Use Web Service Data in Web Resources (REST and SOAP Endpoint)

You can use JScript and Silverlight Web resources to access Microsoft Dynamics CRM 2011 and Microsoft Dynamics CRM Online data from within the application. There are two web services available, each provides specific strengths. The following table describes the appropriate web service to use depending on the task you need to perform.

Task	Web Service
Create, Retrieve, Update and Delete records.	REST Endpoint
Associate and Disassociate records	REST Endpoint
Assign Records	SOAP Endpoint
Retrieve Metadata	SOAP Endpoint
Execute Messages	SOAP Endpoint

Both of these Web services rely on the authentication provided by the Microsoft Dynamics CRM application. They cannot be used by code that executes outside the context of the application. They are effectively limited to use within Silverlight, JScript libraries, or JScript included in Web Page (HTML) Web resources.

The REST endpoint provides a 'RESTful' web service using OData to provide a programming environment that is familiar to many developers. It is the recommended web service to use for tasks that involve creating, retrieving, updating and deleting records. However, in this release of Microsoft Dynamics CRM the capabilities of this Web service are limited to these actions. Future versions of Microsoft Dynamics CRM will enhance the capabilities of the REST endpoint.

The SOAP endpoint provides access to all the messages defined in the Organization service. However, only the types defined within the WSDL will be returned. There is no strong type support. While the SOAP endpoint is also capable of performing create, retrieve, update and delete operations, the REST endpoint provides a better developer experience. In this release of Microsoft Dynamics CRM the SOAP endpoint provides an alternative way to perform operations that the REST endpoint is not yet capable of.

Use the REST Endpoint for Web Resources

The REST endpoint for web resources provides an alternative interface to work with Microsoft Dynamics CRM 2011 and Microsoft Dynamics CRM Online data. You can use the

REST endpoint to execute HTTP requests by using a service that is based on a Uniform Resource Identifier (URI). The REST endpoint is only available for use by JScript and Silverlight web resources.

What Is REST?

REST represents Representational State Transfer. REST is an architectural style in which every resource is addressed by using a unique URI. In Microsoft Dynamics CRM, a resource can be an entity collection or a record.

REST works the way the Internet works. You interact with resources by using HTTP verbs such as **GET**, **POST**, **MERGE**, and **DELETE**. Various libraries can be used to process the HTTP requests and responses. REST provides a standard interface that you can use with any programming language. REST allows for either synchronous or asynchronous processing of operations. The capability to perform asynchronous operations makes REST well suited for AJAX and Silverlight clients.

Microsoft Dynamics CRM Implementation of REST

Microsoft Dynamics CRM 2011 uses the Windows Communication Foundation (WCF) Data Services framework to provide an Open Data Protocol (OData) endpoint that is a REST-based data service. This endpoint is called the **Organization Data Service**. In Microsoft Dynamics CRM, the service root URI is:

Copy

[Your Organization Root URL]/xrmservices/2011/organizationdata.svc

OData sends and receives data by using either ATOM or JavaScript Object Notation (JSON). ATOM is an XML-based format usually used for RSS feeds. JSON is a text format that allows for serialization of JavaScript objects.

To provide a consistent set of URIs that corresponds to the entities used in Microsoft Dynamics CRM, an Entity Data Model (EDM) organizes the data in the form of records of "entity types" and the associations between them.

OData Entity Data Model (EDM)

The Microsoft Dynamics CRM EDM is described in an OData Service Metadata document available at the following path:

Copy

[Your Organization Root URL]/xrmservices/2011/organizationdata.svc/$metadata

This XML document uses conceptual schema definition language (CSDL) to describe the available data. You will download this document and use it to generate typed classes when you use managed code with Silverlight or as a reference for available objects when you use JScript.

Limitations

The REST endpoint provides an alternative to the WCF SOAP endpoint, but there are currently some limitations.

Only Create, Retrieve, Update, and Delete actions can be performed on entity records.

- Messages that require the **Execute** method cannot be performed.
- Associate and disassociate actions can be performed by using navigation properties.

Authentication is only possible within the application.

Use of the REST endpoint is limited to JScript libraries or Silverlight web resources.

The OData protocol is not fully implemented. Some system query options are not available.

You cannot use late binding with managed code with Silverlight.

You will typically use WCF Data Services Client Data Service classes while programming by using managed code for the REST endpoint with Silverlight. These classes allow for early binding so that you get strongly typed classes at design time. The only entities available to you are those defined in the system when the classes were generated. This means that you cannot use late binding to work with custom entities that were not included in the WCF Data Services Client Data Service classes when they were generated.

Use the SOAP Endpoint for Web Resources

Unlike the REST endpoint for web resources, the SOAP endpoint uses the Organization service. This is the same service used when writing applications that exist outside of the Microsoft Dynamics CRM 2011 and Microsoft Dynamics CRM Online application. The differences are:

- Requests are sent to a different URL: <organization URL>/XRMServices/2011/Organization.svc/web.
- Authentication is provided by the application.

Because authentication is provided by the application, you must create web resources in the application before the code that uses the service can operate.

Comparison of Programming Methods

You can use the SOAP endpoint for Web Resources with JScript libraries or by using Microsoft Silverlight. The process for using this endpoint is very different, depending on the technology used.

Using the SOAP Endpoint with Silverlight

With Silverlight, you create a Silverlight 4 application project in Visual Studio. After the project is created, you must:

- Add a service reference to the Organization service.
- Add some additional files to your solution and manually modify the Reference.cs file generated when you add the service reference.
- Write code using asynchronous methods. The proxy created by Silverlight when you add the services reference only supports asynchronous methods.
- Use the late binding syntax because strong types are not available.

Using the SOAP Endpoint with JScript

With JScript, you will be using **XmlHttpRequest** to **POST** requests to the service. The body of the request must contain the XML appropriate for the message you are using. You must also parse the XML returned in a response. With **XmlHttpRequest**, it is possible to make synchronous requests. However it is highly recommended to always use asynchronous requests. Because manually configuring each request is very time consuming, it is expected that you will reuse existing libraries or create your own. Microsoft Dynamics CRM 2011 does not provide a comprehensive set of JScript libraries. The specific syntax used when calling JScript libraries depends on how they are designed. Several sample JScript libraries are provided with the Microsoft Dynamics CRM SDK, but these are not intended to represent the only or best library design. The content in the Microsoft Dynamics CRM SDK focuses on helping you create your own libraries.

OData and Jscript in CRM 2011

OData is also referred as Open Data Protocol. CRM 2011 mainly uses Windows Communication Foundation (WCF) data services framework to provide oData Endpoint which is nothing but a REST based data service. The address for the endpoint is:

http://{OrganizationRootUrl} /XRMServices/2011/OrganizationData.svc

OData uses some data format for sending and receiving the data. Basically it uses the following two formats.

1. ATOM: It is an Xml based format mainly used for the RSS feeds.
2. JSON: JavaScript Object Notation is a text formats which makes very easy for the developers to understand the response what we get.

We would be using the JSON format in our example.

http://{OrganizationRootUrl} /XRMServices/2011/OrganizationData.svc

Examples:
1) For retrieving any Entity Set (here GoalSet)
http://{OrganizationRootUrl} /XRMServices/2011/OrganizationData.svc/GoalSet

2) For retrieving limited attributes use $select query option
http://{OrganizationRootUrl}/XRMServices/2011/OrganizationData.svc/GoalSet
?$select=IsAmount,MetricId
This will retrieve collection of IsAmount and MetricId for all Goals.

3) For retrieving data filtered on some criteria use $filter query option
http://{OrganizationRootUrl}/XRMServices/2011/OrganizationData.svc/GoalSet
?$select=IsAmount,MetricId &$filter=GoalId eq guid'" + goalid + "'";
This will retrieve IsAmount and MetricId for the Goal whose GoalId matches with the value of given goalid.

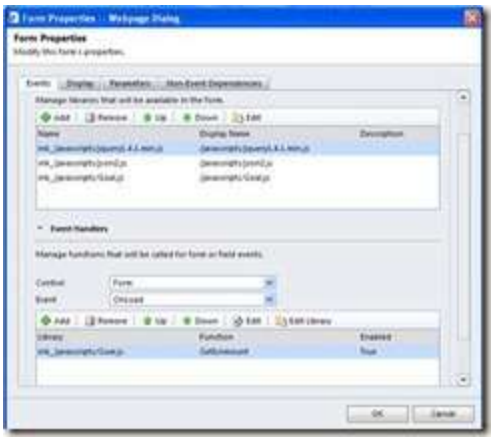
4) For retrieving data from related entity use $expand query option
http://{OrganizationRootUrl}/XRMServices/2011/OrganizationData.svc/GoalSet
?$select=IsAmount,metric_goal/Name,metric_goal/OrganizationId&$expand=metric_goal&
$filter=GoalId eq guid'" + goalid + "'";

- Here 'metric_goal' is the relationship property which defines realtionship between Goal and GoalMetric.

  This retrieves IsAmount from GoalSet .Name and Organizationid are fetched from the related GoalMetricSet.

  Microsoft Dynamics CRM 2011 does not support querying a multi-level relationship property.Maximum 6 expansion are allowed.

  Following is an example for implementing oData using javascript in CRM 2011 to retrieve the required fields from Goal and Goal Metric entities.



As viewed in the snapshot above, three libraries have been included . It is compulsory to include jquery1.4.1.min.js and json2.js for implementing JSON format with oData.Both the files are available in SDK\ samplecode\ js\ restendpoint\restjquerycontacteditor\restjquerycontacteditor \scripts

Limitations of OData EndPoint in CRM 2011

1) The $format and $inlinecount operators are not supported. $filter, $select, $top, $skip, $orderby are supported.

2) Maximum 6 expansions are allowed using $expand operator. Querying a multi-level relationship property is not supported i.e. One level of navigation property selection is allowed.

3) Page size is fixed to max 50 records however it can be changed by doing changes in advanced configuration settings but it is not recommended.

4) When using with distinct queries, we are limited to the total (skip + top) record size = 5000. In CRM the distinct queries does not use paging cookie and so we are limited by the CRM platform limitation to the 5000 record.

5) Conditions on only one group of attributes are allowed. A group of attribute refers to a set of conditions joined by And/Or clause.

6) Arithmetic, datetime and math operators are not supported.

7) Order by clause is only allowed on the root entity.

8) Only Create, Retrieve, Update and Delete actions can be performed on entity records.

Messages that require the Execute method cannot be performed.

Associate and Disassociate actions are performed as updates.

9) Authentication is only possible within the application; Use of the REST endpoint is effectively limited to JScript libraries or Silverlight Web Resources.

10) The OData protocol is not fully implemented in CRM 2011. Some system query options are not available.

Performing CRUD Operations synchronously in CRM 2011 through JSON

Synchronous methods to create, update and delete the records for the Account entity through JSON. These methods works synchronously.

Create Records: To create the record into the CRM, you need to create an object for that record. After that you need to create the object of the **XMLHttpRequest** and open this request through "**POST**" method and need to call the send method with this **jsonEntity** (create from the target record).

```
function createRecord() {
var account = new Object();
account.Name = "Create Sample";
account.Telephone1 = "555-0123";
account.AccountNumber = "ABCDEFGHIJ";
account.EMailAddress1 = "someone1@example.com";
var jsonEntity = window.JSON.stringify(account);
var createRecordReq = new XMLHttpRequest();
var ODataPath = Xrm.Page.context.getServerUrl() +
"/XRMServices/2011/OrganizationData.svc";
createRecordReq.open('POST', ODataPath + "/" + "AccountSet", false);
createRecordReq.setRequestHeader("Accept", "application/json");
createRecordReq.setRequestHeader("Content-Type", "application/json; charset=utf-8");
createRecordReq.send(jsonEntity);
var newRecord = JSON.parse(createRecordReq.responseText).d;
//Alert the Id of the created record
alert("Id is: " + newRecord.AccountId);
}
```

Update Record: Update record request is same as the create but for the Update we need to set the Additional request header for the Method in the **XMLHttpRequest**. Here we have added the "**MERGE**" method. As shown below.

```
updateRecordReq.setRequestHeader("X-HTTP-Method", "MERGE");
```

And when we open the request, we need to pass the guid of the record. As given in the below script.

```
function updateRecord() {
var account = Object();
account.Name = "Update Name of this Account";
var jsonEntity = window.JSON.stringify(account);
var updateRecordReq = new XMLHttpRequest();
var ODataPath = Xrm.Page.context.getServerUrl() +
"/XRMServices/2011/OrganizationData.svc";
updateRecordReq.open('POST', ODataPath + "/" + "AccountSet" + "(guid'" + "E72B45B9-58E0-
E011-B700-00155D005515" + "')", false);
updateRecordReq.setRequestHeader("Accept", "application/json");
updateRecordReq.setRequestHeader("Content-Type", "application/json; charset=utf-8");
updateRecordReq.setRequestHeader("X-HTTP-Method", "MERGE");
updateRecordReq.send(jsonEntity);
}
```

Delete Record: This is same as update request but here we need to call the Method delete and call send method with the **null** parameter.

```
function deleteRecord() {
var deleteRecordReq = new XMLHttpRequest();
var ODataPath = Xrm.Page.context.getServerUrl() +
"/XRMServices/2011/OrganizationData.svc";
deleteRecordReq.open('POST', ODataPath + "/" + "AccountSet" + "(guid'" + "E72B45B9-58E0-
E011-B700-00155D005515" + "')", false);
deleteRecordReq.setRequestHeader("Accept", "application/json");
deleteRecordReq.setRequestHeader("Content-Type", "application/json; charset=utf-8");
deleteRecordReq.setRequestHeader("X-HTTP-Method", "DELETE");
deleteRecordReq.send(null);
}
```