SFDC Rest APIs

There are two types of Rest APIs in Salesforce

1. Standard Rest APIs
2. Custom Rest APIs

Both these APIs support both standard as well as custom objects so don't confuse with their names.

Q: How to create a standard Rest API

Step 1: create connected app in Salesforce

Go to setup and quick search apps. Under connected apps create new app. Make sure you do **Enable OAuth Settings** and provide relevant access. Save and note down client id and secret.

For calling an API you need bearer token. Your client application will generate the bearer by using following details. You can also use Postman to generate the same.

Auth URL https://login.salesforce.com/services/oauth2/token

Post this data to above URL: grant_type=password&client_id=*client_id*&client_secret=*secret*&username=*your salesforce username*&password=*your salesforce pwd followed by password security token*

Above URL will return bearer token which you can use in your client application or in Postman.

Step 2 Create API

2.1 Creating and using standard Rest API

You really don't need to do anything to create a standard API. It is automatically available in Salesforce for a given object. E.g. following is the URL for custom object Employee__c

https://anil99-dev-ed.my.salesforce.com/services/data/v45.0/composite/tree/Employee__c

Note: make sure you use your own domain name and object name in URL above.

Request JSON format for inserting records (request type: post):

Insert single record	**Insert multiple records**

{
"Name" : "Anil Sharma",
"Other_Field__c" : "other field value",
}
	{
	  "records": [{
	    "Name": "Anil Sharma ",
	    "Other_Field__c": "Other field value ",
	
	    "attributes": {
	      "type": "Employee__c",
	      "referenceId": "ref1"
	    }
	  }, {
	    "Name": "Krishna Gupta",
	    "Other_Field__c": "Other field value ",
	    "attributes": {
	      "type": "Employee__c ",
	      "referenceId": "ref2"
	    }
	  }]
	}

2.2 Creating and using custom Rest API

Custom rest API is an Apex class with Rest annotation e.g.

```
@RestResource(urlMapping='/MyAPI/*')
global with sharing class MyRestResource {

  @HttpDelete
  global static void doDelete() {
    RestRequest req = RestContext.request;
    RestResponse res = RestContext.response;
    String accountId = req.requestURI.substring(req.requestURI.lastIndexOf('/')+1);
    Account account = [SELECT Id FROM Account WHERE Id = :accountId];
    delete account;
  }

  @HttpGet
  global static Account doGet() {
    RestRequest req = RestContext.request;
    RestResponse res = RestContext.response;
    String accountId = req.requestURI.substring(req.requestURI.lastIndexOf('/')+1);
    Account result = [SELECT Id, Name, Phone, Website FROM Account WHERE Id = :accountId];
    return result;
  }

 @HttpPost
  global static String doPost(String name,  String phone, String website) {
    Account account = new Account();
    account.Name = name;
    account.phone = phone;
    account.website = website;
    insert account;
    return account.Id;
  }
}
```

Sample JSONs for above three requests:

GET
URL https://<your domain>.salesforce.com/services/apexrest/MyAPI/accountID replace account id with actual id

POST
URL https://<your domain>.salesforce.com/services/apexrest/MyAPI Request JSON {“name” : “anil sharma”,“phone” : “1234567890”,“website” : “www.example.com”}

DELETE
URL https://<your domain>.salesforce.com/services/apexrest/MyAPI/accountID replace account id with actual id

Note: standard API for inserting multiple records allow maximum 200 records at a time. To insert more than 200 records at a time you need to write a custom API as explained below:

Bulk APIs

To operate on large number of records you need a bulk API. Steps to create bulk API:

Step 1: prepare your XML

Post xml below to your instance url. Use your bearer token for authentication

Request XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<jobInfo xmlns="http://www.force.com/2009/06/asyncapi/dataload">
                <operation>insert</operation>
                <object>Account</object>
                <contentType>XML</contentType>
</jobInfo>
```

Response XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<jobInfo
  xmlns="http://www.force.com/2009/06/asyncapi/dataload">
   <id>7507F000008Q3hRQAS</id>
   <operation>insert</operation>
   <object>Account</object>
   <createdById>0057F000000IWSkQAO</createdById>
   <createdDate>2018-11-13T07:42:49.000Z</createdDate>
   <systemModstamp>2018-11-13T07:42:49.000Z</systemModstamp>
   <state>Open</state>
   <concurrencyMode>Parallel</concurrencyMode>
   <contentType>XML</contentType>
   <numberBatchesQueued>0</numberBatchesQueued>
   <numberBatchesInProgress>0</numberBatchesInProgress>
   <numberBatchesCompleted>0</numberBatchesCompleted>
   <numberBatchesFailed>0</numberBatchesFailed>
   <numberBatchesTotal>0</numberBatchesTotal>
   <numberRecordsProcessed>0</numberRecordsProcessed>
   <numberRetries>0</numberRetries>
   <apiVersion>44.0</apiVersion>
   <numberRecordsFailed>0</numberRecordsFailed>
   <totalProcessingTime>0</totalProcessingTime>
   <apiActiveProcessingTime>0</apiActiveProcessingTime>
   <apexProcessingTime>0</apexProcessingTime>
</jobInfo>
```

Step 2 Add a batch to above job

Post to URL: https:// **instanceurl** /services/async/**API_Version**/job /**Job_Id**/batch

Request XML:

```
<sObjects xmlns="http://www.force.com/2009/06/asyncapi/dataload">
        <sObject>
                <type>Account</type>
                <Name>Dummy113</Name>
                <SAP_Customer_Number__c>Dummy113</SAP_Customer_Number__c>
                <Email__c>test@abc.com</Email__c>
        </sObject>
        <sObject>
                <type>Account</type>
                <Name>Dummy114</Name>
                <SAP_Customer_Number__c>Dummy114</SAP_Customer_Number__c>
                <Email__c>test@abc.com</Email__c>
        </sObject>
</sObjects>
```

Response XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<batchInfo   xmlns="http://www.force.com/2009/06/asyncapi/dataload">
  <id>7517F00000BUPNOQA5</id>
  <jobId>7507F000008Q3hRQAS</jobId>
  <state>Queued</state>
  <createdDate>2018-11-13T12:51:10.000Z</createdDate>
  <systemModstamp>2018-11-13T12:51:10.000Z</systemModstamp>
  <numberRecordsProcessed>0</numberRecordsProcessed>
  <numberRecordsFailed>0</numberRecordsFailed>
  <totalProcessingTime>0</totalProcessingTime>
  <apiActiveProcessingTime>0</apiActiveProcessingTime>
  <apexProcessingTime>0</apexProcessingTime>
</batchInfo>
```

Request XML to close the job:

```
<?xml version="1.0" encoding="UTF-8"?>
<jobInfo   xmlns="http://www.force.com/2009/06/asyncapi/dataload">
 <state>Closed</state>
</jobInfo>
```

Steps to get the batch result

GET https:// **instanceurl** /services/async/**API_Version**/job /**Job_Id**/batch/**Batch_Id**/result

Limitations of Bulk API

Following are the limitations of Salesforce Bulk APIs.

- Batches for data loads can consist of a single CSV, XML, or JSON file that is no larger than 10 MB.
- A batch can contain a maximum of 10,000 records.
- A batch can contain a maximum of 10,000,000 characters for all the data in a batch.
- A field can contain a maximum of 32,000 characters.
- A record can contain a maximum of 5,000 fields.
- A record can contain a maximum of 400,000 characters for all its fields.
- A batch must contain some content or an error occurs.

Q: How to refresh token programmatically

Access token is valid for the time based on your profile/org or connected app setiings. Default expiration time is two hours which can be increased upto 24 hours.

Your client application need to refresh access token when expired. In an API call if you get access token expired error you can refresh by following POST request:

Auth URL https://login.salesforce.com/services/oauth2/token

Post this data to above URL: grant_type=password&client_id=*client_id*&client_secret=*secret*&username=*your salesforce username*&password=*your salesforce pwd followed by password security token*