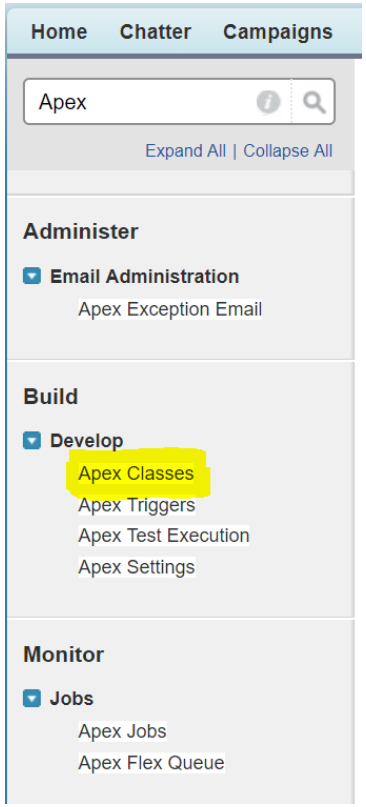
Sample Apex Class getCallerInformation

Implementation

Classic

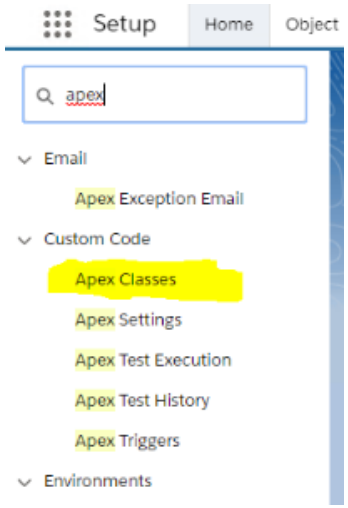
To create a new Apex class, access Setup in SalesForce, and type in "Apex" in the quick search box.

Select "Build ... Develop ... Apex Classes".



Lightning

Access Setup, and search for "Apex" in the search box.



Classic and Lightning

Create a new class. Add the following code:

Sample Apex getCallerInformation

```
@RestResource(urlMapping='/jtelACD/getCallerInformation')
global with sharing class JTELACD_GetCallerInformation {
    @HttpGet
    global static void doGet() {
        RestRequest req = RestContext.request;
        RestResponse res = RestContext.response;
        res.addHeader('Content-Type', 'application/json');

        // This is how to access URLParams if required
        // List<String> URLParams = req.requestURI.split('/');
        // Example: ["","jtelACD","getCallerInformation"]

        // Extract the request parameters
        String CallerID = req.params.get('CallerID');
        String ServiceNumber = req.params.get('ServiceNumber');
        String ServiceName = req.params.get('ServiceName');
        // From release 3.25 - 2021-07-02 and upwards
        String Function = req.params.get('Function');
        String UserData = req.params.get('UserData');
        String UserData2 = req.params.get('UserData2');
        String UserData3 = req.params.get('UserData3');
```

```
// Specifying a Function is intended to allow the developer to perform different types of query.
// As an example, we just return the data back if a function is specified.
// In a real world case, a query for example to search for and return a ticket, contract or some other record would be performed.
if( Function != null && Function != '' ) {
    JSONGenerator gen = JSON.createGenerator( true );
    gen.writeStartObject();
    gen.writeStringField( 'CallerID', CallerID );
    gen.writeStringField( 'ServiceNumber', ServiceNumber );
    gen.writeStringField( 'ServiceName', ServiceName );
    gen.writeStringField( 'Function', Function );
    gen.writeStringField( 'UserData', UserData );
    gen.writeStringField( 'UserData2', UserData2 );
    gen.writeStringField( 'UserData3', UserData3 );
    gen.writeEndObject();

    res.responseBody = Blob.valueOf(gen.getAsString());
    return;
}

// Perform SOSL Query over contacts and leads, searching for the phone number with *wildcards*
String soslSearchString = '*' + CallerID + '*';
List<List<SObject>> searchList = [FIND :soslSearchString IN ALL FIELDS
                                  RETURNING
                                  Contact,
                                  Lead];

List<Contact> contacts = ( (List<Contact>) searchList[0] );
List<Lead> leads       = ( (List<Lead>) searchList[1] );

// Multple hits?
if( contacts.size() + leads.size() > 1 )
{
    // Multiple hits
    res.statusCode = 300;
    return;
}

if( contacts.size() > 0 )
{
    // Query for exact contact
    String Id = (String) contacts.get(0).get('Id');
    Set<String> fieldSet = schema.describeSObjects(new String[] { 'Contact' } )[0].fields.getMap().keyset();
    List<String> fieldList = new List<String>( fieldSet );
    String query = 'SELECT ' + String.join( fieldList, ',' ) + ' FROM Contact WHERE Id = \'' + Id + '\'';
    Contact contact = Database.query( query );
    res.statusCode = 200;
    res.responseBody = Blob.valueOf(JSON.serialize(contact));
    return;
}

if( leads.size() > 0 )
```

```
        {
            // Query for exact lead
            String Id = (String) leads.get(0).get('Id');
            Set<String> fieldSet = schema.describeSObjects(new String[] { 'Lead' } )[0].fields.getMap().keyset();
            List<String> fieldList = new List<String>( fieldSet );
            String query = 'SELECT ' + String.join( fieldList, ',' ) + ' FROM Lead WHERE Id = \'' + Id + '\'';
            Lead lead = Database.query( query );
            res.statusCode = 200;
            res.responseBody = Blob.valueOf(JSON.serialize(lead));
            return;
        }

        // Nothing found
        res.statusCode = 404;
        return;
    }
}
```

CURL Test

First of all, obtain an OAUTH Token, see Testing with CURL.

The following CURL command can be used to test this API:

CURL - getCallerInformation Test

```
curl.exe --silent -i -X GET --header "Authorization: Bearer <OAUTH_TOKEN>" --header "Connection: Close" "https://<SALES_FORCE_INSTANCE_URL>/services/apexrest
/jtelACD/getCallerInformation?CallerID=4989461495000&ServiceNumber=4980012345678&ServiceName=test"
```

The following CURL command can be used to test this API with the additional parameters from Release 3.25 - 2021-07-02 and upwards:

CURL - getCallerInformation Test

```
curl.exe --silent -i -X GET --header "Authorization: Bearer <OAUTH_TOKEN>" --header "Connection: Close" "https://<SALES_FORCE_INSTANCE_URL>/services/apexrest
/jtelACD/getCallerInformation?
CallerID=4989461495000&ServiceNumber=4980012345678&ServiceName=test&Function=testFunction&UserData=testUserData&UserData2=testUserData2&UserData3=testUserData3"
```