



FREE eBook

LEARNING dynamics-crm

Free unaffiliated eBook created from
Stack Overflow contributors.

#dynamics-

crm

Table of Contents

About.....	1
Chapter 1: Getting started with dynamics-crm.....	2
Remarks.....	2
Versions.....	2
Examples.....	2
Download Microsoft CRM SDK.....	2
Chapter 2: Bug - TurboForms Xrm.Page.data.save().then() Errors With ErrorCode "Null", Mess... 3	3
Examples.....	3
onChange() Calls save(), From Field That Is Invalid.....	3
Steps to Reproduce (Turbo Forms, CRM 2015.1, --> CRM 2016.2).....	3
Known Workarounds:.....	4
Chapter 3: Call Actions using Web API.....	5
Introduction.....	5
Examples.....	5
Call actions using Web API.....	5
Chapter 4: CRM 2013: How to hide unwanted Activity Types from the sub grid.....	8
Introduction.....	8
Examples.....	8
Add this function to a javascript web resource.....	8
Chapter 5: Using Web API with jQuery.....	11
Examples.....	11
Using the verbose option to get optionset and lookup values.....	11
Using select to reduce the number of fields.....	11
Using expand to get lookup properties.....	11
Getting accounts.....	12
Using filter to filter your API query.....	12
Chapter 6: Web API posts JSON examples.....	13
Remarks.....	13
Examples.....	13
Creating a note / annotation with attachment.....	13

Creating an account.....	13
Creating a contact with a parent customer.....	13
Creating a quote detail.....	14
Chapter 7: What Not to do when upgrading your Microsoft Dynamics CRM 2016.....	15
Introduction.....	15
Examples.....	15
Let's look at some mandatory items to check off before diving into a full-scale upgrade pr.....	15
Will the benefits of upgrading outweigh the headaches? Is an upgrade worth it?.....	16
Credits.....	21

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: [dynamics-crm](#)

It is an unofficial and free dynamics-crm ebook created for educational purposes. All the content is extracted from [Stack Overflow Documentation](#), which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official dynamics-crm.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with dynamics-crm

Remarks

Microsoft Dynamics CRM SDK allows developers to extend the Microsoft Dynamics CRM product, add new functionalities and meet requirements.

The SDK allows you to operate and communicate with the platform programmatically through web service messages, as well as to add custom code components like plug-ins, custom workflows and custom actions.

The Microsoft Dynamics CRM provides a JavaScript SDK library called Xrm on the client-side, which allows extending the user interface and experience.

Versions

Version	Sdk Version	Download link	Release notes link	Release Date
1.0	1.0			2003-01-01
3.0	3.0			2005-12-01
4.0	4.0	Download	Notes	2007-12-01
2011	5.0	Download	Notes	2011-02-01
2013	6.0	Download	Notes	2013-11-01
2015	7.0	Download	Notes	2014-11-01
2016 (365)	8.0	Download	Notes	2015-11-01

Examples

Download Microsoft CRM SDK

The latest SDK can be downloaded [here](#)

The latest SDK libraries are also available on NuGet under Microsoft's official [crmsdk](#) account

Read [Getting started with dynamics-crm online](#): <https://riptutorial.com/dynamics-crm/topic/1112/getting-started-with-dynamics-crm>

Chapter 2: Bug - TurboForms

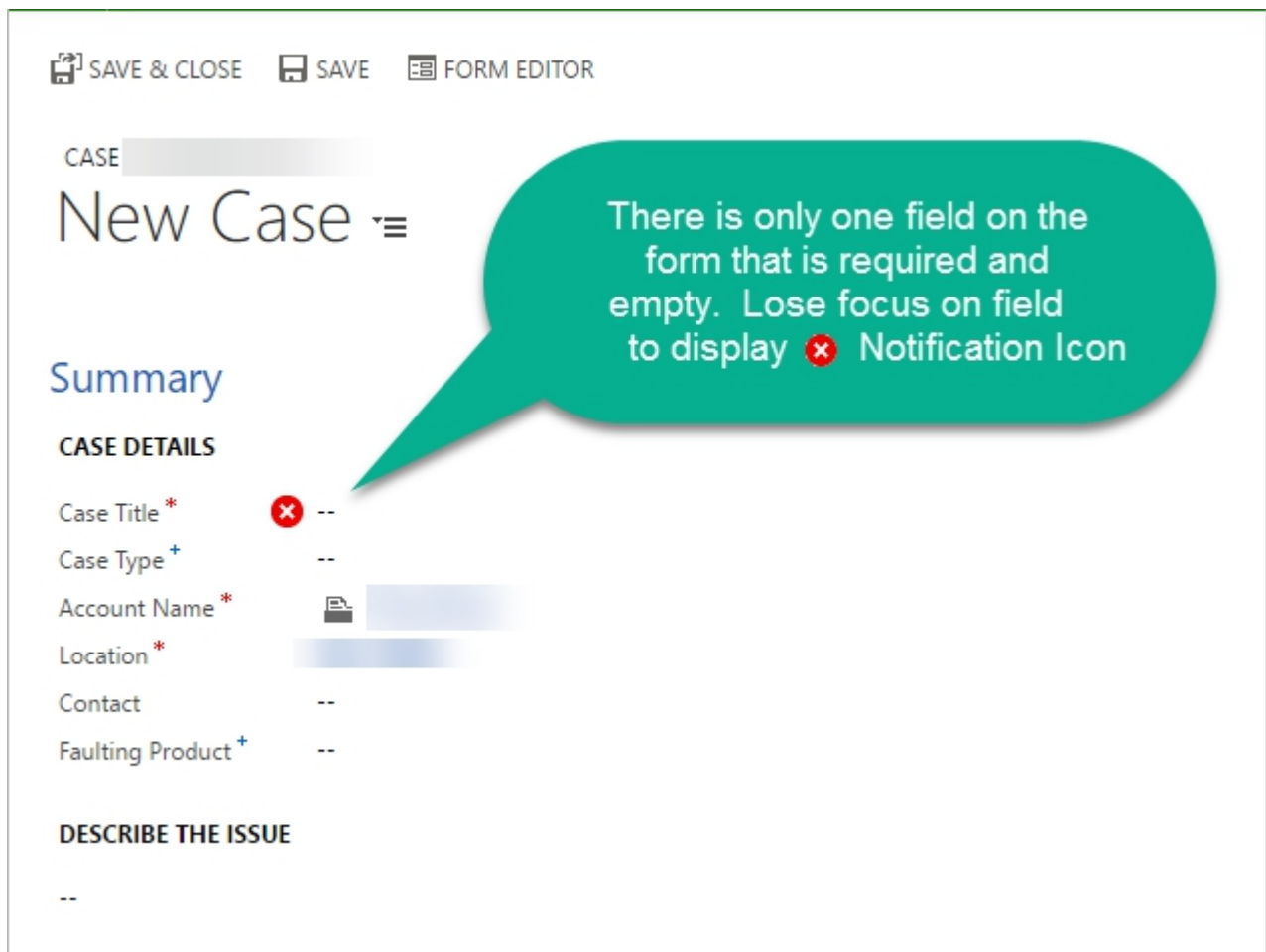
Xrm.Page.data.save().then() Errors With ErrorCode "Null", Message "Undefined"

Examples

onChange() Calls save(), From Field That Is Invalid

Steps to Reproduce (Turbo Forms, CRM 2015.1, --> CRM 2016.2)

1. Form (with or without other required fields) has one field that is empty and required.
2. Lose focus on the empty field ("title" in this example), this triggers the Field Notification Icon:



3. Wire up onChange Handler for empty field to call save:


```
function forceSaveOnChangeOfTitle(){
    Xrm.Page.data.save().then(
        function () {},
        function (error, message) {console.error("Error: " + error + " Message: " + message);}
    );
}
```

```
}
```

4. Enter a Value in empty field.

Result:

- Save fails. Calls failure callback with Error Number of "null", and Message of "undefined".
- Field Notification disappears, but required message is still displayed in the bottom right:

You must provide a value for Case Title. 

Known Workarounds:

Set the value of the attribute to itself:

```
function forceSaveOnChangeOfTitle(){
  var title = Xrm.Page.getAttribute("title");
  title.setValue(title.getValue());
  Xrm.Page.data.save().then(
    function () {},
    function (error, message) {console.error("Error: " + error + " Message: " + message);}
  );
}
```

Use 1ms Timeout with:

```
function forceSaveOnChangeOfTitle(){
  setTimeout(function() {
    Xrm.Page.data.save().then(
      function () {},
      function (error, message) {console.error("Error: " + error + " Message: " +
message);}
    );
  }, 1);
}
```

Read Bug - TurboForms Xrm.Page.data.save().then() Errors With ErrorCode "Null", Message "Undefined" online: <https://riptutorial.com/dynamics-crm/topic/4560/bug---turboforms-xrm-page-data-save---then---errors-with-errorcode--null---message--undefined->

Chapter 3: Call Actions using Web API

Introduction

Examples how to call bound and unbound actions.

Note that in a bound function the full function name includes the namespace Microsoft.Dynamics.CRM. Functions that aren't bound must not use the full name.

Examples

Call actions using Web API

```
function exampleCloseIncident(id, status){
    var parameters = {};
    var incidentresolution = {};
    incidentresolution["incidentid@odata.bind"] = "/incidents(" + id + ")";
    incidentresolution["@odata.type"] = "Microsoft.Dynamics.CRM.incidentresolution";
    parameters.IncidentResolution = incidentresolution;
    parameters.Status = status;

    callUnboundAction("CloseIncident", parameters, true, function(result){
        Xrm.Utility.alertDialog("Incident closed");
    });
}

function exampleQualifyLead(id){
    var payload = {
        "CreateAccount": createAccount,
        "CreateContact": createContact,
        "CreateOpportunity": false,
        "Status":3
    };

    callBoundAction("leads", id, "Microsoft.Dynamics.CRM.QualifyLead", payload, true,
function(result){
    Xrm.Utility.alertDialog("Lead qualified");
});
}

function callUnboundAction(actionname, payload, async, successCallback, errorCallback) {
    var req = new XMLHttpRequest();
    req.open("POST", encodeURI(getWebAPIPath() + actionname), async);
    req.setRequestHeader("OData-MaxVersion", "4.0");
    req.setRequestHeader("OData-Version", "4.0");
    req.setRequestHeader("Accept", "application/json");
    req.setRequestHeader("Content-Type", "application/json; charset=utf-8");
    req.onreadystatechange = function () {
        if (this.readyState === 4) {
            req.onreadystatechange = null;
            if (this.status == 200 || this.status == 204) {
                if (this.status == 200) {
                    var result = JSON.parse(this.response);
                }
            }
        }
    };
}
```



```

        if (successCallback) {
            successCallback(result);
        }
    } else {

        if(errorCallback) {
            errorCallback(this);
        }
        else{
            Xrm.Utility.alertDialog(this.statusText);
        }
    }
}
};

if (payload) {
    req.send(JSON.stringify(payload));
}
else {
    req.send();
}
}

function callBoundAction(entitysetname, id, actionname, payload, async, successCallback,
errorCallback) {
    var req = new XMLHttpRequest();
    req.open("POST", encodeURI(getWebAPIPath() + entitysetname + "(" + id + ")/" +
actionname), async);
    req.setRequestHeader("OData-MaxVersion", "4.0");
    req.setRequestHeader("OData-Version", "4.0");
    req.setRequestHeader("Accept", "application/json");
    req.setRequestHeader("Content-Type", "application/json; charset=utf-8");
    req.onreadystatechange = function () {
        if (this.readyState === 4) {
            req.onreadystatechange = null;
            if (this.status == 200 || this.status == 204) {
                if (this.status == 200) {
                    var result = JSON.parse(this.response);
                }

                if (successCallback) {
                    successCallback(result);
                }
            } else {

                if(errorCallback) {
                    errorCallback(this);
                }
                else{
                    Xrm.Utility.alertDialog(this.statusText);
                }
            }
        }
    }
};

if (payload) {
    req.send(JSON.stringify(payload));
}
else {
    req.send();
}
}

```

```

    }
}

function getClientUrl() {
    //Get the organization URL
    if (typeof GetGlobalContext == "function" &&
        typeof GetGlobalContext().getClientUrl == "function") {
        return GetGlobalContext().getClientUrl();
    }
    else {
        //If GetGlobalContext is not defined check for Xrm.Page.context;
        if (typeof Xrm != "undefined" &&
            typeof Xrm.Page != "undefined" &&
            typeof Xrm.Page.context != "undefined" &&
            typeof Xrm.Page.context.getClientUrl == "function") {
            try {
                return Xrm.Page.context.getClientUrl();
            } catch (e) {
                throw new Error("Xrm.Page.context.getClientUrl is not available.");
            }
        }
        else { throw new Error("Context is not available."); }
    }
}

function getWebAPIPath() {
    return getClientUrl() + "/api/data/v8.2/";
}

```

Read Call Actions using Web API online: <https://riptutorial.com/dynamics-crm/topic/9607/call-actions-using-web-api>

Chapter 4: CRM 2013: How to hide unwanted Activity Types from the sub grid

Introduction

I recently had to modify the Activity sub-grid to remove certain activity types from the add activity menu.

Note, this may not be a supported method on how to do this, but there is no documented supported way to so it, so I had to come up with a solution & this worked, in CRM 2013 anyway.

Examples

Add this function to a javascript web resource

```
var _activitiesGridName = '';
function SetupActivityGridOnload(gridName)
{
    var btnsToHide =
    [
        'AddserviceappointmentButton',
        'AddcampaignresponseButton',
        'AddappointmentButton'
    ];
    _activitiesGridName = gridName;
    setTimeout(function ()
    {
        //setting timeout beacuse subgid take some time to load after the form is loaded
        if (Xrm.Page != null && Xrm.Page != undefined)
        {
            //validating to check if the sub grid is present on the form
            var grid = Xrm.Page.getControl(_activitiesGridName);
            if (!grid)
            {
                // grid not loaded yet - call function again to recheck after timeout
                console.log('grid not loaded yet');
                SetupActivityGridOnload(_activitiesGridName);
            }
            else
            {
                // grid loaded now hide unwanted activity buttons
                var menuItem = null;
                var parentMenu = null;
                $.each(btnsToHide, function (i, val)
                {
                    menuItem = document.getElementById(val);
                    if (menuItem)
                    {
                        if (parentMenu == null)
                        {
                            // load parent node - if not already loaded
                            parentMenu = menuItem.parentNode;
                        }
                        console.log('removing menu item: ' + val);
                        parentMenu.removeChild(menuItem);
                    }
                }
            }
            else
            {
                // grid loaded now hide unwanted activity buttons
                var menuItem = null;
                var parentMenu = null;
                $.each(btnsToHide, function (i, val)
                {
                    menuItem = document.getElementById(val);
                    if (menuItem)
                    {
                        if (parentMenu == null)
                        {
                            // load parent node - if not already loaded
                            parentMenu = menuItem.parentNode;
                        }
                        console.log('removing menu item: ' + val);
                        parentMenu.removeChild(menuItem);
                    }
                }
            }
        }
    }, 1000);
}
```

```
        {
            console.log('menu not found: ' + val);
        }
    });
}
}, 2000);
}
```

Then call the function from the onload event adding the grid name as a parameter like so

Handler Properties -- Webpage Dialog

https://ccp327dev.rs.cocentrix.com/tools/formeditor/dialogs/handlereditor.aspx?appSolutionId=%7b3EAC

Handler Properties

Handler Properties

Details Dependencies

Library

Function *

Enabled

Parameters

Pass execution context as first parameter

Comma separated list of parameters that will be passed to the function

OK

https://ccp327dev.rs.cocentrix.com/tools/formeditor/dialogs/han Internet | Protected Mode: On

Read CRM 2013: How to hide unwanted Activity Types from the sub grid online:
<https://riptutorial.com/dynamics-crm/topic/9836/crm-2013--how-to-hide-unwanted-activity-types-from-the-sub-grid>

Chapter 5: Using Web API with jQuery

Examples

Using the verbose option to get optionset and lookup values

By default you will get the codes and id's for optionsets and lookups. If you want to get the label as well, you need to add an extra header to the call.

```
$.ajax({
  url: Xrm.Page.context.getClientUrl() + '/api/data/v8.0/contacts',
  headers: {
    'Accept': 'Application/json',
    'Prefer': 'odata.include-annotations="OData.Community.Display.V1.FormattedValue"'
  }
}).done(function (result) {
  $.each(result.value, function (key, value) {
    //sample to access a label
    var gendercodeLabel = value['gendercode@OData.Community.Display.V1.FormattedValue'];
    var gendercodeValue = value.gendercode;
  });
});
```

Using select to reduce the number of fields

For performance reasons you should minimize the number of fields you are requesting from the API. You can use the select property to do so.

This example fetches the name property of all accounts:

```
$.ajax({
  url: Xrm.Page.context.getClientUrl() + '/api/data/v8.0/accounts?$select=name',
  headers: {
    'Accept': 'Application/json'
  }
}).done(function (result) {
  $.each(result.value, function (key, value) {
    var lastname = value.primarycontactid.lastname;
  });
});
```

Using expand to get lookup properties

If you fetch a single record and when that record has a lookup, you can also fetch values of the lookup value using the expand option. This reduces the number of calls you need to make to the API.

The sample gets all accounts and the last name of the primary contact:

```
$.ajax({
```

```

url: Xrm.Page.context.getClientUrl() +
'/api/data/v8.0/accounts?$select=name,primarycontactid&$expand=primarycontactid($select=lastname)',

headers: {
  'Accept': 'Application/json'
}
}).done(function (result) {
  $.each(result.value, function (key, value) {
    var lastname = value.primarycontactid.lastname;
  });
});

```

Getting accounts

This sample fetches accounts using a jQuery ajax method. One thing to note is that you need to set the header in the call to make the work.

```

$.ajax({
  url: Xrm.Page.context.getClientUrl() + '/api/data/v8.0/accounts',
  headers: {
    'Accept': 'Application/json'
  }
}).done(function (result) {
  var accounts = result.value;
});

```

Using filter to filter your API query

You can use the filter property to retrieve a subset of values from CRM. In this example only the accounts where the company name equals CompanyName are returned.

```

$.ajax({
  url: Xrm.Page.context.getClientUrl() + '/api/data/v8.0/accounts?$filter=name eq
CompanyName',
  headers: {
    'Accept': 'Application/json'
  }
}).done(function (result) {
  var accounts = result.value;
});

```

Read Using Web API with jQuery online: <https://riptutorial.com/dynamics-crm/topic/2243/using-web-api-with-jquery>

Chapter 6: Web API posts JSON examples

Remarks

Be sure to add the following header to the post request. Otherwise the request will fail:

```
Content-Type: application/json
```

Examples

Creating a note / annotation with attachment

url: /api/data/v8.0/annotations

json:

```
{
  "isdocument": true,
  "mimetype": "text/plain",
  "documentbody": "dGVzdA==",
  "objectid_account@odata.bind" : "/accounts(c6da77b6-d53e-e611-80b9-0050568a6c2d)",
  "filename": "test.txt"
}
```

As the objectid can be almost every entity in CRM you need to define the entity with `_entity` name after objectid.

Creating an account

url: /api/data/v8.0/accounts

json:

```
{
  "name" : "New account"
}
```

Creating a contact with a parent customer

url: /api/data/v8.0/contacts

json:

```
{
  "firstname" : "New",
  "lastname" : "Contact",
  "parentcustomerid_account@odata.bind" : "/accounts(c6da77b6-d53e-e611-80b9-0050568a6c2d)"
}
```



```
}
```

As the `parentcustomerid` can be an account or contact you need to define the type of entity you want to set with `_entityname` after `parentcustomerid`.

Creating a quote detail

url: `/api/data/v8.0/quotedetails`

json:

```
{
  "productid@odata.bind": "/products(11c0dbad-91df-e311-b8e5-6c3be5a8b200)",
  "quoteid@odata.bind" : "/quotes(69b5e1ae-037f-e611-80ed-fc15b428dcdc)",
  "uomid@odata.bind" : "/uoms(73a5daea-6ddc-e311-a678-6c3be5a8c0e8)",
  "quantity": 1
}
```

Read Web API posts JSON examples online: <https://riptutorial.com/dynamics-crm/topic/6367/web-api-posts-json-examples>

Chapter 7: What Not to do when upgrading your Microsoft Dynamics CRM 2016

Introduction

Microsoft Dynamics CRM has evolved drastically over the past few years. There have been many updates and versions released, with a range of new features and improvements at each stage along the path. During the upgrade of a Dynamics CRM environment, there are a few points to bear in mind to ensure a hassle-free upgrade process.

Examples

Let's look at some mandatory items to check off before diving into a full-scale upgrade process

DB backup

Database backup is a must before starting any Dynamics CRM upgrade process. This is mandatory as to always have the option to rollback in case of any major roadblock.

Wrong Estimation

DO NOT underestimate the work involved in a CRM Upgrade process.

Audit your current Microsoft Dynamics CRM and identify third-party solutions in use. For these third party solutions, check their developer website for compatibility with your intended upgrade CRM version. Download the new solution and keep it ready to test after the migration process has been completed.

CRM Organization

DO NOT upgrade the CRM blindly.

As a first step, we need to prepare the CRM organisation to upgrade Microsoft Dynamics CRM.

Few tips to prepare an organisation

Make sure that your CRM environment satisfies the software and hardware component requirements

While upgrading CRM 4, delete records from listed table. (The system will try to delete all the records from the below tables when we migrate to CRM 2011, in case it fails then we have to manually delete the entries from these tables.) Records in the below-listed tables will result in poor performance of the system.

AsyncOperationBase

WorkflowWaitSubscriptionBase

BulkDeleteFailureBase

WorkflowLogBase

DuplicateRecordBase

WorkflowWaitSubscriptionBase

Install the latest rollup before upgrading CRM. For example, in order to upgrade to CRM 2013, the CRM 2011 Server must either be in Update Rollup 6, Update Rollup 14 or a later rollup before an upgrade can be considered. Else, while upgrading the Dynamics CRM environment, an error will be thrown.

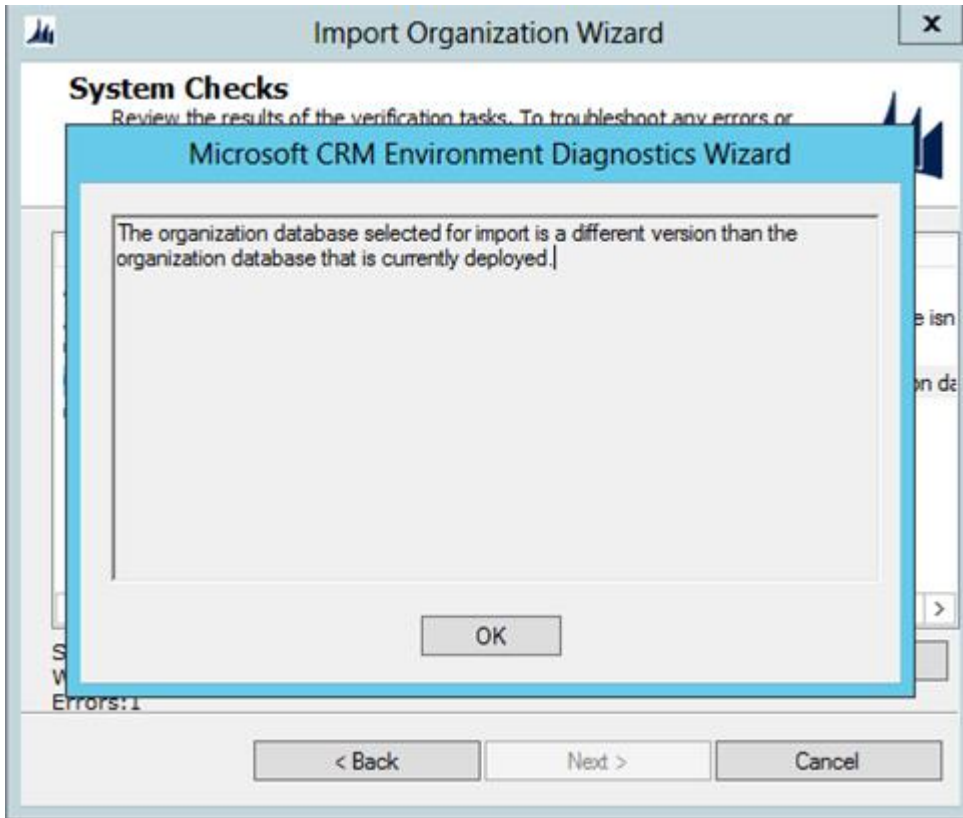
For upgrading to MS CRM 2013, use custom code validation tool to check for unsupported client side codes (JavaScript) that will not work following the upgrade. Also, use the legacy feature check tool to detect any server extensions that use the 2007 endpoint or Microsoft Dynamics CRM 4.0 features.

Each new version of Microsoft Dynamics CRM introduces more powerful functionalities, but along with the allure of the latest and greatest version of software, comes the concern of how the upgrade process will impact your business.

Will the benefits of upgrading outweigh the headaches? Is an upgrade worth it?

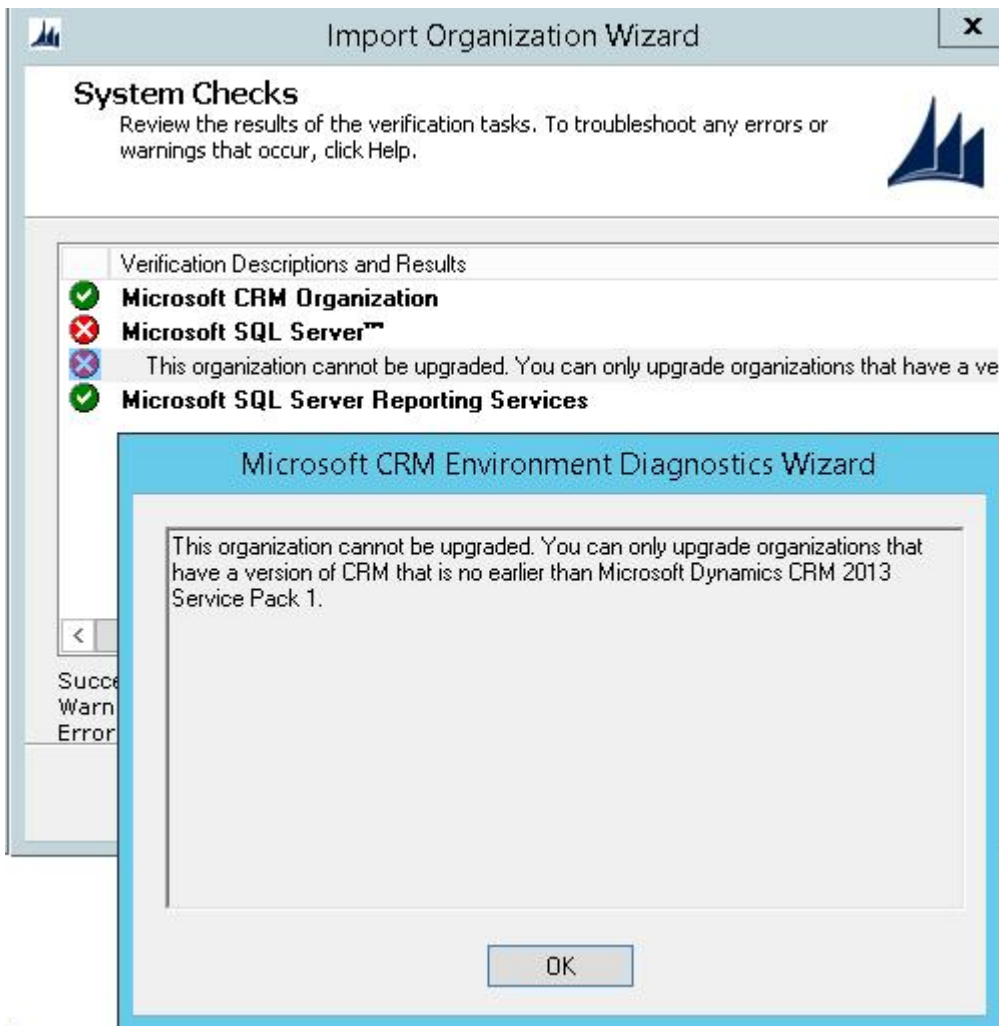
Now, here are some common pitfalls which might come along your way when upgrading your Dynamics CRM system.

*The organisation database selected for the import is a different version than the organisation database that is currently deployed*CRM Organization Database



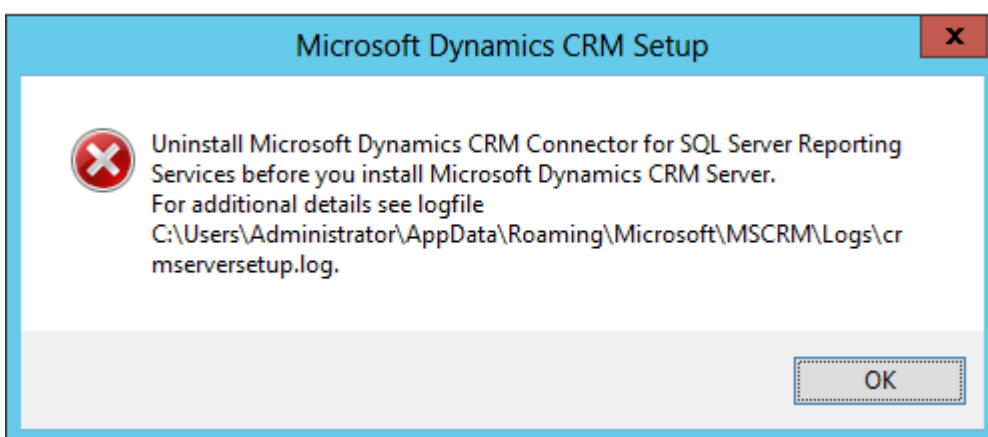
To fix this issue, we need to install either UR6 or above in MS CRM 2011 environment while upgrading to MS CRM 2013. Following this, the upgrade to Dynamics CRM 2016 can be completed without concerns

Organisation cannot be upgraded.



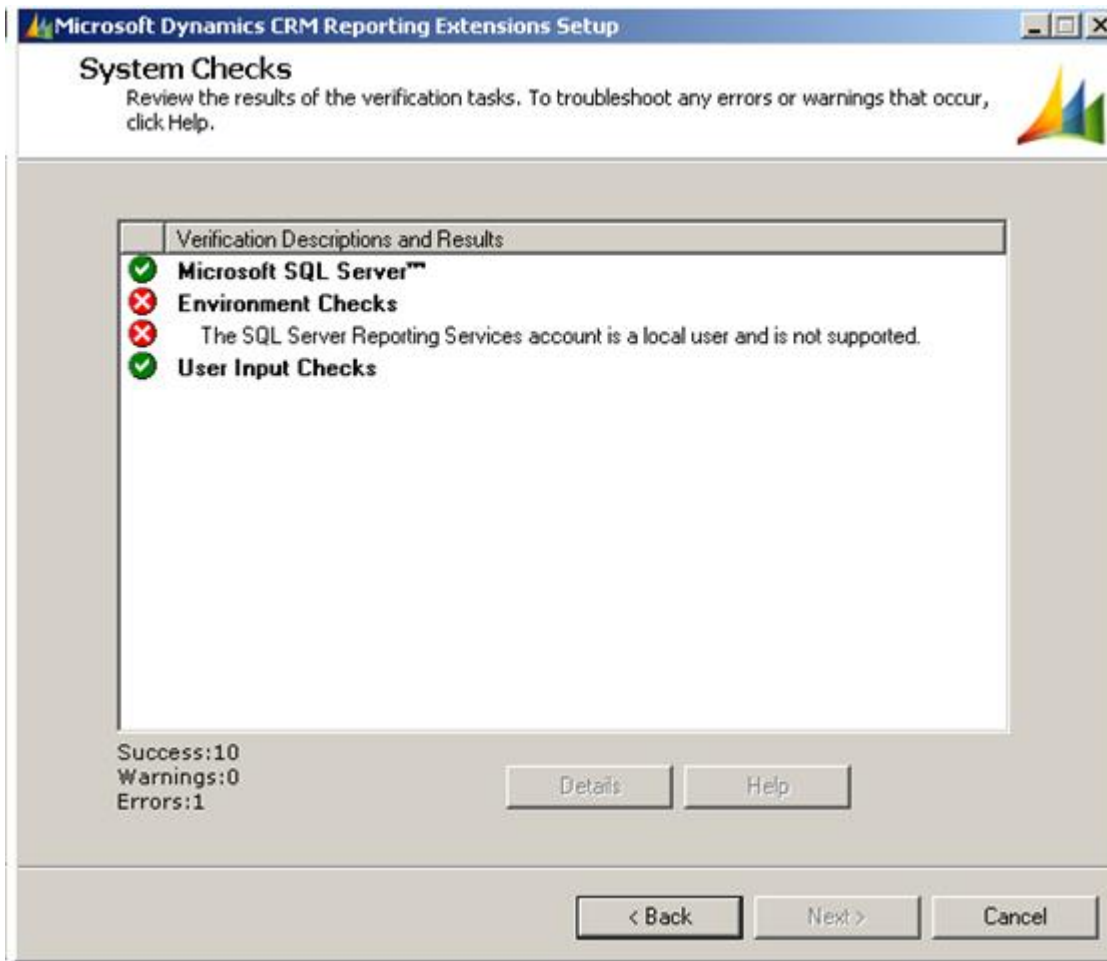
To fix this issue, we need to install SP1 for MS CRM 2013. Following this, the upgrade to Dynamics CRM 2016 can be completed without concerns.

Uninstallation of Microsoft Dynamics CRM connector for SQL Server reporting services

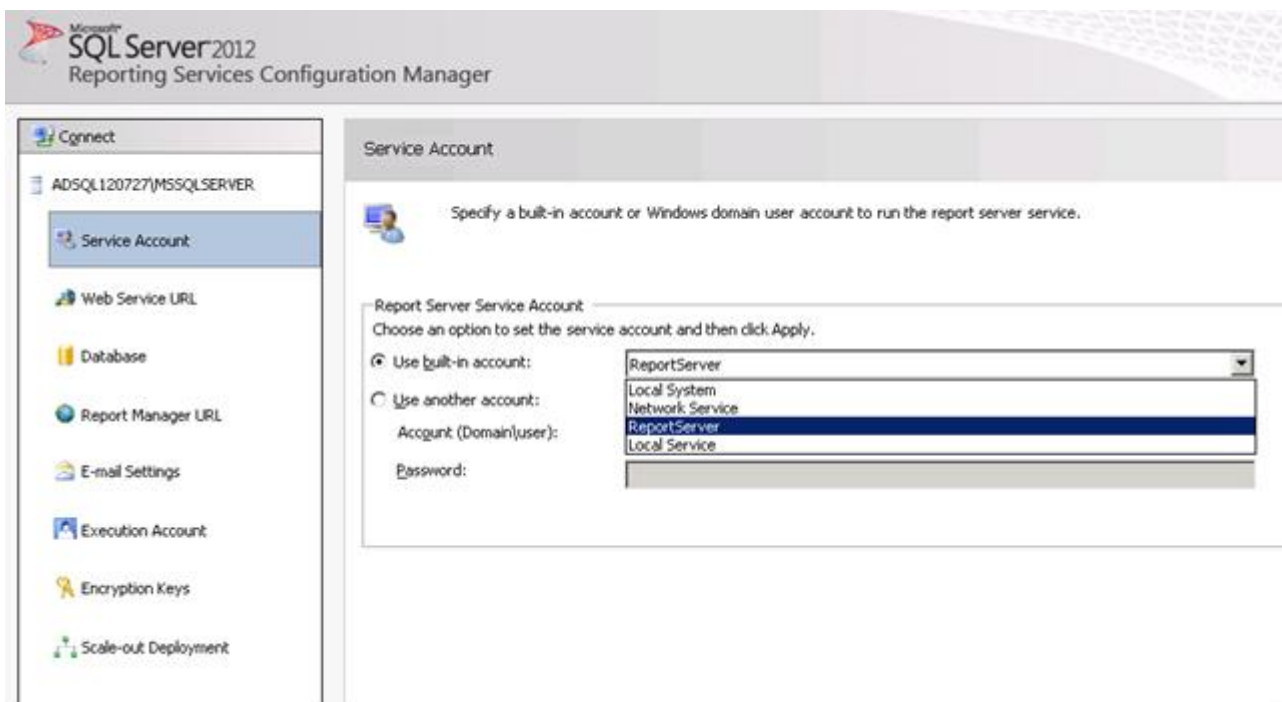


To fix this issue, uninstall Microsoft Dynamics CRM 2013 reporting extensions from Installed programs in Control panel before proceeding with the upgrade to Dynamics CRM 2016.

The SQL Server Reporting Service account is a local user and is not supported when upgrading from CRM 2013 to CRM 2015



If the Microsoft SQL Server 2012 Reporting Service was installed via default settings – then the service account is set to “ReportServer” – to resolve the issue, open the Reporting Services Configuration Manager and update the Service Account to anything else such as “Local System”. Following this, we won’t face any concerns while we upgrading to MS CRM 2016.



Some of our learning have been simple and logical, but many might forgo them.

While upgrading, don't go with as it is upgraded, there are many features that can cater to the organisation's current requirements which are implemented through customization. These can be modelled out of the box to fit the requirements with later versions.

Also, consider the communication/integration components with which the legacy or any existing system is connected with Microsoft Dynamics CRM. These are important links which should not be affected during an upgrade process, and when affected, can greatly impact productivity

Read [What Not to do when upgrading your Microsoft Dynamics CRM 2016 online](https://riptutorial.com/dynamics-crm/topic/9616/what-not-to-do-when-upgrading-your-microsoft-dynamics-crm-2016):

<https://riptutorial.com/dynamics-crm/topic/9616/what-not-to-do-when-upgrading-your-microsoft-dynamics-crm-2016>

Credits

S. No	Chapters	Contributors
1	Getting started with dynamics-crm	Arun Vinoth , Community , Daryl , Mehmet Seckin
2	Bug - TurboForms Xrm.Page.data.save().then() Errors With ErrorCode "Null", Message "Undefined"	Daryl
3	Call Actions using Web API	Arun Vinoth , Pawel Gradecki
4	CRM 2013: How to hide unwanted Activity Types from the sub grid	M Costa
5	Using Web API with jQuery	Arun Vinoth , Daryl , jcjr , Martijn Eikelenboom
6	Web API posts JSON examples	Arun Vinoth , Martijn Eikelenboom , winterfruit
7	What Not to do when upgrading your Microsoft Dynamics CRM 2016	Yuvaraj