# Virtual Agent Integration with MS Dynamics CRM (on-premise)

## Note about references

The following examples use the following references. These examples also use a reference to the Microsoft.Crm.Sdk, and System.Runtime.Serialization.

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.ServiceModel.Description;

using System.Text;

using System.Xml.Linq;

using Microsoft.Xrm.Sdk;

using Microsoft.Xrm.Sdk.Client;

using Microsoft.Xrm.Sdk.Query;

## Connect to the Dynamics server

The following code shows you how to use your authentication credentials to create a reference to a service proxy object. Once this reference is made, you can make calls directly on the service proxy object to access tables and resources on your Dynamics server. The service proxy must be disposed after use.

var \_serverProxy = new OrganizationServiceProxy(new Uri(service\_url), null, new ClientCredentials() { UserName = { UserName = username, Password = password } }, null);

 public void ConnectToDynamics()

 {

 using (\_serverProxy)

 {

 \_serverProxy.Authenticate();

 isConnectedToDynamics = true;

 Console.WriteLine("authenticated with Dynamics");

 }

 }

## Find or create a Contact record

This example shows you how to retrieve an EntityReference, this is an object that directly references an entity record in Dynamics, from the Contact table. We are going to try to find a Contact record with a given email address, and if we cannot find one, we’re going to create one. Contact records require a first and last name to be created in the system. Notice how we use the QueryByAttribute object to construct the query we will send to the server.

 private EntityReference FindContactWithEmail(string email, string firstname, string lastname)

 {

 //is there an existing contact with this email address?

 QueryByAttribute querybyexpression = new QueryByAttribute("contact");

 querybyexpression.Attributes.AddRange("emailaddress1");

 querybyexpression.Values.AddRange(email);

 EntityCollection retrieved = \_serverProxy.RetrieveMultiple(querybyexpression);

 var contact = retrieved.Entities.FirstOrDefault();

 if (contact != null)

 {

 return contact.ToEntityReference();

 }

 //create new contact with this email address?

 Entity newContact = new Entity("contact");

 newContact.Attributes["firstname"] = firstname;

 newContact.Attributes["lastname"] = lastname;

 newContact.Attributes["emailaddress1"] = email;

 Guid newContactGuid = \_serverProxy.Create(newContact);

 //fetch the entity of the newly created contact

 Entity foundnewContact = \_serverProxy.Retrieve("contact", newContactGuid, new ColumnSet(true));

 if (foundnewContact != null)

 {

 return foundnewContact.ToEntityReference();

 }

 throw new Exception("unable to create new contact in Dynamics for " + firstname + " " + lastname + " " + email);

 }

## Create a case in Dynamics

This example shows you how to create an incident record, or case, in Dynamics. First you create an Entity object and specify what type of Entity you are creating in the constructor, in this case it’s an incident Entity. We’re going to populate the title, description, and customerid fields of this new record before sending it to the server (notice how we use the FindContactWithEmail method from the previous example). We can construct a URL to this new record given the record GUID which is returned from the server on creation.

 public void CreateCase(string firstname, string lastname, string email, string question)

 {

 using (\_serverProxy)

 {

 //create the new case/incident/ticket

 Entity entity = new Entity("incident");

 entity.Attributes["title"] = question;

 entity.Attributes["caseorigincode"] = new OptionSetValue() { Value = 100000004 }; //100,000,004

 entity.Attributes["description"] = "Inbounded chat by Chime on " + DateTime.Now;

 entity.Attributes["customerid"] = FindContactWithEmail(email, firstname, lastname);

 Guid newRecordGuid = \_serverProxy.Create(entity);

 newRecordURL = instance\_url + @"main.aspx?etn=incident&id=" + newRecordGuid + @"&pagetype=entityrecord";

 }

 }

## Update a case in Dynamics

This example shows how you can update a record in Dynamics, given that record’s system GUID. This sample will add new text to any existing text of the description field of the case. When it retrieves the record it retrieves all columns. Note, the description field has a 2,000 character limit.

 public void UpdateCase(string comment, Guid recordGuid)

 {

 using (\_serverProxy)

 {

 Entity entity = \_serverProxy.Retrieve("incident", recordGuid, new ColumnSet(true));

 var olddescription = entity.Attributes["description"];

 var newdescription = olddescription + "\n" + comment;

 if (newdescription.Length < 2001)

 {

 entity.Attributes["description"] = newdescription;

 \_serverProxy.Update(entity);

 }

 }

 }