.

## IN THIS SECTION:

[Requirements and Limitations](#)

[Understanding Authentication](#)

Salesforce uses the OAuth protocol to allow users of applications to securely access data without having to reveal username and password credentials.

[Understanding Place Order REST API Resources](#)

Integrate your order creation system with Salesforce by using the Place Order REST API.

[Place Order REST API Resource Reference](#)

Each Place Order REST API resource is a URI used with an HTTP method (such as GET).

## Requirements and Limitations

---

To access the Place Order REST API, you must establish a secure OAuth session ID.

Consider these limitations and general limits when using the Place Order REST API.

### Limits and Limitations

- 2000 records per request or the API maximum limit for your organization—whichever is lower.
- Responses and requests are in JSON.

When using Place Order REST API resources that require a request or response body, use Content-Type: application/json.

- Each call can only affect one top-level entity.

For orders under a contract, you need one call for each new or existing contract you're adding orders, order products, or custom objects to. For orders not under a contract, you need one call for each new or existing order you're adding order products or custom objects to.

- In each resource, you can create custom objects at a depth of one level below the top-level entity.

- `/services/data/latest API version/commerce/sale` supports custom object records on contracts and orders.
- `/services/data/latest API version/commerce/sale/order` supports custom object records on orders and order products.
- Custom objects are not supported as children of other custom objects.
- To filter GET results, query parameters must be a fully qualified field name. The parent entity must be lower-cased (such as `order`), and the field must match the defined relationship name (such as `orders.StatusCode`).  
For example, to get a list of all orders with a draft status under a given contract, you must use `contract.orders.StatusCode='Draft'`.
- When you create a new order, the `Status Code` must be Draft and the `Status` must be any value that corresponds to a `Status Code` of Draft.
- You can't update existing records.

## Understanding Authentication

---

Salesforce uses the OAuth protocol to allow users of applications to securely access data without having to reveal username and password credentials.

Before making REST API calls, you must authenticate the application user using [OAuth 2.0](#). To do so, you'll need to:

- [Set up your application as a connected app](#) in the Salesforce organization.
- Determine the correct Salesforce [OAuth endpoint](#) for your connected app to use.
- Authenticate the connected app user via one of several different OAuth 2.0 authentication flows. An OAuth authentication flow defines a series of steps used to coordinate the authentication process between your application and Salesforce. Supported OAuth flows include:
  - [Web server flow](#), where the server can securely protect the consumer secret.
  - [User-agent flow](#), used by applications that cannot securely store the consumer secret.
  - [Username-password flow](#), where the application has direct access to user credentials.

After successfully authenticating the connected app user with Salesforce, you'll receive an access token which can be used to make authenticated REST API calls.

### IN THIS SECTION:

#### [Defining Connected Apps](#)

To authenticate using OAuth, you must create a connected app that defines your application's OAuth settings for the Salesforce organization.

#### [Understanding OAuth Endpoints](#)

OAuth endpoints are the URLs you use to make OAuth authentication requests to Salesforce.

#### [Understanding the Web Server OAuth Authentication Flow](#)

The web server authentication flow is used by apps that are hosted on a secure server. A critical aspect of the web server flow is that the server must be able to protect the consumer secret. You can use code challenge and verifier values in the flow to prevent authorization code interception.

### [Understanding the User-Agent OAuth Authentication Flow](#)

The user-agent authentication flow is used by client apps (consumers) that reside on the user's device or computer. It's also used by client apps running in a browser using a scripting language such as JavaScript. These apps can protect per-user secrets. But, because the apps are widely distributed, the client secret can't be confidential.

### [Understanding the Username-Password OAuth Authentication Flow](#)

Use the username-password authentication flow to authenticate when the consumer already has the user's credentials.

### [Understanding the OAuth Refresh Token Process](#)

The Web server OAuth authentication flow and user-agent flow both provide a refresh token that can be used to obtain a new access token.

### [Finding Additional Resources](#)

## Defining Connected Apps

To authenticate using OAuth, you must create a connected app that defines your application's OAuth settings for the Salesforce organization.

When you develop an external application that needs to authenticate with Salesforce, you need to define it as a new connected app within the Salesforce organization that informs Salesforce of this new authentication entry point.

Use the following steps to create a new connected app.

1. From Setup, enter *Apps* in the *Quick Find* box, then select **Apps** and click **New** in the Connected Apps section of the page to start defining a connected app.
2. Enter the name of your application.
3. Enter the contact email information, as well as any other information appropriate for your application.
4. Select **Enable OAuth Settings**.
5. Enter a *Callback URL*. Depending on which OAuth flow you use, this is typically the URL that a user's browser is redirected to after successful authentication. As this URL is used for some OAuth flows to pass an access token, the URL must use secure HTTP (HTTPS) or a custom URI scheme.
6. Add all supported OAuth scopes to **Selected OAuth Scopes**. These scopes refer to permissions given by the user running the connected app.
7. Enter a URL for *Info URL*. This is where the user can go for more information about your application.
8. Click **Save**. The *Consumer Key* is created and displayed, and the *Consumer Secret* is created (click the link to reveal it).

Once you define a connected app, you use the consumer key and consumer secret to authenticate your application. See [Creating a Connected App](#) in the Salesforce online help for specific steps to create a connected app for the type of authentication you need.

## Understanding OAuth Endpoints

OAuth endpoints are the URLs you use to make OAuth authentication requests to Salesforce.

You need to use the correct Salesforce OAuth endpoint when issuing authentication requests in your application. The primary OAuth endpoints are:

- For authorization: `https://login.salesforce.com/services/oauth2/authorize`
- For token requests: `https://login.salesforce.com/services/oauth2/token`
- For revoking OAuth tokens: `https://login.salesforce.com/services/oauth2/revoke`

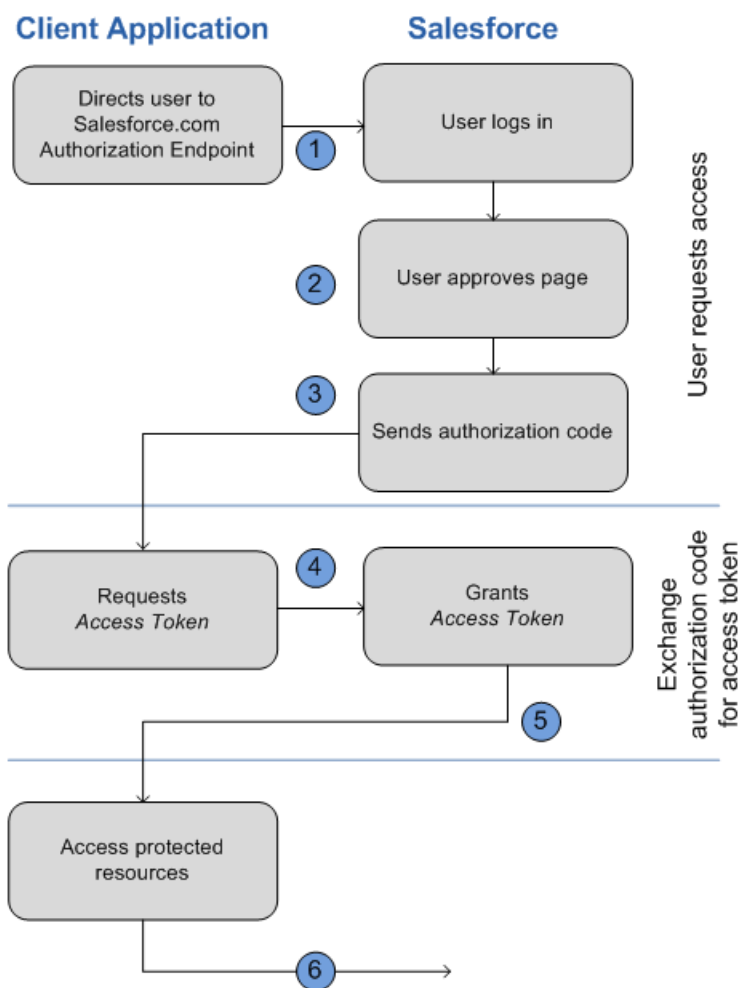
All endpoints require secure HTTP (HTTPS). Each OAuth flow defines which endpoints you need to use and what request data you need to provide.

If you're verifying authentication on a sandbox organization, use "test.salesforce.com" instead of "login.salesforce.com" in all the OAuth endpoints listed above.

## Understanding the Web Server OAuth Authentication Flow

The web server authentication flow is used by apps that are hosted on a secure server. A critical aspect of the web server flow is that the server must be able to protect the consumer secret. You can use code challenge and verifier values in the flow to prevent authorization code interception.

In this flow, the client application requests the authorization server to redirect the user to another web server or resource that authorizes the user and sends the application an authorization code. The application uses the authorization code to request an access token. The following shows the steps for this flow.




1. The application redirects the user to the appropriate Salesforce authorization endpoint, such as <https://login.salesforce.com/services/oauth2/authorize>. The following parameters are required:

Parameter	Description
<code>response_type</code>	Must be <code>code</code> for this authentication flow.



Parameter	Description
<code>client_id</code>	The <code>Consumer Key</code> from the connected app definition.
<code>redirect_uri</code>	The <code>Callback URL</code> from the connected app definition.

The following parameters are optional:

Parameter	Description
<code>code_challenge</code>	<p>Specifies the SHA256 hash value of the <code>code_verifier</code> value in the token request to help prevent authorization code interception attacks. The hash value must be base64url encoded as defined here:  <a href="https://tools.ietf.org/html/rfc4648#section-5">https://tools.ietf.org/html/rfc4648#section-5</a>.</p> <ul style="list-style-type: none"> <li>If the <code>code_challenge</code> value is provided in the authorization request and a <code>code_verifier</code> value is provided in the token request, Salesforce compares the <code>code_challenge</code> to the <code>code_verifier</code>. If the <code>code_challenge</code> is invalid or doesn't match, the login fails with the <code>invalid_request</code> error code.</li> <li>If the <code>code_challenge</code> value is provided in the authorization request, but a <code>code_verifier</code> value is not provided in the token request, the login fails with the <code>invalid_grant</code> error code.</li> </ul> <p> <b>Note:</b> The value should be base64url-encoded only once.</p>
<code>display</code>	<p>Changes the login page's display type. Valid values are:</p> <ul style="list-style-type: none"> <li><code>page</code>—Full-page authorization screen. This is the default value if none is specified.</li> <li><code>popup</code>—Compact dialog optimized for modern Web browser popup windows.</li> <li><code>touch</code>—Mobile-optimized dialog designed for modern smartphones such as Android and iPhone.</li> <li><code>mobile</code>—Mobile optimized dialog designed for smartphones such as BlackBerry OS 5 that don't support touch screens.</li> </ul>
<code>immediate</code>	<p>Determines whether the user should be prompted for login and approval. Values are either <code>true</code> or <code>false</code>. Default is <code>false</code>.</p> <ul style="list-style-type: none"> <li>If set to <code>true</code>, and if the user is currently logged in and has previously approved the application, the approval step is skipped.</li> <li>If set to <code>true</code> and the user is not logged in or has not previously approved the application, the session is</li> </ul>

Parameter	Description
	immediately terminated with the <code>immediate_unsuccessful</code> error code.
<code>login_hint</code>	Provides a valid username value to pre-populate the login page with the username. For example: <code>login_hint=username@company.com</code> . If a user already has an active session in the browser, then the <code>login_hint</code> parameter does nothing; the active user session continues.
<code>nonce</code>	Specifies a value to be returned in the response; this is useful for detecting "replay" attacks. Optional with the <code>openid</code> scope for getting a user ID token.
<code>prompt</code>	<p>Specifies how the authorization server prompts the user for reauthentication and reapproval. This parameter is optional. The only values Salesforce supports are:</p> <ul style="list-style-type: none"> <li><code>login</code>—The authorization server <i>must</i> prompt the user for reauthentication, forcing the user to log in again.</li> <li><code>consent</code>—The authorization server <i>must</i> prompt the user for reapproval before returning information to the client.</li> </ul> <p>It is valid to pass both values, separated by a space, to require the user to both log in and reauthorize. For example:</p> <pre>?prompt=login%20consent</pre>
<code>scope</code>	Specifies what data your application can access. See “Scope Parameter Values” in the online help for more information.
<code>state</code>	Specifies any additional URL-encoded state data to be returned in the callback URL after approval.

An example authorization URL might look something like the following:

```
https://login.salesforce.com/services/oauth2/authorize?response_type=code
&client_id=3MVG9lKcPoNINVBIPJjdwlJ9LLM82HnFVVX19KY1uA5mu0QqEWhqKpoW3svG3X
HrXDICQjKlmdgAvhCscA9GE&redirect_uri=https%3A%2F%2Fwww.mysite.com%2F
code_callback.jsp&state=mystate
```

- The user logs into Salesforce with their credentials. The user is interacting with the authorization endpoint directly, so the application never sees the user's credentials. After successfully logging in, the user is asked to authorize the application. Note that if the user has already authorized the application, this step is skipped.
- After Salesforce confirms that the client application is authorized, the end-user's Web browser is redirected to the callback URL specified by the `redirect_uri` parameter. Salesforce appends authorization information to the redirect URL with the following values:

Parameters	Description
code	Authorization code the consumer must use to obtain the access and refresh tokens.
state	The state value that was passed in as part of the initial request, if applicable.

An example callback URL with authorization information might look something like:


```
https://www.mysite.com/authcode_callback?code=aWekysIEeqM9PiT
hEfM0Cnr6MoLIfwWyRJcqOqHdF8f9INokharAS09ia7UNP6RiVScerfhc4w%3D%3D
```

- The application extracts the authorization code and passes it in a request to Salesforce for an access token. This request is a POST request sent to the appropriate Salesforce token request endpoint, such as `https://login.salesforce.com/services/oauth2/token`. The following parameters are required:

Parameter	Description
grant_type	Value must be <code>authorization_code</code> for this flow.
client_secret	The Consumer Secret from the connected app definition. Required unless the Require Secret for Web Server Flow setting is not enabled in the connected app definition.
client_id	The Consumer Key from the connected app definition.
redirect_uri	The Callback URL from the connected app definition.
code	Authorization code the consumer must use to obtain the access and refresh tokens.

The following parameters are optional:


Parameter	Description
client_assertion	Instead of passing in <code>client_secret</code> you can choose to provide a <code>client_assertion</code> and <code>client_assertion_type</code> . If a <code>client_secret</code> parameter is not provided, Salesforce checks for the <code>client_assertion</code> and <code>client_assertion_type</code> automatically. The value of <code>client_assertion</code> must be a typical JWT bearer token, signed with the private key associated with the OAuth consumer's uploaded certificate. Only the RS256 algorithm is currently supported. For more information on using <code>client_assertion</code> , see the OpenID Connect specifications for the <code>private_key_jwt</code> client authentication method.

Parameter	Description
<code>client_assertion_type</code>	Provide this value when using the <code>client_assertion</code> parameter. The value of <code>client_assertion_type</code> must be <code>urn:ietf:params:oauth:client-assertion-type:jwt-bearer</code> .
<code>code_verifier</code>	<p>Specifies 128 bytes of random data with high enough entropy to make it difficult to guess the value to help prevent authorization code interception attacks. The value also must be base64url encoded as defined here: <a href="https://tools.ietf.org/html/rfc4648#section-5">https://tools.ietf.org/html/rfc4648#section-5</a>.</p> <ul style="list-style-type: none"> <li>If the <code>code_verifier</code> value is provided in the token request and a <code>code_challenge</code> value is in the authorization request, Salesforce compares the <code>code_verifier</code> to the <code>code_challenge</code>. If the <code>code_verifier</code> is invalid or doesn't match, the login fails with the <code>invalid_grant</code> error code.</li> <li>If the <code>code_verifier</code> value is provided in the token request, but a <code>code_challenge</code> value was not provided in the authorization request, the login fails with the <code>invalid_grant</code> error code.</li> </ul> <p> <b>Note:</b> The value should be base64url-encoded only once.</p>
<code>format</code>	<p>Expected return format. The default is <code>json</code>. Values are:</p> <ul style="list-style-type: none"> <li><code>urlencoded</code></li> <li><code>json</code></li> <li><code>xml</code></li> </ul> <p>The return format can also be specified in the header of the request using one of the following:</p> <ul style="list-style-type: none"> <li><code>Accept: application/x-www-form-urlencoded</code></li> <li><code>Accept: application/json</code></li> <li><code>Accept: application/xml</code></li> </ul>

An example access token POST request might look something like:

```
POST /services/oauth2/token HTTP/1.1
Host: login.salesforce.com
grant_type=authorization_code&code=aPrxsmIEeqM9PiQroGEWx1UiMQd95_5JUJ
VEhsOFhS8EVvbfYBBJ1i2W5fn3zbo.8hojaNW_1g%3D%3D&client_id=3MVG9lKcPoNI
NVBIPJjdwlJ9LLM82HnFVVX19KY1uA5mu0QqEWhqKpoW3svG3XHrXDicQjK1mdgAvhCs
cA9GE&client_secret=1955279925675241571&
redirect_uri=https%3A%2F%2Fwww.mysite.com%2Fcode_callback.jsp
```

- If this request is successful, the server returns a response body that contains the following:

Parameters	Description
<code>access_token</code>	Access token that acts as a session ID that the application uses for making requests. This token should be protected as though it were user credentials.
<code>refresh_token</code>	Token that can be used in the future to obtain new access tokens.   <b>Warning:</b> This value is a secret. You should treat it like the user's password and use appropriate measures to protect it.
<code>instance_url</code>	Identifies the Salesforce instance to which API calls should be sent.
<code>id</code>	Identity URL that can be used to both identify the user as well as query for more information about the user. Can be used in an HTTP request to get more information about the end user.
<code>issued_at</code>	When the signature was created, represented as the number of seconds since the Unix epoch (00:00:00 UTC on 1 January 1970).
<code>signature</code>	Base64-encoded HMAC-SHA256 signature signed with the consumer's private key containing the concatenated ID and <code>issued_at</code> value. The <code>signature</code> can be used to verify that the identity URL wasn't modified because it was sent by the server.

An example JSON response body might look something like:

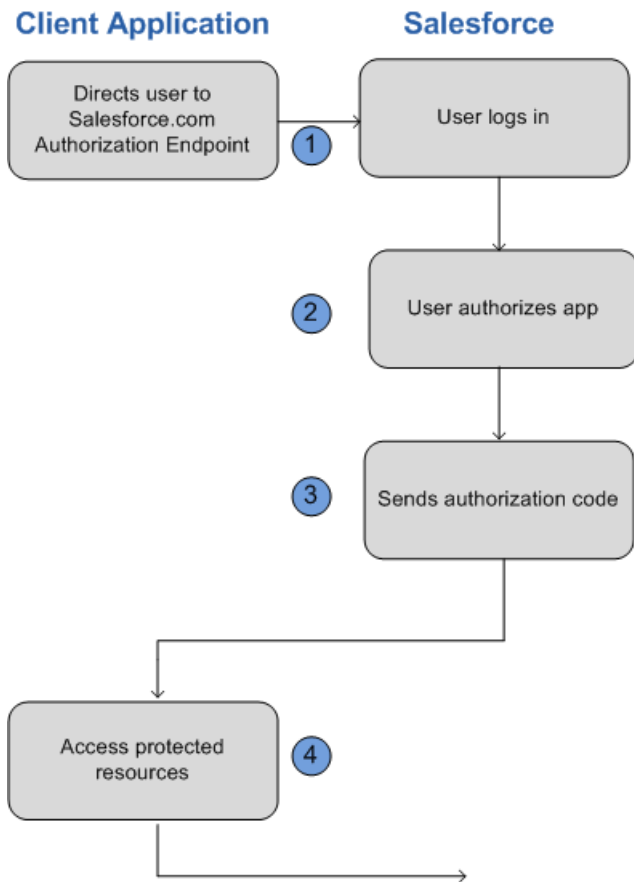
```
{
  "id": "https://login.salesforce.com/id/00Dx0000000BV7z/005x00000012Q9P",
  "issued_at": "1278448101416",
  "refresh_token": "5Aep8614iLM.Dq661ePDmPEgaAW9Oh_L3JKkDpB4xReb54_pZebnUG0h6Sb4KUVDpNtWEofWM39yg==",
  "instance_url": "https://yourInstance.salesforce.com/",
  "signature": "CMJ4l+CCaPQiKjoOEwEig9H4wqhpuLSk4J2urAe+fVg=",
  "access_token": "00Dx0000000BV7z!AR8AQp0jITN80ESESj5EbaZTFG0RNBaT1cyWk7TrqoDjoNIWQ2ME_sTZzBjfmOE6zMHq6y8PIW4eWze9JksNEkWU1.Cju7m4"
}
```

- The application uses the provided access token and refresh token to access protected user data.

## Understanding the User-Agent OAuth Authentication Flow

The user-agent authentication flow is used by client apps (consumers) that reside on the user's device or computer. It's also used by client apps running in a browser using a scripting language such as JavaScript. These apps can protect per-user secrets. But, because the apps are widely distributed, the client secret can't be confidential.

In this flow, the client application requests the authorization server to redirect the user to another Web server or resource which is capable of extracting the access token and passing it back to the application. The following shows the steps for this flow.



1. The application redirects the user to the appropriate Salesforce authorization endpoint, such as <https://login.salesforce.com/services/oauth2/authorize>. The following parameters are required:

Parameter	Description
<code>response_type</code>	Must be <code>token</code> for this authentication flow
<code>client_id</code>	The <code>Consumer Key</code> from the connected app definition.
<code>redirect_uri</code>	The <code>Callback URL</code> from the connected app definition.

The following parameters are optional:


Parameter	Description
<code>display</code>	<p>Changes the login page's display type. Valid values are:</p> <ul style="list-style-type: none"> <li><code>page</code>—Full-page authorization screen. This is the default value if none is specified.</li> <li><code>popup</code>—Compact dialog optimized for modern Web browser popup windows.</li> <li><code>touch</code>—Mobile-optimized dialog designed for modern smartphones such as Android and iPhone.</li> </ul>

Parameter	Description
	<ul style="list-style-type: none"> <li><b>mobile</b>—Mobile optimized dialog designed for smartphones such as BlackBerry OS 5 that don't support touch screens.</li> </ul>
scope	Specifies what data your application can access. See "Scope Parameter Values" in the online help for more information.
state	Specifies any additional URL-encoded state data to be returned in the callback URL after approval.

An example authorization URL might look something like the following:

```
https://login.salesforce.com/services/oauth2/authorize?response_type=token&
client_id=3MVG91KcPoNINVBIPJjdwlJ9LLJbP_pqwoJYyuisjQhr_LLurNDv7AgQvDTZwCoZuD
ZrXcPCmBv4o.8ds.5iE&redirect_uri=https%3A%2F%2Fwww.mysite.com%2Fuser_callback.jsp&
state=ystate
```

- The user logs into Salesforce with their credentials. The user interacts with the authorization endpoint directly, so the application never sees the user's credentials.
- Once authorization is granted, the authorization endpoint redirects the user to the redirect URL. This URL is defined in the remote access application created for the application. Salesforce appends access token information to the redirect URL with the following values:

Parameters	Description
access_token	Access token that acts as a session ID that the application uses for making requests. This token should be protected as though it were user credentials.
expires_in	Amount of time the access token is valid, in seconds.
refresh_token	<p>Token that can be used in the future to obtain new access tokens.</p> <p> <b>Warning:</b> This value is a secret. You should treat it like the user's password and use appropriate measures to protect it.</p> <p>The refresh token is only returned if the redirect URI is <code>https://login.salesforce.com/services/oauth2/success</code> or used with a custom protocol that is not HTTPS.</p>
state	The state value that was passed in as part of the initial request, if applicable.
instance_url	Identifies the Salesforce instance to which API calls should be sent.
id	Identity URL that can be used to both identify the user as well as query for more information about the user. Can be used in an HTTP request to get more information about the end user.

Parameters	Description
<code>issued_at</code>	When the signature was created, represented as the number of seconds since the Unix epoch (00:00:00 UTC on 1 January 1970).
<code>signature</code>	Base64-encoded HMAC-SHA256 signature signed with the consumer's private key containing the concatenated ID and <code>issued_at</code> value. The <code>signature</code> can be used to verify that the identity URL wasn't modified because it was sent by the server.

An example callback URL with access information appended after the hash sign (#) might look something like:

```
https://www.mysite.com/user_callback.jsp#access_token=00Dx000000BV7z%21AR8
AQBm8J_xr9kLqmZIRyQxZgLCm4HVi4laGtW0qW3JCzf5xdTGGGSovim8FfJkZEqxbjaFbberKGk
8v8AnYrvChG4qJbQo8&refresh_token=5Aep8614iLM.Dq661ePDmPEgaAW9Oh_L3JKkDpB4xR
eb54_pZfVtildPEk8aimw4Hr9ne7VXXVSIQ%3D%3D&expires_in=7200&state=mystate
```

4. The application uses the provided access token and refresh token to access protected user data.

Keep the following considerations in mind when using the user-agent OAuth flow:

- Because the access token is encoded into the redirection URI, it might be exposed to the end-user and other applications residing on the computer or device. If you're authenticating using JavaScript, call `window.location.replace()` to remove the callback from the browser's history.

## Understanding the Username-Password OAuth Authentication Flow

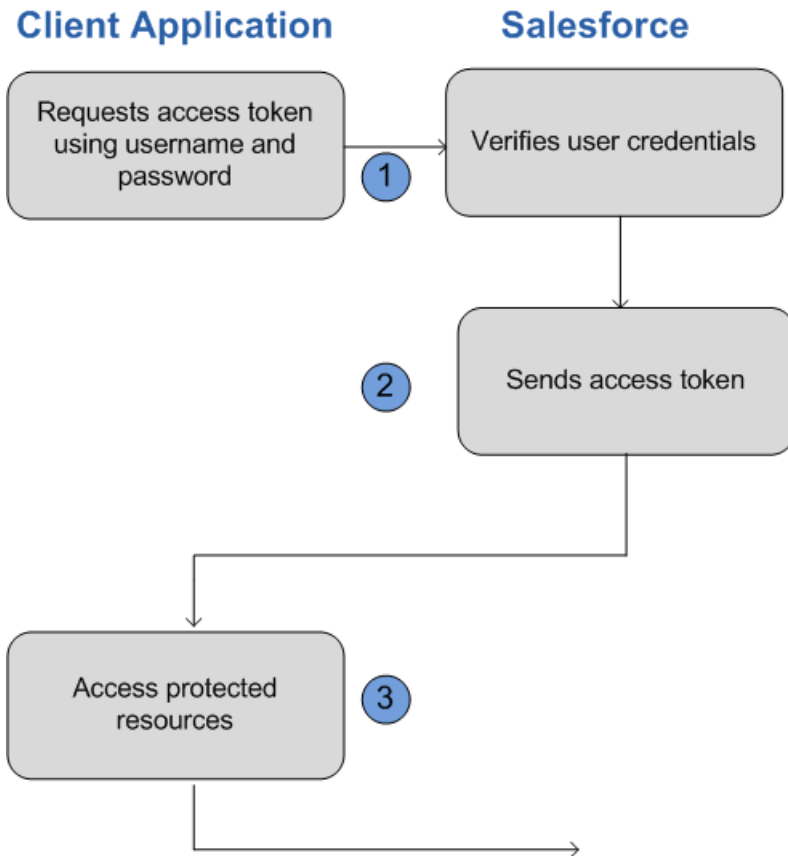
Use the username-password authentication flow to authenticate when the consumer already has the user's credentials.

In this flow, the user's credentials are used by the application to request an access token as shown in the following steps.



**Warning:** This OAuth authentication flow passes the user's credentials back and forth. Use this authentication flow only when necessary. No refresh token is issued.





1. The application uses the user's username and password to request an access token. This is done via an out-of-band POST request to the appropriate Salesforce token request endpoint, such as `https://login.salesforce.com/services/oauth2/token`. These request fields are required:

Parameter	Description
<code>grant_type</code>	Must be <code>password</code> for this authentication flow.
<code>client_id</code>	The <code>Consumer Key</code> from the connected app definition.
<code>client_secret</code>	The <code>Consumer Secret</code> from the connected app definition. Required unless the <code>Require Secret for Web Server Flow</code> setting is not enabled in the connected app definition.
<code>username</code>	End-user's username.
<code>password</code>	End-user's password.



**Note:** You must append the user's security token to their password. A security token is an automatically-generated key from Salesforce. For example, if a user's password is `mypassword`, and their security token is `XXXXXXXXXX`, then the value provided for this parameter must be `mypasswordXXXXXXXXXX`. For more information on

Parameter	Description
	security tokens see “Reset Your Security Token” in the online help.

An example request body might look something like the following:

```
grant_type=password&client_id=3MVG9lKcPoNINVBIPJjdwlJ9LLM82Hn
FVVX19KY1uA5mu0QqEWhqKpoW3svG3XHrXDICQjKlmdgAvhCscA9GE&client_secret=
1955279925675241571&username=testuser%40salesforce.com&password=mypassword123456
```

2. Salesforce verifies the user credentials, and if successful, sends a response to the application with the access token. This response contains the following values:

Parameters	Description
access_token	Access token that acts as a session ID that the application uses for making requests. This token should be protected as though it were user credentials.
instance_url	Identifies the Salesforce instance to which API calls should be sent.
id	Identity URL that can be used to both identify the user as well as query for more information about the user. Can be used in an HTTP request to get more information about the end user.
issued_at	When the signature was created, represented as the number of seconds since the Unix epoch (00:00:00 UTC on 1 January 1970).
signature	Base64-encoded HMAC-SHA256 signature signed with the consumer's private key containing the concatenated ID and issued_at value. The signature can be used to verify that the identity URL wasn't modified because it was sent by the server.

An example response body might look something like:

```
{ "id": "https://login.salesforce.com/id/00Dx0000000BV7z/005x00000012Q9P",
  "issued_at": "1278448832702", "instance_url": "https://yourInstance.salesforce.com/",
  "signature": "0CmxinZir53Yex7nE0TD+zMpvIWYGb/bdJh6XfOH6EQ=", "access_token":
  "00Dx0000000BV7z!AR8AQAxo9UfVkh8AlV0Gomt9Czx9LjHnSSpwBMbRcgKFmxOtvxjTrKW1
  9ye6PE3Ds1eQz3z8jr3W7_VbWmEu4Q8TVGSTHxs" }
```

3. The application uses the provided access token to access protected user data.

Keep the following considerations in mind when using the username-password OAuth flow:

- Since the user is never redirected to login at Salesforce in this flow, the user can't directly authorize the application, so no refresh tokens can be used. If your application requires refresh tokens, you should consider using the Web server or user-agent OAuth flow.

## Understanding the OAuth Refresh Token Process

The Web server OAuth authentication flow and user-agent flow both provide a refresh token that can be used to obtain a new access token.

Access tokens have a limited lifetime specified by the session timeout in Salesforce. If an application uses an expired access token, a “Session expired or invalid” error is returned. If the application is using the Web server or user-agent OAuth authentication flows, a refresh token may be provided during authorization that can be used to get a new access token.

The client application obtains a new access token by sending a POST request to the token request endpoint with the following request parameters:

Parameters	Description
<code>grant_type</code>	Value must be <code>refresh_token</code> .
<code>refresh_token</code>	The refresh token the client application already received.
<code>client_id</code>	The <code>Consumer Key</code> from the connected app definition.
<code>client_secret</code>	The <code>Consumer Secret</code> from the connected app definition. Required unless the <code>Require Secret for Web Server Flow</code> setting is not enabled in the connected app definition. This parameter is optional.
<code>format</code>	<p>Expected return format. The default is <code>json</code>. Values are:</p> <ul style="list-style-type: none"><li>• <code>urlencoded</code></li><li>• <code>json</code></li><li>• <code>xml</code></li></ul> <p>The return format can also be specified in the header of the request using one of the following:</p> <ul style="list-style-type: none"><li>• <code>Accept: application/x-www-form-urlencoded</code></li><li>• <code>Accept: application/json</code></li><li>• <code>Accept: application/xml</code></li></ul> <p>This parameter is optional.</p>

An example refresh token POST request might look something like:

```
POST /services/oauth2/token HTTP/1.1
Host: https://login.salesforce.com/
grant_type=refresh_token&client_id=3MVG9lKcPoNINVBIPJjdW1J9LLM82HnFVVX19KY1uA5mu0
QqEWhqKpoW3svG3XHrXDiCQjKlmdgAvhCscA9GE&client_secret=1955279925675241571
&refresh_token=your token here
```

Once Salesforce verifies the refresh token request, it sends a response to the application with the following response body parameters:

Parameters	Description
<code>access_token</code>	Access token that acts as a session ID that the application uses for making requests. This token should be protected as though it were user credentials.
<code>instance_url</code>	Identifies the Salesforce instance to which API calls should be sent.
<code>id</code>	Identity URL that can be used to both identify the user as well as query for more information about the user. Can be used in an HTTP request to get more information about the end user.
<code>issued_at</code>	When the signature was created, represented as the number of seconds since the Unix epoch (00:00:00 UTC on 1 January 1970).
<code>signature</code>	Base64-encoded HMAC-SHA256 signature signed with the consumer's private key containing the concatenated ID and <code>issued_at</code> value. The <code>signature</code> can be used to verify that the identity URL wasn't modified because it was sent by the server.

An example JSON response body might look something like:

```
{ "id": "https://login.salesforce.com/id/00Dx0000000BV7z/005x00000012Q9P",
  "issued_at": "1278448384422", "instance_url": "https://yourInstance.salesforce.com/",
  "signature": "SSSbLO/gBhmmYNUvN18ODBDFYHzakxOMgqYtu+hDPsc=",
  "access_token": "00Dx0000000BV7z!AR8AQp0jITN80ESESj5EbaZTFG0RNBaT1cyWk7T
  rqoDjoNIWQ2ME_sTZzBjfmOE6zMHq6y8PIW4eWze9JksNEkWU1.Cju7m4" }
```

Keep in mind the following considerations when using the refresh token OAuth process:

- The session timeout for an access token can be configured in Salesforce from Setup by entering *Session Settings* in the Quick Find box, then selecting **Session Settings**.
- If the application uses the username-password OAuth authentication flow, no refresh token is issued, as the user cannot authorize the application in this flow. If the access token expires, the application using username-password OAuth flow must re-authenticate the user.

## Finding Additional Resources

The following resources provide additional information about using OAuth with Salesforce:

- [Authenticate Apps with OAuth](#)
- [Digging Deeper into OAuth on Force.com](#)
- [Using OAuth to Authorize External Applications](#)

The following resources are examples of third party client libraries that implement OAuth that you might find useful:

- For Ruby on Rails: [OmniAuth](#)
- For Java: [Apache Amber](#)
- Additional OAuth client libraries: [OAuth.net](#)

# Understanding Place Order REST API Resources

---

Integrate your order creation system with Salesforce by using the Place Order REST API.

Use this API to:

## IN THIS SECTION:

[Add a Contract and Orders to an Existing Account](#)

[Add an Order to an Existing Account](#)

[Add Orders to an Existing Contract](#)

[Add Order Products to an Existing Order](#)

[Get Details About a Contract](#)

[Get Details About an Order](#)

[Filter Details About a Contract](#)

[Filter Details About an Order](#)

## Add a Contract and Orders to an Existing Account

Here's an example of a POST request using the [Create Contract-based Orders](#) resource to create a contract—with child orders, order products, and custom objects—to an existing account.

### Example usage

```
/services/data/v30.0/commerce/sale
```

### Example request body

```
{
  "contract": [
    {
      "attributes": {
        "type": "Contract"
      },
      "AccountId": "001D000000JRDAv",
      "StartDate": "2013-08-01",
      "Status": "Draft",
      "ContractTerm": "1",
      "Test_Contract1__r": {
        "records": [
          {
            "attributes": {
              "type": "Test_Contract1__c"
            },
            "Name": "Contract Custom Object"
          }
        ]
      }
    }
  ],
  "Orders": {
    "records": [
      {

```

```

        "attributes": {
            "type": "Order"
        },
        "EffectiveDate": "2013-08-11",
        "Status": "Draft",
        "billingCity": "SFO-Inside-OrderEntity-1",
        "Pricebook2Id": "01sD0000000G2NjIAK",
        "OrderItems": {
            "records": [
                {
                    "attributes": {
                        "type": "OrderItem"
                    },
                    "PricebookEntryId": "01uD00000001c5toIAA",
                    "quantity": "1",
                    "UnitPrice": "10"
                }
            ]
        }
    }
}

```

#### Example JSON response body

```

{
    "totalSize" : 1,
    "done" : true,
    "records" : [ {
        "attributes" : {
            "type" : "Contract",
            "url" : "/services/data/v30.0/subjects/Contract/800D0000000P1cMIAW"
        },
        "Id" : "800D0000000P1cMIAW",
        "Orders" : {
            "totalSize" : 1,
            "done" : true,
            "records" : [ {
                "attributes" : {
                    "type" : "Order",
                    "url" : "/services/data/v30.0/subjects/Order/801D0000000G0ylIAC"
                },
                "Id" : "801D0000000G0ylIAC",
                "OrderItems" : {
                    "totalSize" : 1,
                    "done" : true,
                    "records" : [ {
                        "attributes" : {
                            "type" : "OrderItem",
                            "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CKoyIAG"
                        },
                        "Id" : "802D0000000CKoyIAG"
                    }
                ]
            }
        ]
    }
}

```

```

        } ]
    }
} ]
},
"Test_Contract1__r" : {
    "totalSize" : 1,
    "done" : true,
    "records" : [ {
        "attributes" : {
            "type" : "Test_Contract1__c",
            "url" : "/services/data/v30.0/subjects/Test_Contract1__c/a02D0000006YYKZIA4"
        },
        "Id" : "a02D0000006YYKZIA4"
    } ]
}
} ]
}
} ]
}

```

## Add an Order to an Existing Account

Here's an example of a POST request using the [Create Order](#) resource to create an order with order products for an existing account.

### Example usage

```
/services/data/v30.0/commerce/sale/order
```

### Example request body

```

{
  "order": [
    {
      "attributes": {
        "type": "Order"
      },
      "EffectiveDate": "2013-07-11",
      "Status": "Draft",
      "billingCity": "SFO-Inside-OrderEntity-1",
      "accountId": "001D000000JRDAv",
      "Pricebook2Id": "01sD0000000G2NjIAK",
      "OrderItems": {
        "records": [
          {
            "attributes": {
              "type": "OrderItem"
            },
            "PricebookEntryId": "01uD00000001c5toIAA",
            "quantity": "1",
            "UnitPrice": "15.99"
          }
        ]
      }
    }
  ]
}

```

**Example JSON response body**

```
{
  "totalSize" : 1,
  "done" : true,
  "records" : [ {
    "attributes" : {
      "type" : "Order",
      "url" : "/services/data/v30.0/subjects/Order/801D0000000G0ySIAS"
    },
    "Id" : "801D0000000G0ySIAS",
    "OrderItems" : {
      "totalSize" : 1,
      "done" : true,
      "records" : [ {
        "attributes" : {
          "type" : "OrderItem",
          "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CKp8IAG"
        },
        "Id" : "802D0000000CKp8IAG"
      } ]
    }
  } ]
}
```

## Add Orders to an Existing Contract

Here's an example of a PATCH request using the [Contract-based Orders](#) resource to add new orders and order products to an existing contract.

**Example usage**

```
/services/data/v30.0/commerce/sale/800D0000000PFL8IAO
```

**Example request body**

```
{
  "contract": [
    {
      "attributes": {
        "type": "Contract"
      },
      "Id" : "800D0000000PFL8IAO",
      "Orders": {
        "records": [
          {
            "attributes": {
              "type": "Order"
            },
            "EffectiveDate": "2013-08-11",
            "Status": "Draft",
            "billingCity": "SFO-Inside-OrderEntity-1",
            "contractId": "800D0000000PFL8IAO",
            "pricebook2Id": "01sD0000000G2JbIAK",
            "OrderItems": {
```



```

    "records": [
      {
        "attributes": {
          "type": "OrderItem"
        },
        "PricebookEntryId": "01uD0000001c5tjIAA",
        "quantity": "12",
        "UnitPrice": "10"
      },
      {
        "attributes": {
          "type": "OrderItem"
        },
        "PricebookEntryId": "01uD0000001cAkMIAU",
        "quantity": "2",
        "UnitPrice": "20"
      }
    ]
  },
  {
    "attributes": {
      "type": "Order"
    },
    "EffectiveDate": "2013-10-11",
    "Status": "Draft",
    "billingCity": "SFO-Inside-OrderEntity-1",
    "contractId": "800D0000000PFL8IAO",
    "pricebook2Id": "01sD0000000G2JbIAK",
    "OrderItems": {
      "records": [
        {
          "attributes": {
            "type": "OrderItem"
          },
          "PricebookEntryId": "01uD0000001cAkRIAUI",
          "quantity": "11",
          "UnitPrice": "10"
        },
        {
          "attributes": {
            "type": "OrderItem"
          },
          "PricebookEntryId": "01uD0000001cAkWIAU",
          "quantity": "2",
          "UnitPrice": "20"
        },
        {
          "attributes": {
            "type": "OrderItem"
          },
          "PricebookEntryId": "01uD0000001cAkgIAE",
          "quantity": "14",
          "UnitPrice": "10"
        }
      ]
    }
  }
]

```

```

    }
  ]
}

```

### Example JSON response body

```

{
  "totalSize" : 1,
  "done" : true,
  "records" : [ {
    "attributes" : {
      "type" : "Contract"
    },
    "Orders" : {
      "totalSize" : 2,
      "done" : true,
      "records" : [ {
        "attributes" : {
          "type" : "Order",
          "url" : "/services/data/v29.0/subjects/Order/801D0000000G0xsIAC"
        },
        "Id" : "801D0000000G0xsIAC",
        "OrderItems" : {
          "totalSize" : 2,
          "done" : true,
          "records" : [ {
            "attributes" : {
              "type" : "OrderItem",
              "url" : "/services/data/v29.0/subjects/OrderItem/802D0000000CKoPIAW"
            },
            "Id" : "802D0000000CKoPIAW"
          }, {
            "attributes" : {
              "type" : "OrderItem",
              "url" : "/services/data/v29.0/subjects/OrderItem/802D0000000CKoQIAW"
            },
            "Id" : "802D0000000CKoQIAW"
          } ]
        }
      } ],
      {
        "attributes" : {
          "type" : "Order",
          "url" : "/services/data/v29.0/subjects/Order/801D0000000G0xtIAC"
        },
        "Id" : "801D0000000G0xtIAC",
        "OrderItems" : {
          "totalSize" : 3,
          "done" : true,
          "records" : [ {

```

```

        "attributes" : {
            "type" : "OrderItem",
            "url" : "/services/data/v29.0/subjects/OrderItem/802D0000000CKoRIAW"
        },
        "Id" : "802D0000000CKoRIAW"
    }, {
        "attributes" : {
            "type" : "OrderItem",
            "url" : "/services/data/v29.0/subjects/OrderItem/802D0000000CKoSIAW"
        },
        "Id" : "802D0000000CKoSIAW"
    }, {
        "attributes" : {
            "type" : "OrderItem",
            "url" : "/services/data/v29.0/subjects/OrderItem/802D0000000CKoTIAW"
        },
        "Id" : "802D0000000CKoTIAW"
    } ]
}
} ]
}
} ]
}

```

## Add Order Products to an Existing Order

Here's an example of a PATCH request using the [Order](#) resource to add order products to an existing order.

### Example usage

```
/services/data/v30.0/commerce/sale/order/801D0000000Frh8
```

### Example request body

```

{
  "order": [
    {
      "attributes": {
        "type": "Order"
      },
      "Id": "801D0000000Frh8",
      "OrderItems": {
        "records": [
          {
            "attributes": {
              "type": "OrderItem"
            },
            "PricebookEntryId": "01uD0000001cCd1",
            "quantity": "1",
            "UnitPrice": "100",
            "orderId": "801D0000000Frh8"
          },
          {
            "attributes": {
              "type": "OrderItem"
            }
          }
        ]
      }
    }
  ]
}

```

```

    },
    "PricebookEntryId": "01uD0000001cCd1",
    "quantity": "2",
    "UnitPrice": "200",
    "orderId": "801D00000000Frh8"
  }
]
}
}
]
}

```

### Example JSON response body

```

{
  "totalSize" : 1,
  "done" : true,
  "records" : [ {
    "attributes" : {
      "type" : "Order"
    },
    "OrderItems" : {
      "totalSize" : 2,
      "done" : true,
      "records" : [ {
        "attributes" : {
          "type" : "OrderItem",
          "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CJtMIAW"
        },
        "Id" : "802D0000000CJtMIAW"
      }, {
        "attributes" : {
          "type" : "OrderItem",
          "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CJtNIAW"
        },
        "Id" : "802D0000000CJtNIAW"
      } ]
    }
  } ]
}

```

## Get Details About a Contract

Here's an example of a GET request using the [Contract-based Orders](#) resource to query details about a contract and its child orders, order products, and custom objects.

### Example usage

```
/services/data/v30.0/commerce/sale/800D0000000PFHp
```

### Example request body

None

**Example JSON response body**

```
{
  "totalSize" : 1,
  "done" : true,
  "records" : [ {
    "attributes" : {
      "type" : "Contract",
      "url" : "/services/data/v30.0/subjects/Contract/800D0000000PFHpIAO"
    },
    "Id" : "800D0000000PFHpIAO",
    "Orders" : {
      "totalSize" : 4,
      "done" : true,
      "records" : [ {
        "attributes" : {
          "type" : "Order",
          "url" : "/services/data/v30.0/subjects/Order/801D0000000FqzsIAC"
        },
        "Id" : "801D0000000FqzsIAC",
        "OrderItems" : {
          "totalSize" : 3,
          "done" : true,
          "records" : [ {
            "attributes" : {
              "type" : "OrderItem",
              "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CJX0IAO"
            },
            "Id" : "802D0000000CJX0IAO"
          }, {
            "attributes" : {
              "type" : "OrderItem",
              "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CJYDIA4"
            },
            "Id" : "802D0000000CJYDIA4"
          }, {
            "attributes" : {
              "type" : "OrderItem",
              "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CKFCIA4"
            },
            "Id" : "802D0000000CKFCIA4"
          } ]
        },
        "Custom_Objects__r" : null
      }, {
        "attributes" : {
          "type" : "Order",
          "url" : "/services/data/v30.0/subjects/Order/801D0000000FpNEIA0"
        },
        "Id" : "801D0000000FpNEIA0",
        "OrderItems" : {
          "totalSize" : 3,
          "done" : true,
          "records" : [ {
```

```

    "attributes" : {
      "type" : "OrderItem",
      "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CIWBIA4"
    },
    "Id" : "802D0000000CIWBIA4"
  }, {
    "attributes" : {
      "type" : "OrderItem",
      "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CIWCIA4"
    },
    "Id" : "802D0000000CIWCIA4"
  }, {
    "attributes" : {
      "type" : "OrderItem",
      "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CIWDIA4"
    },
    "Id" : "802D0000000CIWDIA4"
  } ]
},
"Custom_Objects__r" : null
}, {
  "attributes" : {
    "type" : "Order",
    "url" : "/services/data/v30.0/subjects/Order/801D0000000FqztIAC"
  },
  "Id" : "801D0000000FqztIAC",
  "OrderItems" : null,
  "Custom_Objects__r" : null
}, {
  "attributes" : {
    "type" : "Order",
    "url" : "/services/data/v30.0/subjects/Order/801D0000000FpkNIAS"
  },
  "Id" : "801D0000000FpkNIAS",
  "OrderItems" : null,
  "Custom_Objects__r" : null
} ]
},
"Test_Contract1__r" : null
} ]
}

```

## Get Details About an Order

Here's an example of a GET request using the [Order](#) resource to query details about an order and its order products and custom object records.

### Example usage

```
/services/data/v30.0/commerce/sale/order/801D0000000FzsM
```

### Example request body

None

**Example JSON response body**

```
{
  "totalSize" : 1,
  "done" : true,
  "records" : [ {
    "attributes" : {
      "type" : "Order",
      "url" : "/services/data/v30.0/subjects/Order/801D0000000FzsMIAS"
    },
    "Id" : "801D0000000FzsMIAS",
    "OrderItems" : {
      "totalSize" : 2,
      "done" : true,
      "records" : [ {
        "attributes" : {
          "type" : "OrderItem",
          "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CKIHIA4"
        },
        "Id" : "802D0000000CKIHIA4"
      }, {
        "attributes" : {
          "type" : "OrderItem",
          "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CKIGIA4"
        },
        "Id" : "802D0000000CKIGIA4"
      } ]
    },
    "Custom_Objects__r" : null
  } ]
}
```

## Filter Details About a Contract

Here's an example of a GET request using the [Contract-based Orders](#) resource to query a given contract's activated orders.

**Example usage**

```
/services/data/v30.0/commerce/sale/800D0000000PFL8?contract.orders.StatusCode='Activated'
```

**Example request body**

None

**Example JSON response body**

```
{
  "totalSize" : 1,
  "done" : true,
  "records" : [ {
    "attributes" : {
      "type" : "Contract",
      "url" : "/services/data/v30.0/subjects/Contract/800D0000000PFHpIAO"
    },
    "Id" : "800D0000000PFHpIAO",
    "Orders" : {
```

```

    "totalSize" : 1,
    "done" : true,
    "records" : [ {
      "attributes" : {
        "type" : "Order",
        "url" : "/services/data/v30.0/subjects/Order/801D0000000FpNEIA0"
      },
      "Id" : "801D0000000FpNEIA0",
      "OrderItems" : {
        "totalSize" : 3,
        "done" : true,
        "records" : [ {
          "attributes" : {
            "type" : "OrderItem",
            "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CIWBIA4"
          },
          "Id" : "802D0000000CIWBIA4"
        }, {
          "attributes" : {
            "type" : "OrderItem",
            "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CIWCIA4"
          },
          "Id" : "802D0000000CIWCIA4"
        }, {
          "attributes" : {
            "type" : "OrderItem",
            "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CIWDIA4"
          },
          "Id" : "802D0000000CIWDIA4"
        } ]
      },
      "Custom_Objects__r" : null
    } ]
  },
  "Test_Contract1__r" : null
} ]
}

```

## Filter Details About an Order

Here's an example of a GET request using the [Order](#) resource to query details for order products with a certain start date for a given order.

### Example usage

```
/services/data/v30.0/commerce/sale/order/801D0000000FzsM?order.orderItems.effectiveDate=2013-08-05
```

### Example request body

None

### Example JSON response body

```

{
  "totalSize" : 1,
  "done" : true,
  "records" : [ {

```



```

    "attributes" : {
      "type" : "Order",
      "url" : "/services/data/v30.0/subjects/Order/801D0000000FzsMIAS"
    },
    "Id" : "801D0000000FzsMIAS",
    "OrderItems" : {
      "totalSize" : 1,
      "done" : true,
      "records" : [ {
        "attributes" : {
          "type" : "OrderItem",
          "url" : "/services/data/v30.0/subjects/OrderItem/802D0000000CKIHIA4"
        },
        "Id" : "802D0000000CKIHIA4"
      } ]
    },
    "Custom_Objects__r" : null
  } ]
}

```

## Place Order REST API Resource Reference

Each Place Order REST API resource is a URI used with an HTTP method (such as GET).

Resources for the Place Order REST API are:

Resource	Supported HTTP Method	Description
/services/data/ <b>latest API version</b> /commerce/sale	POST	Add new orders, order products, and custom objects to a new contract.
/services/data/ <b>latest API version</b> /commerce/sale/ <b>contract ID</b>	PATCH, GET	Add new orders, order products, and custom objects to an existing contract. Retrieve a contract's child orders, order products, and custom objects.
/services/data/ <b>latest API version</b> /commerce/sale/order	POST	Add new order products and custom objects to a new order.
/services/data/ <b>latest API version</b> /commerce/sale/order/ <b>order ID</b>	PATCH, GET	Add new order products and custom objects to an existing order. Retrieve an order's child order products and custom objects.

## Create Contract-based Orders

With this resource, you can create a new contract with orders and order products, as well as custom object records on the contract or order level.

## Syntax

### URI

```
/services/data/latest API version/commerce/sale
```

### Available since release

30.0

### Formats

JSON

### HTTP methods

POST

### Request body

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the new contract.	30.0
AccountId	String	Unique record identifier for the parent account.	30.0
Status	String	Status of the contract.	30.0
Orders	<a href="#">Orders[]</a>	Child orders of the new contract.	30.0
<i>CustomObject__r</i>	<a href="#">Custom Objects[]</a>	Child custom object records of the new contract.	30.0

### Attributes

Property	Type	Description	Since Version
type	String	Format of the resource.	30.0

### Orders

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the new order record.	30.0
Status	String	Status of the order.	30.0
Pricebook2Id	String	Unique record identifier for the associated price book.	30.0
OrderItems	<a href="#">Order Products[]</a>	Child order products of the new order.	30.0
<i>CustomObject__r</i>	<a href="#">Custom Objects[]</a>	Child custom object records of the new order.	30.0

**Custom Object Records**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the custom object record.	30.0
Id	String	Unique record identifier.	30.0

**Order Products**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the order product record.	30.0
PricebookEntryId	String	Unique record identifier for the associated price book entry.	30.0
quantity	Number	Number of units of the order product.	30.0
UnitPrice	Currency	The unit price for the order product.	30.0

**Request parameters**

None

**Response body**

Property	Type	Description
totalSize	Number	Total number of records added.
done	Boolean	When <code>true</code> , the operation was successful.
records	<a href="#">Records</a> []	Attributes and IDs of records.

**Records**

Property	Type	Description
attributes	<a href="#">Attributes</a>	Type and URL for the new record.
Id	String	Unique identifier for the new record.

**Attributes**

Property	Type	Description
type	String	Format of the resource.
url	String	Resource URL.

## Example

See [Add a Contract and Orders to an Existing Account](#) on page 17.

## Usage

You can only create one contract at a time. You can't POST new orders onto an existing contract. For that, see the [Contract-based Orders](#) resource.

## Contract-based Orders

With this resource, you can add one or more new orders to an existing contract, as well as order products and custom object records for each order, or you can retrieve data for a specific contract.

If available, GET method retrieves the contract's child orders and order products, as well as custom objects under the contract and orders.

## Syntax

### URI

```
/services/data/latest API version/commerce/sale/contractId
```

For retrieving filtered data:

```
/services/data/latest API version/commerce/sale/contract ID?contract.orders.fieldname=value
```

### Available since release

30.0

### Formats

JSON

### HTTP methods

PATCH, GET

### Request parameters

- You can use parameters for all standard and custom fields on contracts, orders, order products, and any custom objects directly related to these objects.
- The parameters must be fully qualified. For example: **objectname.relationshipname.fieldname=value**.
  - Object name must be all lower-case.
  - Relationship name must match what is defined on the object and is case-sensitive.
  - Field name isn't case sensitive.
  - If the value is a string, it must be encased in single quotation marks. If the value is a number, it must not be encased. If the value is a date, it should be in the YYYY-MM-DD format.
- You can use multiple parameter fields, separated by "&", to make more detailed filters. For example:

```
/services/data/v30.0/commerce/sale/{contractId?contract.status='Activated'  
&contract.Orders.status='Draft'&contract.Orders.OrderItems.unitprice=300
```

The following aren't supported:

- Arrays of values. For example: `contract.orders.Status='Activated','Draft'`.
- Operators: `>`, `>=`, `<` and `<=`
- The OR condition

Parameters	Description	Since Version
<code>contract</code>	The object name of the record being filtered. In this resource, this is always <code>contract</code> .	30.0
<code>orders</code>	The relationship name of the field that the order's data will be filtered by. In this resource, this is always <code>orders</code> .	30.0
field name	The field whose value to filter by. For example, if you want to only retrieve orders with a status category, the field name is <code>StatusCode</code> .	30.0
value	The value to filter by. For example, if you want to only retrieve orders with a status category of <code>Activated</code> , the value is <code>Activated</code> .	30.0

### Request body

Property	Type	Description	Since Version
<code>attributes</code>	<a href="#">Attributes</a>	Type of the contract.	30.0
<code>Id</code>	String	Unique contract identifier.	30.0
<code>Orders</code>	<a href="#">Orders[]</a>	Child orders of the contract.	30.0

### Attributes

Property	Type	Description	Since Version
<code>type</code>	String	Format of the resource.	30.0

### Order Records

Property	Type	Description	Since Version
<code>attributes</code>	<a href="#">Attributes</a>	Type of the order record.	30.0
<code>Status</code>	String	Status of the order.	30.0
<code>contractId</code>	String	Unique record identifier for the parent contract.	30.0
<code>pricebook2Id</code>	String	Unique record identifier for the associated price book.	30.0
<code>OrderItems</code>	<a href="#">OrderProducts[]</a>	Child order products of the order.	30.0

Property	Type	Description	Since Version
<i>CustomObject__r</i>	<a href="#">Custom Objects[]</a>	Child custom object records of the order.	30.0

**Order Products**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the order product.	30.0
PricebookEntryId	String	Unique record identifier for the associated price book entry.	30.0
quantity	Number	Number of units of the order product.	30.0
UnitPrice	Currency	The unit price for the order product.	30.0

**Custom Objects**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the custom object.	30.0
Id	String	Unique custom object identifier.	30.0

**Response body**

Property	Type	Description
totalSize	Number	Total number of records retrieved.
done	Boolean	When <code>true</code> , the operation was successful.
records	<a href="#">Records[]</a>	Attributes and ID of contract record.

**Records**

Property	Type	Description
attributes	<a href="#">Attributes</a>	Type and URL of the record.
Id	String	Unique contract identifier.

**Attributes**

Property	Type	Description
type	String	Format of the resource.
url	String	Resource URL.

## Examples

- [Add Orders to an Existing Contract](#) on page 20
- [Get Details About a Contract](#) on page 24
- [Filter Details About a Contract](#) on page 27

## Create Order

With this resource, you can create a new order with order products and custom objects.

If you don't want to add the order to a contract, you can add it directly to an account. You can only create one new order per call. The request body must have either an account or a contract as its parent record, and it must have a reference to a price book.

## Syntax

**URI**

```
/services/data/latest API version/commerce/sale/order
```

**Available since release**

30.0

**Formats**

JSON

**HTTP methods**

POST

**Request body**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the new order.	30.0
OrderItems	<a href="#">Order Products</a> []	Child order products of the new order.	30.0
<i>CustomObject__r</i>	<a href="#">Custom Objects</a> []	Child custom object records of the new order.	30.0

**Attributes**

Property	Type	Description	Since Version
type	String	Format of the resource.	30.0
url	String	Resource URL.	30.0

**Order Products**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the order product.	30.0
PricebookEntryId	String	Unique record identifier for the associated price book entry.	30.0
quantity	Number	Number of units of the order product.	30.0
UnitPrice	Currency	The unit price for the order product.	30.0
Id	String	Unique order product identifier.	30.0

**Custom Objects**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the custom object.	30.0
Id	String	Unique custom object identifier.	30.0

**Request parameters**

None

**Response body**

Property	Type	Description
totalSize	Number	Total number of records retrieved.
done	Boolean	When <code>true</code> , the operation was successful.
records	<a href="#">Records[]</a>	Attributes and ID of contract record.



### Records

Property	Type	Description
attributes	<a href="#">Attributes</a> on page 37	Type and URL of the record.
Id	String	Unique contract identifier.

### Attributes

Property	Type	Description
type	String	Format of the resource.
url	String	Resource URL.

## Example

See [Add an Order to an Existing Account](#) on page 19.

## Order

Use this resource to add one or more new order products and custom object records to an existing order or to retrieve data for a specific order.

You can only PATCH one order at a time.

If available, GET method retrieves the orders' child order products and custom objects under the order or order products.

## Syntax

### URI

```
/services/data/latest API version/commerce/sale/order/order ID
```

For retrieving filtered data:

```
/services/data/latest API version/commerce/sale/order/order ID?order.orderItems.field name=value
```

### Available since release

30.0

### Formats

JSON

### HTTP methods

POST

**Request body**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the order.	30.0
OrderItems	<a href="#">Order Products</a>	Child order products of the new order.	30.0
<i>CustomObject__r</i>	<a href="#">Custom Object</a>	Child custom object records of the new order.	30.0

**Attributes**

Property	Type	Description	Since Version
type	String	Format of the resource.	30.0

**Order Products**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the order product.	30.0
PricebookEntryId	String	Unique record identifier for the associated price book entry.	30.0
quantity	Number	Number of units of the order product.	30.0
UnitPrice	Currency	The unit price for the order product.	30.0
orderId	String	Unique record identifier for the parent order.	30.0
Id	String	Unique order product identifier.	30.0

**Custom Objects**

Property	Type	Description	Since Version
attributes	<a href="#">Attributes</a>	Type and URL of the custom object.	30.0
Id	String	Unique custom object identifier.	30.0

**Request parameters**

- You can use parameters for all standard and custom fields on contracts, orders, order products, and any custom objects directly related to these objects.
- The parameters must be fully qualified. For example: ***objectname.relationshipname.fieldname=value***.

- Object name must be all lower-case.
- Relationship name must match what is defined on the object and is case-sensitive.
- Field name isn't case sensitive.
- If the value is a string, it must be encased in single quotation marks. If the value is a number, it must not be encased. If the value is a date, it should be in the YYYY-MM-DD format.
- You can use multiple parameter fields, separated by "&", to make more detailed filters. For example:

```
/services/data/v30.0/commerce/sale/{contractId}?contract.status='Activated'
&contract.Orders.status='Draft'&contract.Orders.OrderItems.unitprice=300
```

The following aren't supported.

- Arrays of values. For example: `order.orderItems.effectiveDate=2013-01-01,2013-01-02`.
- Operators: `>`, `>=`, `<` and `<=`
- The OR condition

Parameters	Description
<code>order</code>	The object name of the record being filtered. In this resource, this is always <code>order</code> .
<code>orderItems</code>	The relationship name of the field that the order's data will be filtered by. In this resource, this is always <code>orderItems</code> .
field name	The field whose value to filter by. For example, if you want to only retrieve order products with a certain start date, the field name is <code>effectivedate</code> .
value	The value to filter by. For example, if you want to only retrieve order products that started on January 1, 2013, the value is <code>2013-01-01</code> .

## Response body

Property	Type	Description
<code>totalSize</code>	Number	Total number of records listed.
<code>records</code>	<a href="#">Records[]</a>	Attributes and IDs of the new records.

## Records

Property	Type	Description
<code>attributes</code>	<a href="#">Attributes</a>	Type and URL for the record.
<code>Id</code>	String	Unique record identifier.

**Attributes**

Property	Type	Description
type	String	Format of the resource.
url	URI	Resource URL.

**Examples**

- [Add Order Products to an Existing Order](#) on page 23
- [Get Details About an Order](#) on page 26
- [Filter Details About an Order](#) on page 28

# INDEX

## A

- Add order products
  - on existing order [23](#)
- Authentication
  - Additional resources [16](#)
  - OAuth [2–4](#), [9](#), [12](#), [15](#)
  - OAuth endpoints [3](#)
  - Remote access applications [3](#)

## G

- Get contract data
  - filtered [27](#)
  - unfiltered [24](#)
- Get order data
  - filtered [28](#)
  - unfiltered [26](#)

## I

- Introduction [1](#)

## L

- Limitations [1](#)
- Limits, general [1](#)

## O

- OAuth
  - Additional resources [16](#)
  - Refresh token [15](#)
  - User-agent OAuth flow [9](#)
  - Username-password OAuth flow [12](#)
  - Web server OAuth flow [4](#)

## P

- Place order with order products
  - on existing account [19](#)
  - on existing contract [20](#)
  - on new contract [17](#)
  - standalone [19](#)

## R

- Reference [29](#)
- Requirements [1](#)
- Resource
  - GET data for an order [37](#)
  - GET filtered data for an order [37](#)
  - PATCH custom object, on existing order [37](#)
  - PATCH custom object, on new order product [37](#)
  - PATCH existing order [37](#)
  - PATCH order products, on existing order [37](#)
- Resources
  - examples [17](#)
  - GET data for a contract [32](#)
  - GET filtered data for a contract [32](#)
  - PATCH custom object, on existing contract [32](#)
  - PATCH custom object, on new order [32](#)
  - PATCH existing contract [32](#)
  - PATCH orders and order products, on existing contract [32](#)
  - POST an order [35](#)
  - POST contract [29](#)
  - POST custom object records, on contract-based orders [29](#)
  - POST custom object records, on new contract [29](#)
  - POST custom object records, on new order [35](#)
  - POST custom object records, on new order products [35](#)
  - POST order products, on new contract-based orders [29](#)
  - POST order products, on new order [35](#)
  - POST orders, on new contract [29](#)