# 最最常用的方法

## 拿到字段的值

var value= Xrm.Page.getAttribute("attributename").getValue();

Xrm.Page.getAttribute("attributename").setValue(value);

## 操作lookup字段值

//Get a lookup value

var lookup = new Array();

lookup = Xrm.Page.getAttribute("attributename").getValue();

if (lookup != null) {

var name = lookup[0].name;

var id = lookup[0].id;

var entityType = lookup[0].entityType;

}

//Set a lookup value

var lookup = new Array();

lookup[0] = new Object();

lookup[0].id = recorid;

lookup[0].name = recordname;

lookup[0].entityType = entityname;

Xrm.Page.getAttribute("attributename").setValue(lookup);

Alternate method to set lookup value

Xrm.Page.getAttribute("attributename").setValue([{ id: recorid, name: recordname, entityType: entityname}]);

## 拿到当前记录的id

Xrm.Page.data.entity.getId();

## 拿到当前用户的id

Xrm.Page.context.getUserId();

## 获取当前页面的实体名

Xrm.Page.data.entity.getEntityName();

## 改变字段的必填

Xrm.Page.data.entity.attributes.get('new\_utilizationtype').setRequiredLevel("required");

Xrm.Page.data.entity.attributes.get('new\_adjustmenteffectiveweek').setRequiredLevel("none");

## 货币字段赋值

Xrm.Page.data.entity.attributes.get("currencyfield ").setValue(value);

Xrm.Page.data.entity.attributes.get("currencyfield ").getValue(value);

## lookup字段赋值

var owner = new Array();

owner[0] = new Object();

owner[0].entityType = "systemuser";

owner[0].id = Xrm.Page.context.getUserId();

Xrm.Page.data.entity.attributes.get("new\_fcaapprover").setValue(owner);

## 日期类型字段赋值

Xrm.Page.data.entity.attributes.get("new\_fcaapproverejecttime").setValue(new Date());

## 设置字段的保存模式

Xrm.Page.data.entity.attributes.get("new\_fcaapprover").setSubmitMode("always");

## 设置section的显示/隐藏

var secActual = Xrm.Page.ui.tabs.get("tabAdjustmentDetails").sections.get("secActual");

secActual.setVisible(false);

## 常用的保存数据的方法

Xrm.Page.data.refresh(true);

Xrm.Page.data.save().then(

function () {

successHandler();

},

function (error) {

Xrm.Utility.alertDialog(error.message);

});

//保存并刷新页面

Xrm.Page.data.refresh(true)

.then(function () {

var id = Xrm.Page.data.entity.getId();

var entity = Xrm.Page.data.entity.getEntityName();

Xrm.Utility.openEntityForm(entity, id);

});

## 阻止保存页面的方法





eContext 是上图中箭头所指，即页面context对象，

\_formSave: function (eContext) {

eContext.getEventArgs().preventDefault();

}

## IFrame里边设置父页面(CRM页面)上字段的值

parent.Xrm.Page.data.entity.attributes.get("new\_tagselected").setValue("value");

# Script操作数据

## 创建

//创建

SDK.REST.createRecord(

account,

"Account",

function (account) {

writeMessage("The account named \"" + account.Name + "\" was created with the AccountId : \"" + account.AccountId + "\".");

writeMessage("Retrieving account with the AccountId: \"" + account.AccountId + "\".");

retrieveAccount(account.AccountId)

},

errorHandler

);

this.setAttribute("disabled", "disabled");

}

## 查询

//查询

function retrieveAccount(AccountId) {

SDK.REST.retrieveRecord(

AccountId,

"Account",

null,null,

function (account) {

},

errorHandler

);

}

或者是：

function ParticipantAliasChange() {

var participantAlias = Xrm.Page.data.entity.attributes.get("new\_participantalias").getValue();

var option = "$filter=&$select=";

var result = RetrieveMultipleRecords("", option);

}

function RetrieveMultipleRecords(type, option) {

var serverUrl = Xrm.Page.context.getServerUrl();

var ODATA\_ENDPOINT = "/XRMServices/2011/OrganizationData.svc";

var ret;

$.ajax({

type: "GET",

async: false,

contentType: "application/json; charset=utf-8",

datatype: "json",

url: serverUrl + ODATA\_ENDPOINT + "/" + type + "Set?" + option,

beforeSend: function (XMLHttpRequest) {

//Specifying this header ensures that the results will be returned as JSON.

XMLHttpRequest.setRequestHeader("Accept", "application/json");

},

success: function (data, textStatus, XmlHttpRequest) {

ret = data.d.results;

},

error: function (XmlHttpRequest, textStatus, errorThrown) {

}

});

return ret;

}

function RetrieveRecord(type, id) {

var serverUrl = Xrm.Page.context.getServerUrl();

var ODATA\_ENDPOINT = "/XRMServices/2011/OrganizationData.svc";

var ret;

$.ajax({

type: "GET",

async: false,

contentType: "application/json; charset=utf-8",

datatype: "json",

url: serverUrl + ODATA\_ENDPOINT + "/" + type + "Set(guid'" + id + "')",

beforeSend: function (XMLHttpRequest) {

//Specifying this header ensures that the results will be returned as JSON.

XMLHttpRequest.setRequestHeader("Accept", "application/json");

},

success: function (data, textStatus, XmlHttpRequest) {

ret = data.d;

},

error: function (XmlHttpRequest, textStatus, errorThrown) {

}

});

return ret;

}

## 更新

//更新

function updateAccount(AccountId) {

var account = {};

writeMessage("Changing the account Name to \"Updated Account Name\".");

account.Name = "Updated Account Name";

writeMessage("Adding Address information");

account.Address1\_AddressTypeCode = { Value: 3 }; //Address 1: Address Type = Primary

account.Address1\_City = "Sammamish";

account.Address1\_Line1 = "123 Maple St.";

account.Address1\_PostalCode = "98074";

account.Address1\_StateOrProvince = "WA";

writeMessage("Setting E-Mail address");

account.EMailAddress1 = "someone@microsoft.com";

SDK.REST.updateRecord(

AccountId,

account,

"Account",

function () {

writeMessage("The account record changes were saved");

deleteAccount(AccountId);

},

errorHandler

);

}

## 删除

//删除

function deleteAccount(AccountId) {

if (confirm("Do you want to delete this account record?")) {

writeMessage("You chose to delete the account record.");

SDK.REST.deleteRecord(

AccountId,

"Account",

function () {

writeMessage("The account was deleted.");

enableResetButton();

},

errorHandler

);

}

else {

var li = document.createElement("li");

var span = document.createElement("span");

setElementText(span, "You chose not to delete the record. You can view the record ");

var link = document.createElement("a");

link.href = SDK.REST.\_getClientUrl() + "/main.aspx?etc=1&id=%7b" + AccountId + "%7d&pagetype=entityrecord";

link.target = "\_blank";

setElementText(link, "here");

li.appendChild(span);

li.appendChild(link);

output.appendChild(li);

enableResetButton();

}

}

function getFirstContactToBePrimaryContact() {

SDK.REST.retrieveMultipleRecords(

"Contact",

"$select=ContactId,FullName&$top=1",

function (results) {

var firstResult = results[0];

if (firstResult != null) {

primaryContact = results[0];

}

else {

writeMessage("No Contact records are available to set as the primary contact for the account.");

}

},

errorHandler,

function () {

//OnComplete handler

}

);

}

## 分派

var ret = AssignRequest("new\_ubiadjustment", Xrm.Page.data.entity.getId(), "team", assignee[0].TeamId, null);

if (ret == false) {

alert("Action Failed, the Team you assigned to have not enough privilege on this record.");

return;

}

function AssignRequest(entityType, entityId, assigneeType, assigneeId, assigneeName) {

var ret = true;

var request = "<s:Envelope xmlns:s=\"http://schemas.xmlsoap.org/soap/envelope/\">";

request += "<s:Body>";

request += "<Execute xmlns=\"http://schemas.microsoft.com/xrm/2011/Contracts/Services\"";

request += " xmlns:i=\"http://www.w3.org/2001/XMLSchema-instance\">";

request += "<request i:type=\"b:AssignRequest\"";

request += " xmlns:a=\"http://schemas.microsoft.com/xrm/2011/Contracts\"";

request += " xmlns:b=\"http://schemas.microsoft.com/crm/2011/Contracts\">";

request += "<a:Parameters xmlns:c=\"http://schemas.datacontract.org/2004/07/System.Collections.Generic\">";

request += "<a:KeyValuePairOfstringanyType>";

request += "<c:key>Target</c:key>";

request += "<c:value i:type=\"a:EntityReference\">";

request += "<a:Id>" + entityId + "</a:Id>";

request += "<a:LogicalName>" + entityType + "</a:LogicalName>";

request += "<a:Name i:nil=\"true\" />";

request += "</c:value>";

request += "</a:KeyValuePairOfstringanyType>";

request += "<a:KeyValuePairOfstringanyType>";

request += "<c:key>Assignee</c:key>";

request += "<c:value i:type=\"a:EntityReference\">";

request += "<a:Id>" + assigneeId + "</a:Id>";

request += "<a:LogicalName>" + assigneeType + "</a:LogicalName>";

if (assigneeName == null) {

request += "<a:Name i:nil=\"true\" />";

}

else {

request += "<a:Name>" + assigneeName + "</a:Name>";

}

request += "</c:value>";

request += "</a:KeyValuePairOfstringanyType>";

request += "</a:Parameters>";

request += "<a:RequestId i:nil=\"true\" />";

request += "<a:RequestName>Assign</a:RequestName>";

request += "</request>";

request += "</Execute>";

request += "</s:Body>";

request += "</s:Envelope>";

$.ajax({

type: "POST",

async: false,

contentType: "text/xml; charset=utf-8",

datatype: "xml",

url: Xrm.Page.context.getServerUrl() + "/XRMServices/2011/Organization.svc/web",

data: request,

beforeSend: function (XMLHttpRequest) {

XMLHttpRequest.setRequestHeader("Accept", "application/xml, text/xml, \*/\*");

XMLHttpRequest.setRequestHeader("SOAPAction", "http://schemas.microsoft.com/xrm/2011/Contracts/Services/IOrganizationService/Execute");

},

success: function (data, textStatus, XmlHttpRequest) {

ret = true;

},

error: function (XMLHttpRequest, textStatus, errorThrown) {

ret = false;

}

});

return ret;

}

# Script操作表单元素

## 操作元素数据

### 货币

account.Revenue = { Value: "2000000.00" }; //赋值

### 选项集

Odata的方式更新记录的时候：

account.PreferredContactMethodCode = { Value: 2 };

直接在页面上改字段值的时候：

Xrm.Page.getAttribute("new\_optionfield").setValue(1);

### 文本

account.Address1\_City = "Sammamish";

### 查找字段

account.PrimaryContactId = { Id: primaryContact.ContactId, LogicalName: "contact", Name: primaryContact.FullName } [赋值];

### 改变只读字段状态编辑字段进行调试

contentIFrame.Xrm.Page.getControl("Key").setDisabled(false)

### 添加删除选项

﻿/\*

Entity:new\_build\_standard 销一货架建标

Create:2013.12.23 Reven

\*/

var FORM\_TYPE\_CREATE = 1;

var FORM\_TYPE\_UPDATE = 2;

var FORM\_TYPE\_READ\_ONLY = 3;

var FORM\_TYPE\_DISABLED = 4;

var FORM\_TYPE\_QUICK\_CREATE = 5;

var FORM\_TYPE\_BULK\_EDIT = 6;

//onload

function onloadFunction() {

var formType = Xrm.Page.ui.getFormType();

if (formType==FORM\_TYPE\_UPDATE) {

Xrm.Page.getControl("new\_apply\_type").setDisabled(true);

}

}

//onchange

//修改单据类型

function onchangeNew\_apply\_type() {

//var formType = Xrm.Page.ui.getFormType();

//if (formType == FORM\_TYPE\_CREATE) {

// Xrm.Page.getAttribute("new\_counter\_type").setValue(null);

// var new\_apply\_type = Xrm.Page.getAttribute("new\_apply\_type").getValue();

// if (new\_apply\_type == 10) {//形象专柜费用申请

// Xrm.Page.getControl("new\_counter\_type").clearOptions();

// var options = Xrm.Page.getAttribute("new\_counter\_type").getOptions();

// for (var i = 0; i < options.length; i++) {

// if (options[i].value != 190 && options[i].value != 200 && options[i].value != "null") {

// try { Xrm.Page.getControl("new\_counter\_type").addOption(options[i], i + 1); } catch (e) { }

// }

// }

// } else if (new\_apply\_type == 20) {//陈列费用标准申请

// Xrm.Page.getControl("new\_counter\_type").clearOptions();

// Xrm.Page.getControl("new\_counter\_type").addOption(Xrm.Page.getAttribute("new\_counter\_type").getOption(190), 1);//整洁

// Xrm.Page.getControl("new\_counter\_type").addOption(Xrm.Page.getAttribute("new\_counter\_type").getOption(200), 2);//男士

// }

//}

}

//if (new\_flow\_type == 80 && new\_prtype == 10 && (formType == 1 || formType == 2)) {

// Xrm.Page.getAttribute("new\_receive\_type").setRequiredLevel("required");

// Xrm.Page.ui.controls.get("new\_receive\_type").setDisabled(false);

// Xrm.Page.getControl("new\_receive\_type").clearOptions();

// Xrm.Page.getControl("new\_receive\_type").addOption(Xrm.Page.getAttribute("new\_receive\_type").getOption(10), 1);

// Xrm.Page.getControl("new\_receive\_type").addOption(Xrm.Page.getAttribute("new\_receive\_type").getOption(20), 2);

// if (new\_receive\_type != null && formType == 2) {

// Xrm.Page.data.entity.attributes.get("new\_receive\_type").setValue(new\_receive\_type);

// }

//}

//else {

// Xrm.Page.getControl("new\_receive\_type").clearOptions();

// Xrm.Page.getControl("new\_receive\_type").addOption(Xrm.Page.getAttribute("new\_receive\_type").getOption(10), 1);

// Xrm.Page.getControl("new\_receive\_type").addOption(Xrm.Page.getAttribute("new\_receive\_type").getOption(20), 2);

// Xrm.Page.getControl("new\_receive\_type").addOption(Xrm.Page.getAttribute("new\_receive\_type").getOption(30), 3);

//}

如果A实体中有lookup字段关联的是A实体自己，Message=Delete在plugin中获取lookup字段的值必须是验证前的

### 获取初始值

Xrm.Page.data.entity.attributes.get("new\_flow\_type").getInitialValue()

### 获取窗体是否未保存

Xrm.Page.data.entity.getIsDirty()

## 设置表单元素

### 设置Tab

Xrm.Page.ui.tabs.get("Key").setVisible(false);//隐藏

Xrm.Page.ui.tabs.get("Key").setVisible(true);//显示

### Disable页面所有字段

function DoesControlHaveAttribute(control) {

var controlType = control.getControlType();

return controlType != "iframe" && controlType != "webresource" && controlType != "subgrid";

}

function DisableFormFields() {

Xrm.Page.ui.controls.forEach(function (control, index) {

if (DoesControlHaveAttribute(control)) {

control.setDisabled(true);

}

});

}

### 设置Tab中的节

Xrm.Page.ui.tabs.get("tab\_1").sections.get("tab\_2\_section\_6").setVisible(false);//隐藏

Xrm.Page.ui.tabs.get("tab\_1").sections.get("tab\_2\_section\_6").setVisible(true);//显示

### 设置字段只读/可编辑

Xrm.Page.getControl("new\_systemuserid").setDisabled(true);

Xrm.Page.getControl("new\_systemuserid").setDisabled(false);

### 设置字段可见/隐藏

Xrm.Page.getControl("new\_systemuserid").setVisible(true);

Xrm.Page.getControl("new\_systemuserid").setVisible(false);

### 设置字段赋值后自动提交

Xrm.Page.getAttribute("new\_systemuserid").setSubmitMode("always");

### 传递dialog的值给后边的页面

window.opener.window.parent.Xrm.Page.getAttribute("ws\_name").setValue('fsdfs');

### 设置子网格的显示和隐藏

**var jj\_subGrid = Xrm.Page.ui.controls.get("jj\_ContactsGrid");**

**// For hiding**

**jj\_subGrid.setVisible(false);**

**// For Showing**

**jj\_subGrid.setVisible(true);**

**// For Checking If it is Visible or Hide**

**jj\_subGrid.getVisible();**

## 其他常用方法

### 拿到用户所属的team

function GetTeamsOfUser(ownerId) {

var xml = "" +

"<?xml version=\"1.0\" encoding=\"utf-8\"?>" +

"<soap:Envelope xmlns:soap=\"http://schemas.xmlsoap.org/soap/envelope/\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" xmlns:xsd=\"http://www.w3.org/2001/XMLSchema\">" +

GenerateAuthenticationHeader() +

" <soap:Body>" +

" <RetrieveMultiple xmlns=\"http://schemas.microsoft.com/crm/2007/WebServices\">" +

" <query xmlns:q1=\"http://schemas.microsoft.com/crm/2006/Query\" xsi:type=\"q1:QueryExpression\">" +

" <q1:EntityName>team</q1:EntityName>" +

" <q1:ColumnSet xsi:type=\"q1:ColumnSet\">" +

" <q1:Attributes>" +

" <q1:Attribute>name</q1:Attribute>" +

" </q1:Attributes>" +

" </q1:ColumnSet>" +

" <q1:Distinct>false</q1:Distinct>" +

" <q1:LinkEntities>" +

" <q1:LinkEntity>" +

" <q1:LinkFromAttributeName>teamid</q1:LinkFromAttributeName>" +

" <q1:LinkFromEntityName>team</q1:LinkFromEntityName>" +

" <q1:LinkToEntityName>teammembership</q1:LinkToEntityName>" +

" <q1:LinkToAttributeName>teamid</q1:LinkToAttributeName>" +

" <q1:JoinOperator>Inner</q1:JoinOperator>" +

" <q1:LinkCriteria>" +

" <q1:FilterOperator>And</q1:FilterOperator>" +

" <q1:Conditions>" +

" <q1:Condition>" +

" <q1:AttributeName>systemuserid</q1:AttributeName>" +

" <q1:Operator>Equal</q1:Operator>" +

"<q1:Values>" +

//code to get the owner

"<q1:Value xsi:type=\"xsd:string\">" + ownerId + "</q1:Value>" +

"</q1:Values>" +

" </q1:Condition>" +

" </q1:Conditions>" +

" </q1:LinkCriteria>" +

" </q1:LinkEntity>" +

" </q1:LinkEntities>" +

" </query>" +

" </RetrieveMultiple>" +

" </soap:Body>" +

"</soap:Envelope>" +

"";

var xmlHttpRequest = new ActiveXObject("Msxml2.XMLHTTP");

xmlHttpRequest.Open("POST", "/mscrmservices/2007/CrmService.asmx", false);

xmlHttpRequest.setRequestHeader("SOAPAction", "http://schemas.microsoft.com/crm/2007/WebServices/RetrieveMultiple");

xmlHttpRequest.setRequestHeader("Content-Type", "text/xml; charset=utf-8");

xmlHttpRequest.setRequestHeader("Content-Length", xml.length);

xmlHttpRequest.send(xml);

var resultXml = xmlHttpRequest.responseXML;

var entityNodes = resultXml.selectNodes("//RetrieveMultipleResult/BusinessEntities/BusinessEntity");

var teamsArray = new Array();

for (var i = 0; i < entityNodes.length; i++) {

var teamItem = new Object();

var entityNode = entityNodes[i];

var teamidNode = entityNode.selectSingleNode("q1:teamid");

var teamNode = entityNode.selectSingleNode("q1:name");

var teamid = (teamidNode == null) ? null : teamidNode.text;

var team = (teamNode == null) ? null : teamNode.text;

teamItem.teamID = teamid;

teamItem.teamName = team;

teamsArray[i] = teamItem;

}

return teamsArray;

}

### 查询N:N记录

Type是这个N:N 关系的logicalname

var type = "new\_campaign\_contact";

var option = "$select=new\_campaign\_contactId&$filter=campaignid eq guid'" + Xrm.Page.data.entity.getId()+"+'";

var result = RetrieveMultipleRecords(type, option);

function RetrieveMultipleRecords(type, option) {

var serverUrl = Xrm.Page.context.getServerUrl();

var ODATA\_ENDPOINT = "/XRMServices/2011/OrganizationData.svc";

var ret;

$.ajax({

type: "GET",

async: false,

contentType: "application/json; charset=utf-8",

datatype: "json",

url: serverUrl + ODATA\_ENDPOINT + "/" + type + "Set?" + option,

beforeSend: function (XMLHttpRequest) {

//Specifying this header ensures that the results will be returned as JSON.

XMLHttpRequest.setRequestHeader("Accept", "application/json");

},

success: function (data, textStatus, XmlHttpRequest) {

ret = data.d.results;

},

error: function (XmlHttpRequest, textStatus, errorThrown) {

}

});

return ret;

}

function RetrieveNNRelationship(FromEntity, ToEntity, RelationShip, reasonId) {

var fetchxml =

"<fetch version='1.0' output-format='xml-platform' mapping='logical' distinct='true' >" +

"<entity name='" + FromEntity + "'>" +

"<attribute name='" + FromEntity + "id' />" +

"<attribute name='domainname'/>" +

"<attribute name='internalemailaddress'/>" +

"<attribute name='firstname'/>" +

"<attribute name='lastname'/>" +

"<link-entity name='" + RelationShip + "' from='" + FromEntity + "id' to='" + FromEntity + "id'>" +

"<link-entity name='" + ToEntity + "' from='" + ToEntity + "id' to='" + ToEntity + "id'>" +

"<filter type='and'>" +

"<condition attribute='" + ToEntity + "id' operator='eq' value='" + reasonId + "' />" +

"</filter>" +

"</link-entity>" +

"</link-entity>" +

"</entity>" +

"</fetch>";

var request = "<s:Envelope xmlns:s=\"http://schemas.xmlsoap.org/soap/envelope/\">";

request += "<s:Body>";

request += '<Execute xmlns="http://schemas.microsoft.com/xrm/2011/Contracts/Services">' + '<request i:type="b:RetrieveMultipleRequest" ' + ' xmlns:b="http://schemas.microsoft.com/xrm/2011/Contracts" ' + ' xmlns:i="http://www.w3.org/2001/XMLSchema-instance">' + '<b:Parameters xmlns:c="http://schemas.datacontract.org/2004/07/System.Collections.Generic">' + '<b:KeyValuePairOfstringanyType>' + '<c:key>Query</c:key>' + '<c:value i:type="b:FetchExpression">' + '<b:Query>';

request += CrmEncodeDecode.CrmXmlEncode(fetchxml);

request += '</b:Query>' + '</c:value>' + '</b:KeyValuePairOfstringanyType>' + '</b:Parameters>' + '<b:RequestId i:nil="true"/>' + '<b:RequestName>RetrieveMultiple</b:RequestName>' + '</request>' + '</Execute>';

request += '</s:Body></s:Envelope>';

var serverURL = Xrm.Page.context.getServerUrl();

var xmlhttp = new XMLHttpRequest();

xmlhttp.open("POST", serverURL + "/XRMServices/2011/Organization.svc/web", false);

xmlhttp.setRequestHeader("Accept", "application/xml, text/xml, \*/\*");

xmlhttp.setRequestHeader("Content-Type", "text/xml; charset=utf-8");

xmlhttp.setRequestHeader("SOAPAction", "http://schemas.microsoft.com/xrm/2011/Contracts/Services/IOrganizationService/Execute");

try { xmlhttp.responseType = 'msxml-document'; } catch (e) { };

xmlhttp.send(request);

xmlhttp.responseXML.setProperty("SelectionNamespaces", "xmlns:a=\'http://schemas.microsoft.com/xrm/2011/Contracts\'");

var sFetchResult = xmlhttp.responseXML.selectSingleNode("//a:Entities").xml;

var resultDoc = new ActiveXObject("Microsoft.XMLDOM");

resultDoc.async = false;

resultDoc.loadXML(sFetchResult);

//parse result xml into array of jsDynamicEntity objects

var results = new Array(resultDoc.firstChild.childNodes.length);

for (var i = 0; i < resultDoc.firstChild.childNodes.length; i++) {

var userEntity = resultDoc.firstChild.childNodes[i];

var userItem = new Object();

var userId = "";

var userName = "";

for (var j = 0; j < userEntity.childNodes.length; j++) {

var key = userEntity.childNodes[j];

if (key.baseName == "Attributes") {

for (var k = 0; k < key.childNodes.length; k++) {

var attr = key.childNodes[k];

if (attr.childNodes[0].nodeTypedValue == FromEntity + "id") {

userId = attr.childNodes[1].nodeTypedValue;

}

if (attr.childNodes[0].nodeTypedValue == "firstname")

var firstname = attr.childNodes[1].nodeTypedValue;

if (attr.childNodes[0].nodeTypedValue == "lastname")

var lastname = attr.childNodes[1].nodeTypedValue;

userName = firstname + " " + lastname;

}

}

}

userItem.id = userId;

userItem.name = userName;

results[i] = userItem;

}

return results;

}

### 跳转页面

function redirectaccount(accountid) {

Xrm.Page.ui.close();//关闭当前页面

Xrm.Utility.openEntityForm("account", accountid);//打开其他记录页面

}

### 打开创建记录页面并传递参数

var parameters = {};

parameters["new\_productidname"]=new\_name;

//parameters["new\_productidtype"]="new\_product"; // 如果是客户或者用户等一些关于实体可能还需要给type设值

// 选项集

parameters["new\_appoinment"] = (new\_reservationmode.Value).toString();

        Xrm.Utility.openEntityForm("new\_subscribe",null,parameters);//打开创建的记录

### 刷新Form上的子网格 SubGrid

**Xrm.Page.getControl("addressee").refresh()**

**其中参数是子网格的名字**

### 刷新Form

**this.parent.window.location.reload();**

**Xrm.Page.data.refresh();**

**保存并刷新**

Xrm.Page.data.refresh(true)

.then(function () {

var id = Xrm.Page.data.entity.getId();

var entity = Xrm.Page.data.entity.getEntityName();

Xrm.Utility.openEntityForm(entity, id);

});

### 刷新主界面的View

**Mscrm.Utilities.refreshCurrentGrid(10029);**

**其中10029是实体的Code值，可以在后台数据库"EntityView"视图 中查看。**

### 新建记录保存后刷新当前页面

（CRM2013以上版本保存的时候只进行局部刷新，无法刷新导航栏）：

Xrm.Utility.openEntityForm(Xrm.Page.data.entity.getEntityName(), Xrm.Page.data.entity.getId());

# Plugin(C#)

## 分派

AssignRequest assign = new AssignRequest();

assign.Assignee = prEntity["ownerid"] as EntityReference;

assign.Target = new EntityReference("new\_budgetused", new\_budgetusedId);

\_service.Execute(assign);

## 共享

#region 插件中CRM共享共用方法

/// <summary>

/// 共享

/// </summary>

/// <param name="initentityName">要求共享的实体名称</param>

/// <param name="userId">要求共享的实体GUID</param>

/// <param name="entityName">指定共享的实体名称</param>

/// <param name="entityId">指定共享的实体GUID</param>

/// <param name="service">CRM组织服务</param>

private void ShareEntity(string initentityName, Guid userId, string entityName, Guid entityId, IOrganizationService service)

{

GrantAccessRequest grant = new GrantAccessRequest();

grant.PrincipalAccess = new PrincipalAccess

{

//读、写、附加等权限共享

AccessMask = AccessRights.ReadAccess | AccessRights.WriteAccess | AccessRights.AppendAccess,

Principal = new EntityReference(initentityName, userId)//共享给某个用户

};

grant.Target = new EntityReference(entityName, entityId);//要共享的对象

service.Execute(grant);

}

#endregion

## 取消共享

public void RevokeShareRecords(string LogicName,string targetEntityName,string usedAttrName, IOrganizationService service, Guid targetEntityId,string[] attrs,object[] values,string[] columnSet)

{

QueryByAttribute query = new QueryByAttribute(LogicName);

query.Attributes.AddRange(attrs);

query.Values.AddRange(values);

query.ColumnSet = new ColumnSet(columnSet);

EntityCollection userCollection = service.RetrieveMultiple(query);

foreach (Entity entity in userCollection.Entities)

{

//Guid AttrId = (Guid)entity.Attributes[SegmentInfo.SystemUserId];

Guid UserId = (Guid)entity.Attributes[usedAttrName];

EntityReference er = new EntityReference("systemuser", UserId);

RevokeAccessRequest revokeAccessRequest = new RevokeAccessRequest {

Revokee = new EntityReference("systemuser", UserId),

Target = new EntityReference(targetEntityName, targetEntityId)

};

service.Execute(revokeAccessRequest);

}

}

## 查询

string fetchxml = @"<fetch version='1.0' output-format='xml-platform' mapping='logical' distinct='false'>

<entity name='" + objecttype + @"'>

<attribute name='new\_name' />

<filter type='and'>

<condition attribute='statecode' operator='eq' value='0' />

<condition attribute='new\_name' operator='eq' value='" + name + @"' />

</filter>

</entity>

</fetch>";

EntityCollection entityCollection = service.RetrieveMultiple(new FetchExpression(fetchxml));

## 获取最顶级的上下文，用于获取当前操作用户

public IPluginExecutionContext GetTopContext(IPluginExecutionContext currentContext)

{

IPluginExecutionContext topContext = currentContext;

for (int i = 0; i <= currentContext.Depth; i++)

{

if (topContext.ParentContext != null)

{

topContext = topContext.ParentContext;

}

}

return topContext;

}

## 获取sharepoint上的文档

public static DataCollection<Entity> RetrieveSharepointNotes(IOrganizationService service)

{

QueryExpression query = new QueryExpression("annotation")

{

//ColumnSet = new ColumnSet(true),

Criteria =

{

Filters =

{

new FilterExpression(LogicalOperator.Or)

{

Conditions =

{

new ConditionExpression("subject",ConditionOperator.Equal, "Attachment Error"),

},

Filters =

{

new FilterExpression(LogicalOperator.And)

{

Conditions =

{

new ConditionExpression("subject", ConditionOperator.Equal, "File Attachment"),

new ConditionExpression("notetext", ConditionOperator.Like, "http://spark.partners.extranet.microsoft.com%"),

}

}

}

}

}

}

};

EntityCollection ec = service.RetrieveMultiple(query);

return ec.Entities;

}

## Query 转化为FetchXML/ FetchXML转化为Query

public static string ConvertQueryToFetchXml(QueryExpression query, IOrganizationService crmService)

{

QueryExpressionToFetchXmlRequest request = new QueryExpressionToFetchXmlRequest();

request.Query = query;

QueryExpressionToFetchXmlResponse response

= (QueryExpressionToFetchXmlResponse)crmService.Execute(request);

return response.FetchXml;

}

public static QueryExpression ConvertFetchXmlToQuery(string fetchXml, IOrganizationService crmService)

{

FetchXmlToQueryExpressionRequest fetchXmlRequest = new FetchXmlToQueryExpressionRequest();

fetchXmlRequest.FetchXml = fetchXml;

FetchXmlToQueryExpressionResponse fetchXmlResponse

= (FetchXmlToQueryExpressionResponse)crmService.Execute(fetchXmlRequest);

return fetchXmlResponse.Query;

}

## 获取optionset字段的显示名

//string name = GetPickListText("opportunity", "new\_opportunitystate", 0, service);

public static string GetPickListText(string entityName, string attributeName, int optionSetValue, IOrganizationService service)

{

string AttributeName = attributeName;

string EntityLogicalName = entityName;

RetrieveEntityRequest retrieveDetails = new RetrieveEntityRequest();

retrieveDetails.EntityFilters = EntityFilters.All;

retrieveDetails.LogicalName = EntityLogicalName;

RetrieveEntityResponse retrieveEntityResponseObj = (RetrieveEntityResponse)service.Execute(retrieveDetails);

EntityMetadata metadata = retrieveEntityResponseObj.EntityMetadata;

PicklistAttributeMetadata picklistMetadata = metadata.Attributes.FirstOrDefault(attribute => String.Equals(attribute.LogicalName, attributeName, StringComparison.OrdinalIgnoreCase)) as PicklistAttributeMetadata;

OptionSetMetadata options = picklistMetadata.OptionSet;

IList<OptionMetadata> picklistOption = (from o in options.Options where o.Value.Value == optionSetValue select o).ToList();

string picklistLabel = (picklistOption.First()).Label.UserLocalizedLabel.Label;

return picklistLabel;

}

## 改变记录的状态

public static void ChangeRecordState(IOrganizationService service, EntityReference entityMoniker, OptionSetValue state, OptionSetValue status)

{

SetStateRequest req = new SetStateRequest

{

EntityMoniker = entityMoniker,

State = state,

Status = status

};

service.Execute(req);

}

## 从template创建邮件

public static void CreateEmailFromTemplate(IOrganizationService service, EntityCollection sendFromCol, EntityCollection sendToCol, string emailSubject, string previewTriggerVaule, EntityReference regard, Guid contactId, Entity template, string btnOrder)

{

if (template != null)

{

// Use the InstantiateTemplate message to create an e-mail message using a template.

InstantiateTemplateRequest instTemplateReq = new InstantiateTemplateRequest

{

TemplateId = template.Id,

ObjectId = contactId,

ObjectType = "contact"

};

InstantiateTemplateResponse instTemplateResp = (InstantiateTemplateResponse)service.Execute(instTemplateReq);

Entity email = instTemplateResp.EntityCollection.Entities[0];

email.Attributes["new\_campaigninvatationid"] = previewTriggerVaule;

email.Attributes["subject"] = emailSubject;

email.Attributes.Add("from", sendFromCol);

service.Create(email);

}

}

## 查询N:N记录

public static EntityCollection RetrieveNNRecords(IOrganizationService service, Guid ToEntityId, string FromEntity, string ToEntity, string RelationShip)

{

QueryExpression qe = new QueryExpression()

{

EntityName = FromEntity,

ColumnSet = new ColumnSet(true),

Criteria =

{

FilterOperator = LogicalOperator.And,

Conditions =

{

new ConditionExpression("statecode",ConditionOperator.Equal,0)

}

},

LinkEntities =

{

new LinkEntity()

{

LinkFromEntityName = FromEntity,

LinkToEntityName = RelationShip,

LinkFromAttributeName = FromEntity+"id",

LinkToAttributeName = FromEntity+"id",

EntityAlias = RelationShip,

Columns = new ColumnSet(),

JoinOperator = JoinOperator.Inner,

LinkEntities =

{

new LinkEntity()

{

LinkFromEntityName = FromEntity,

LinkToEntityName = ToEntity,

LinkFromAttributeName = ToEntity+"id",

LinkToAttributeName = ToEntity+"id",

EntityAlias =ToEntity,

Columns = new ColumnSet(),

JoinOperator = JoinOperator.Inner,

LinkCriteria = new FilterExpression

{

Conditions =

{

new ConditionExpression

{

AttributeName = ToEntity+"id",

Operator = ConditionOperator.Equal,

Values = {

ToEntityId

}

}

}

}

}

}

}

}

};

return service.RetrieveMultiple(qe);

}

}

## 用户是否有指定的security role

public static bool UserHaveSpeficySecurityRole(IOrganizationService service, string roleName, Guid userID)

{

bool result = false;

QueryExpression query = new QueryExpression()

{

EntityName = "role",

ColumnSet = new ColumnSet(),

Criteria = new FilterExpression { FilterOperator = LogicalOperator.And, Conditions = { new ConditionExpression { AttributeName = "name", Operator = ConditionOperator.Equal, Values = { roleName } } } },

LinkEntities =

{

new LinkEntity

{

LinkFromEntityName = "role",

LinkFromAttributeName = "roleid",

LinkToEntityName = "systemuserroles",

LinkToAttributeName = "roleid",

LinkCriteria = new FilterExpression

{

FilterOperator = LogicalOperator.And,

Conditions =

{

new ConditionExpression

{

AttributeName = "systemuserid",

Operator = ConditionOperator.Equal,

Values = { userID}

}

}

}

}

}

};

EntityCollection resultColl = service.RetrieveMultiple(query);

result = resultColl.Entities.Count > 0 ? true : false;

return result;

}

## 查询一条记录分派给哪些用户

public static List<EntityReference> RetrieveSharedPrincipalsAndAccess(IOrganizationService service, EntityReference entityRef)

{

try

{

var accessRequest = new RetrieveSharedPrincipalsAndAccessRequest

{

Target = entityRef

};

RetrieveSharedPrincipalsAndAccessResponse accessResponse = (RetrieveSharedPrincipalsAndAccessResponse)service.Execute(accessRequest);

List<EntityReference> accessList = null;

if (accessResponse != null && accessResponse.PrincipalAccesses.Length > 0)

{

accessList = new List<EntityReference>();

for (int i = 0; i < accessResponse.PrincipalAccesses.Length; i++)

{

EntityReference accessOwner = accessResponse.PrincipalAccesses[i].Principal;

accessList.Add(accessOwner);

}

}

return accessList;

}

catch (Exception e)

{

throw new Exception("Customized Plugin RetrieveSharedPrincipalsAndAccess Error: " + e.Message);

}

}

## 拿到拥有指定security role的所有用户

public static ArrayList getUsersinRole(string Role\_Name, IOrganizationService service)

{

ArrayList usersinrole = new ArrayList();

StringBuilder fetch2 = new StringBuilder();

fetch2.Append("<fetch mapping='logical' count='50' version='1.0'> ");

fetch2.Append(" <entity name='systemuser'>");

fetch2.Append(" <attribute name='fullname'/> ");

fetch2.Append(" <link-entity name='systemuserroles' to='systemuserid' from='systemuserid'> ");

fetch2.Append(" <link-entity name='role' to='roleid' from='roleid'> ");

fetch2.Append(" <filter> ");

fetch2.Append(" <condition attribute='name' operator='eq' value='" + Role\_Name + "'/> ");

fetch2.Append(" </filter> ");

fetch2.Append(" </link-entity> ");

fetch2.Append(" </link-entity>");

fetch2.Append(" </entity> ");

fetch2.Append("</fetch>");

EntityCollection result = service.RetrieveMultiple(new FetchExpression(fetch2.ToString()));

foreach (var c in result.Entities)

{

usersinrole.Add(c.Attributes["fullname"].ToString());

}

return usersinrole;

}

## 判断用户是否是某团队的一员

public bool IsMemberInTeam(IOrganizationService service, Guid teamId, Guid memberId)

{

OrganizationServiceContext context = new OrganizationServiceContext(service);

var query = from relationship in context.CreateQuery("teammembership")

where relationship.GetAttributeValue<Guid>("teamid") == teamId

&& relationship.GetAttributeValue<Guid>("systemuserid") == memberId

select relationship;

return query.FirstOrDefault() != null;

}

## 添加用户到一个团队

OrganizationRequest request = new AddMembersTeamRequest { MemberIds = new Guid[] { user.Id }, TeamId = team.Id };

var response = service.Execute(request) as AddMembersTeamResponse;

## C# 分页查询记录

public static void DoTest(IOrganizationService service)

{

// Query using the paging cookie.

// Define the paging attributes.

// The number of records per page to retrieve.

int fetchCount = 20;

// Initialize the page number.

int pageNumber = 1;

// Define the order expression to retrieve the records.

OrderExpression order = new OrderExpression();

order.AttributeName = "name";

order.OrderType = OrderType.Ascending;

// Create the query expression and add condition.

QueryExpression pagequery = new QueryExpression();

pagequery.EntityName = "opportunity";

// pagequery.Criteria.AddCondition(pagecondition);

pagequery.Orders.Add(order);

pagequery.ColumnSet.AddColumns();

// Assign the pageinfo properties to the query expression.

pagequery.PageInfo = new PagingInfo();

pagequery.PageInfo.Count = fetchCount;

pagequery.PageInfo.PageNumber = pageNumber;

// The current paging cookie. When retrieving the first page,

// pagingCookie should be null.

pagequery.PageInfo.PagingCookie = null;

while (true)

{

// Retrieve the page.

EntityCollection results = service.RetrieveMultiple(pagequery);

if (results.Entities != null)

{

for (int i = 0; i < results.Entities.Count;i++ )

{

Console.WriteLine(" {0} {1}", results.Entities[i].Id, i + 1);

}

}

// Check for more records, if it returns true.

if (results.MoreRecords)

{

// Increment the page number to retrieve the next page.

pagequery.PageInfo.PageNumber++;

// Set the paging cookie to the paging cookie returned from current results.

pagequery.PageInfo.PagingCookie = results.PagingCookie;

}

else

{

// If no more records are in the result nodes, exit the loop.

break;

}

}

}

## 更新用户所在的业务部门

SetBusinessSystemUserRequest req = new SetBusinessSystemUserRequest();

req.BusinessId = Guid.Parse("52A18602-09B8-E511-80C2-807DB137DB06"); BU的GUID

req.UserId = Guid.Parse("7F038A46-5BB4-E511-80C6-DD48FB4179EC"); User的GUID

req.ReassignPrincipal = new EntityReference("systemuser", Guid.Parse("7F038A46-5BB4-E511-80C6-DD48FB4179EC")); User的GUID

service.Execute(req);

## ConditionOperator.In的使用方式

OptionSet的时候：

query.Criteria.AddCondition("new\_hpy\_zpttype", ConditionOperator.In, 1, 4, 5 );

其他时候：

query.Criteria.AddCondition("new\_hpy\_zpttype", ConditionOperator.In, new object[]{“123”,”456”,”789”} );

## C#方法触发计算汇总字段

CalculateRollupFieldRequest 详见SDK

# 其他C#方法

## 写日志

public void Write(string text)

{

string fileName = DateTime.Now.Year.ToString() + DateTime.Now.Month.ToString() + DateTime.Now.Day;

FileStream fs = new FileStream("C:\\" + fileName + "test.txt", FileMode.Append);

StreamWriter sw = new StreamWriter(fs, Encoding.Default);

sw.Write(text);

sw.Close();

fs.Close();

}

## C# 分页查询记录

private static void Run(String connectionString)

{

try

{

// Establish a connection to the organization web service using CrmConnection.

CrmConnection connection = CrmConnection.Parse(connectionString);

// Obtain an organization service proxy.

// The using statement assures that the service proxy will be properly disposed.

using (service = new OrganizationService(connection))

{

// //type your code below

// //-----------------------------------------------------------------------

// Define the fetch attributes.

// Set the number of records per page to retrieve.

int fetchCount = 100;

// Initialize the page number.

int pageNumber = 1;

// Initialize the number of records.

int recordCount = 0;

// Specify the current paging cookie. For retrieving the first page,

// pagingCookie should be null.

string pagingCookie = null;

var recordCountEdit = 0;

// Create the FetchXml string for retrieving all child accounts to a parent account.

// This fetch query is using 1 placeholder to specify the parent account id

// for filtering out required accounts. Filter query is optional.

// Fetch query also includes optional order criteria that, in this case, is used

// to order the results in ascending order on the name data column.

var fetchxml =

"<fetch>" +

"<entity name='account' >" +

"<attribute name='name' />" +

"<attribute name='new\_advisorydivision' />" +

"<filter type='and' >" +

//"<condition attribute='new\_cus\_cardtype' operator='eq' value='1' />" +

"<condition attribute='new\_advisorydivision' operator='not-null' />" +

"</filter>" +

"<link-entity name='systemuser' from='systemuserid' to='ownerid' >" +

"<link-entity name='businessunit' from='businessunitid' to='new\_belongcenter' >" +

"<filter type='and' >" +

"<condition attribute='name' operator='eq' value='伊美尔幸福医院' />" +

"</filter>" +

"</link-entity>" +

"</link-entity>" +

"</entity>" +

"</fetch>";

while (true)

{

var startTime = DateTime.Now;

// Build fetchXml string with the placeholders.

string xml = CreateXml(fetchxml, pagingCookie, pageNumber, fetchCount);

// Excute the fetch query and get the xml result.

RetrieveMultipleRequest fetchRequest1 = new RetrieveMultipleRequest

{

Query = new FetchExpression(xml)

};

EntityCollection returnCollection = ((RetrieveMultipleResponse)service.Execute(fetchRequest1)).EntityCollection;

foreach (var c in returnCollection.Entities)

{

//do something here

}

// Check for morerecords, if it returns 1.

if (returnCollection.MoreRecords)

{

Console.WriteLine("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\nPage number {0}\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*", pageNumber);

// Increment the page number to retrieve the next page.

pageNumber++;

// Set the paging cookie to the paging cookie returned from current results.

pagingCookie = returnCollection.PagingCookie;

}

else

{

// If no more records in the result nodes, exit the loop.

break;

}

var endTime = DateTime.Now;

Console.WriteLine("time used: " + (endTime-startTime));

if (pageNumber == 5)

break;

}

Console.WriteLine("ok");

}

}

// Catch any service fault exceptions that Microsoft Dynamics CRM throws.

catch (FaultException<OrganizationServiceFault>)

{

throw;

}

}

public static string CreateXml(string xml, string cookie, int page, int count)

{

StringReader stringReader = new StringReader(xml);

XmlTextReader reader = new XmlTextReader(stringReader);

// Load document

XmlDocument doc = new XmlDocument();

doc.Load(reader);

return CreateXml(doc, cookie, page, count);

}

public static string CreateXml(XmlDocument doc, string cookie, int page, int count)

{

XmlAttributeCollection attrs = doc.DocumentElement.Attributes;

if (cookie != null)

{

XmlAttribute pagingAttr = doc.CreateAttribute("paging-cookie");

pagingAttr.Value = cookie;

attrs.Append(pagingAttr);

}

XmlAttribute pageAttr = doc.CreateAttribute("page");

pageAttr.Value = System.Convert.ToString(page);

attrs.Append(pageAttr);

XmlAttribute countAttr = doc.CreateAttribute("count");

countAttr.Value = System.Convert.ToString(count);

attrs.Append(countAttr);

StringBuilder sb = new StringBuilder(1024);

StringWriter stringWriter = new StringWriter(sb);

XmlTextWriter writer = new XmlTextWriter(stringWriter);

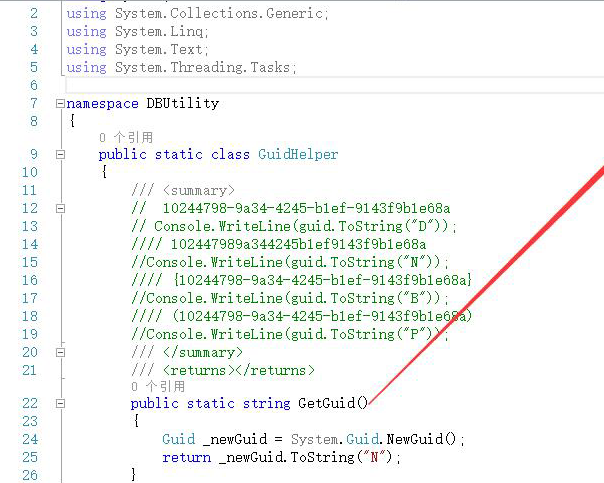
doc.WriteTo(writer);

writer.Close();

return sb.ToString();

}

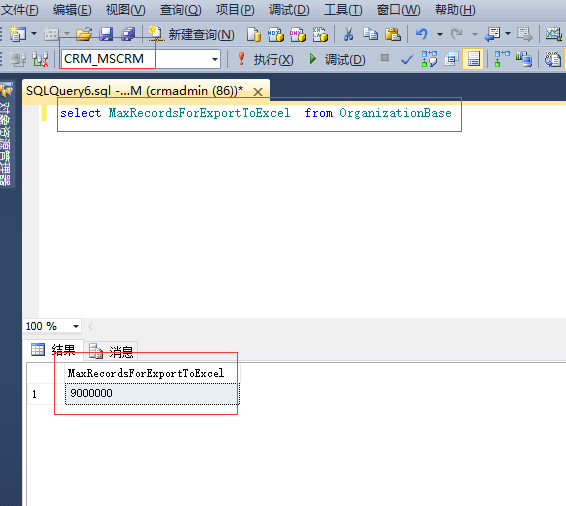
## 生成GUID



## 修改crm数据导出时的限制数量

在目标组织下查询，得到最大的到处记录数量。

select MaxRecordsForExportToExcel from OrganizationBase



然后sql更新这个值

Update OrganizationBase Set MaxRecordsForExportToExcel=65000 where Name = ‘OrganizationName’

If the database update still failed, then you should restart first then update the regedit if still not worked.