Dataverse Toolbelt

Dev Kit

User Guide

Contents

[Extension Overview 2](#_Toc140078445)

[Dataverse Instance Connections 2](#_Toc140078446)

[Accessing saved connections 2](#_Toc140078447)

[Creating a new connection 3](#_Toc140078448)

[Create a Connection using the SDK Tooling Connector 4](#_Toc140078449)

[Create a connection saved and encrypted by the app 5](#_Toc140078450)

[Deploy Plugin Assemblies and Plugin Step Registrations 7](#_Toc140078451)

[Deploy Plugin Assembly and Plugin Types 8](#_Toc140078452)

[Create and manage SDK Message Processing Steps 9](#_Toc140078453)

[One Click Plugin Assembly Update 11](#_Toc140078454)

[Generate C# Constants for Data Model 12](#_Toc140078455)

[Deploy Web Resources 14](#_Toc140078456)

[Release Packages 15](#_Toc140078457)

[Create Release Package 16](#_Toc140078458)

[Add Data to Release Package 19](#_Toc140078459)

[Deploy Release Package 21](#_Toc140078460)

[Deploy Solution in Release package 23](#_Toc140078461)

[Deploy Records in Release package 26](#_Toc140078462)

[Query & Update Data 27](#_Toc140078463)

[Solution Template 29](#_Toc140078464)

[Template Console Project 31](#_Toc140078465)

[Plugin Project 31](#_Toc140078466)

[Test Project 32](#_Toc140078467)

[Web Resources Project 33](#_Toc140078468)

[Item Templates 33](#_Toc140078469)

# Extension Overview

This visual studio connects Visual Studio to Dataverse instance and provides a range of features to improve development and deployment tasks.

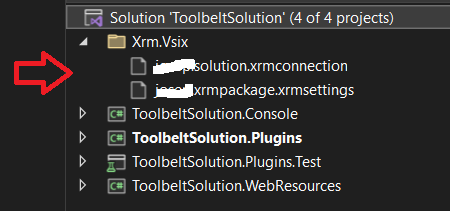
# Dataverse Instance Connections

The Dev Kit allows multiple instance connections to be saved in the package settings for a visual studio solution

The '’Active' connection (generally a development instance) will be used for the majority of features including code generation, code deployment, querying and updating of data, and creating release packages

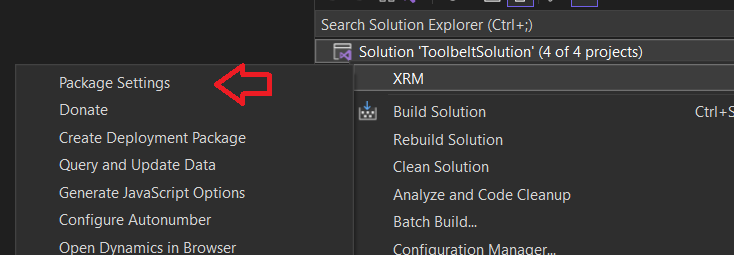
Other connections (generally test, UAT or production instances) will generally just be used to deploy release package into

Package Settings are saved per user by prefixing the settings file with usernames. This means connections are saved and encrypted separately for each user of the solution. The screenshot below shows the location of these saved settings within the visual studio solution

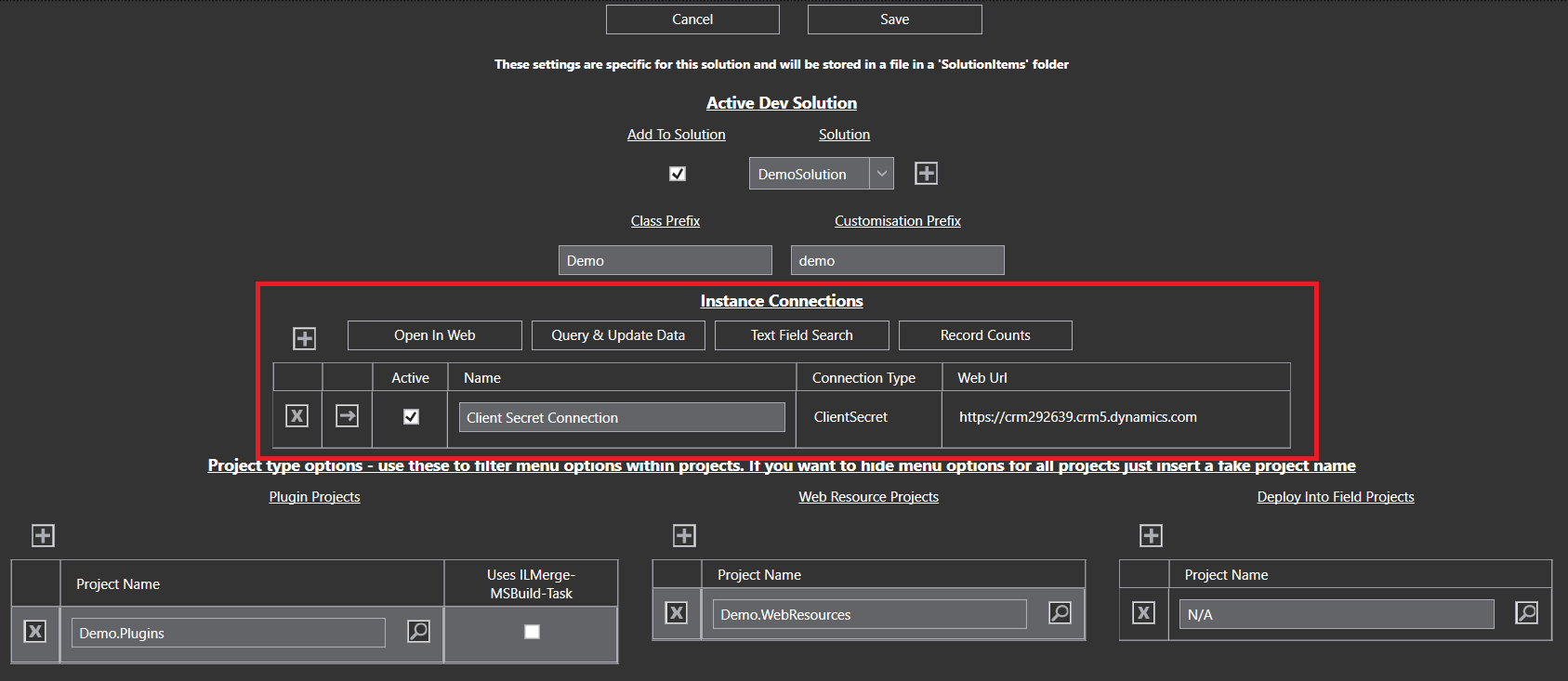


## Accessing saved connections

The screenshot below shows the menu location to access and create saved connections. Right click the solution in solution explorer and click XRM -> Package Settings in the flyout menu

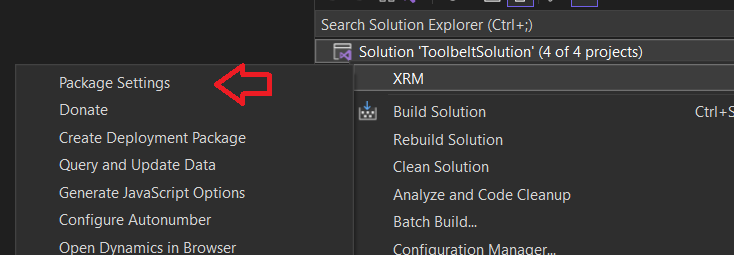
[](https://josephmcmac.files.wordpress.com/2022/11/image-58.png)

In the Package Settings form the Instance Connections grid list the saved connections

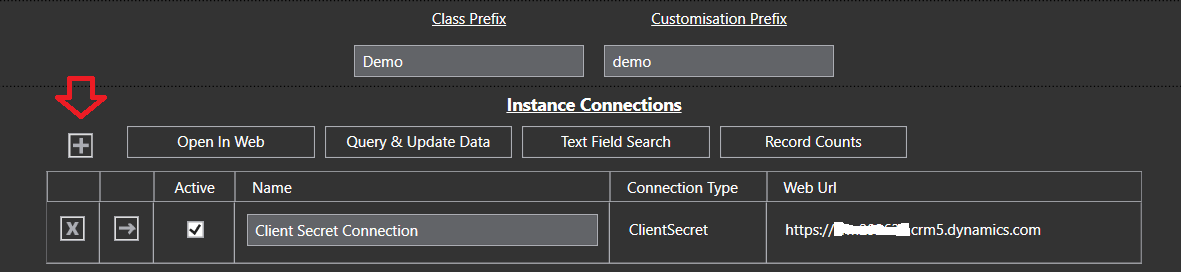


## Creating a new connection

The screenshot below shows the menu location to access and create saved connections. Right click the solution in solution explorer and click XRM -> Package Settings in the flyout menu

[](https://josephmcmac.files.wordpress.com/2022/11/image-58.png)

To add a new connection, click the + button at the top left of the Instance Connections grid



When creating a connection there are 2 options

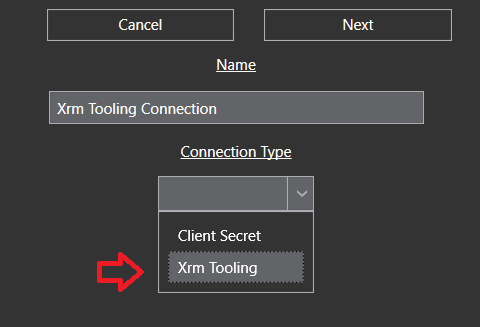
1. Create a connection using the Microsoft SDK Tooling Connector
2. Create a connection saved and encrypted by the app

Option 1 has the advantage of MFA support, and saves connections in your windows user accounts credentials managers  
<https://support.microsoft.com/en-us/windows/accessing-credential-manager-1b5c916a-6a16-889f-8581-fc16e8165ac0?WT.mc_id=DX-MVP-5004940#create-a-non-interactive-user-account>

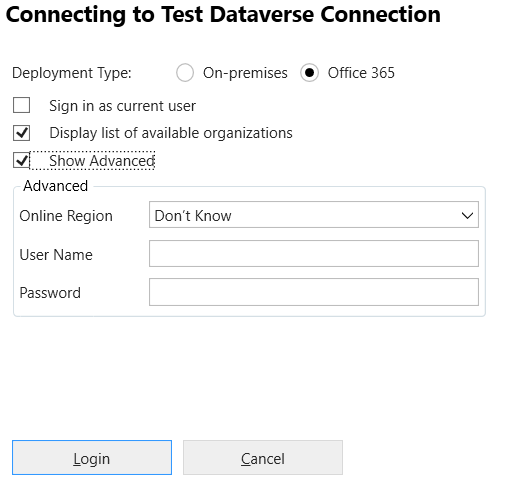
Option 2 is generally simpler to connect, but does not support MFA. It saves connection in a different form by storing them in the apps settings folder on disk encrypted with .NET libraries securely for the current user

## Create a Connection using the SDK Tooling Connector

To connect using the SDK connector select the Connection Type of Xrm Tooling.

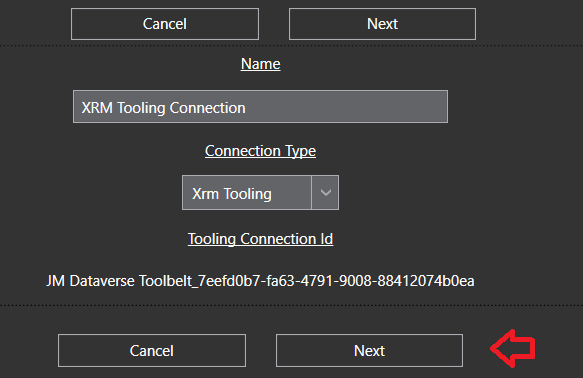


This will pop up a login dialog which is part of the Microsoft SDK assemblies. If connecting to a Microsoft online instance you’ll need to click Office 365 and Show Advanced, then enter username and password before hitting Login. This will then progress through the login including any subsequent MFA process which is required.



This dialog is managed by Microsoft assemblies using the SDK control documented at the link below  
https://learn.microsoft.com/en-us/power-apps/developer/data-platform/xrm-tooling/use-xrm-tooling-common-login-control-client-applications?WT.mc\_id=DX-MVP-5004940

Once the dialog is completed click Next to commit and save to your saved connections

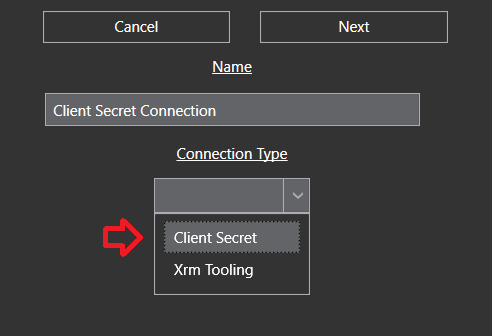


This new connection will be added to the Instance Connections grid in the Package Settings

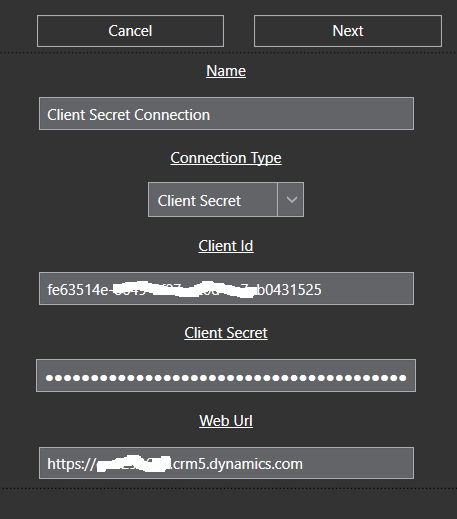
Note no connections added to the instance connections grid will be saved until the Save button is clicked. Click save in the Package settings form to ensure the new connection is persisted to the saved settings and set active

## Create a Client / Secret connection for an Application User

To create a client secret connection saved and encrypted by the app, select the Connection Type as Client Secret

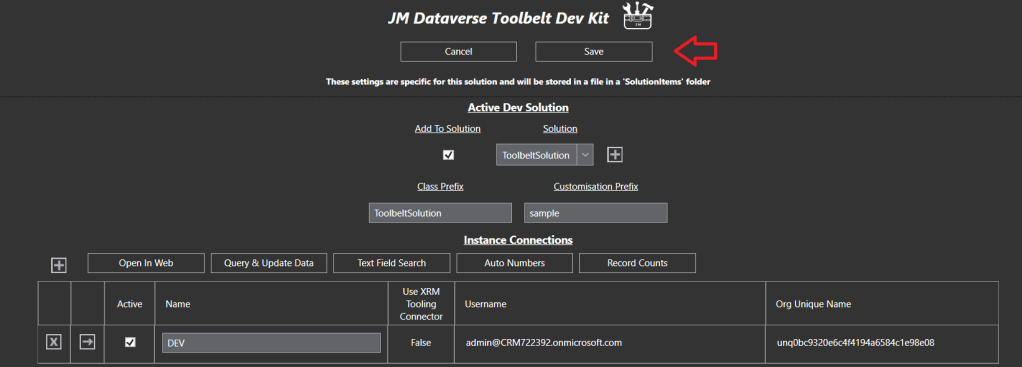


In this case client ID and client secret along with the instance web URL are entered directly into the connection form.



Once all details are entered click Next to commit and save to your saved connections. This new connection will be added to the Instance Connections grid in the Package Settings

Note no connections added to the instance connections grid will be saved until the Save button is clicked. Click save in the Package settings form to ensure the new connection is persisted to the saved settings and set active

[](https://josephmcmac.files.wordpress.com/2022/11/image-67.png)

## Deploy Plugin Assemblies and Plugin Step Registrations

The extension has several features to simplify deployment and configuration of custom plugin assemblies into the Dataverse platform

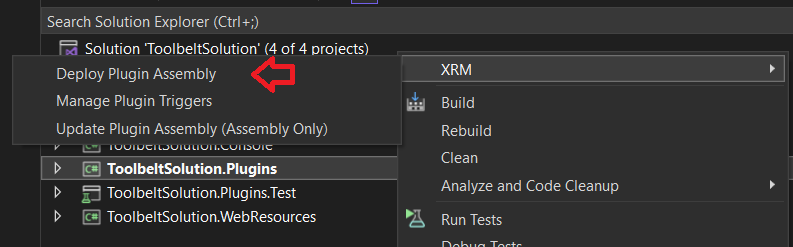
1. Deployment of Plugins Assembles along with creation of Plugin Type in the assembly
2. Create and Update of SDK Message Processing Steps for a plugin assembly
3. One-click Plugin Assembly update from the XRM menu in Solution Explorer

Each feature is further detailed in the following sections

### Deploy Plugin Assembly and Plugin Types

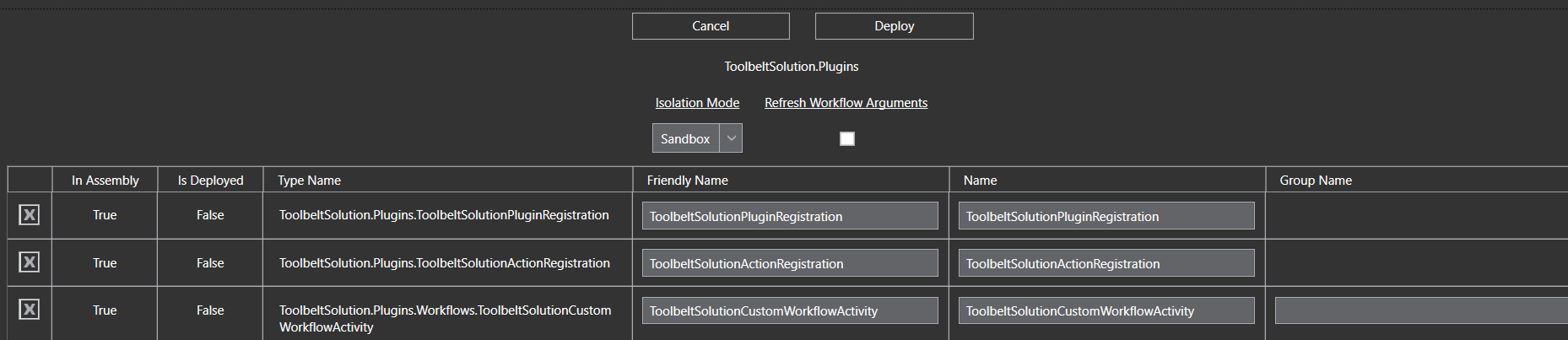
This feature of the extension deploys plugins assembles along its plugin types into the Datavere platform

The Deploy Plugin Assembly button is made available in the XRM menu when a project is right-clicked in solution explorer



When the Deploy Plugin Assembly process is initiated the extension will

1. Trigger a build of the plugin project
2. Verify the assembly is signed
3. Load the assembly and search for [Plugin](https://learn.microsoft.com/en-us/dotnet/api/microsoft.xrm.sdk.iplugin?WT.mc_id=DX-MVP-5004940) and [CodeActivity](https://learn.microsoft.com/en-us/dotnet/api/system.activities.codeactivity?WT.mc_id=DX-MVP-5004940) (Custom Workflow Activity) classes
4. Query the active instance for presence of matching [Plugin Assembly](https://learn.microsoft.com/en-us/power-apps/developer/data-platform/reference/entities/pluginassembly?WT.mc_id=DX-MVP-5004940) and [Plugin Type](https://learn.microsoft.com/en-us/power-apps/developer/data-platform/reference/entities/plugintype?WT.mc_id=DX-MVP-5004940) records
5. Load the form to confirm details of the assembly deployment



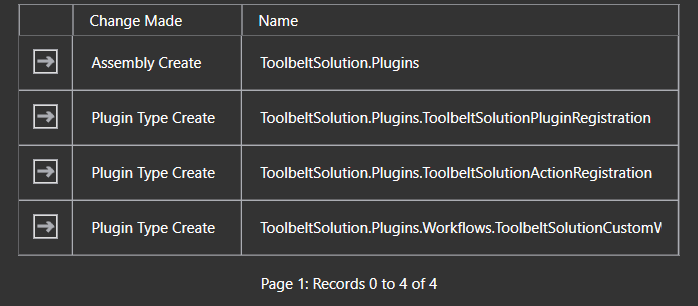
This form has deployment options including

1. Isolation Mode - change the Assemblies Isolation Mode to None (on premise only)
2. Refresh Workflow Arguments – if ticket the deployment process will set a field on each Plugin Type already deployed intended to refresh all Input and Output Argumenta based on the updated assembly classes. If not ticked where Input and Output arguments have changed the Plugin Types may not update adequately
3. Friendly Name - name used for plugins in the Dataverse platform
4. Name – other named used for plugins in the Dataverse platform
5. Group Name – menu group the custom workflow activity is displayed in when configuring in Workflows-

Click Deploy on the form and the extension will work through required data updates to deploy the assembly and types into the Dataverse platform



At completion a summery of updates will be displayed

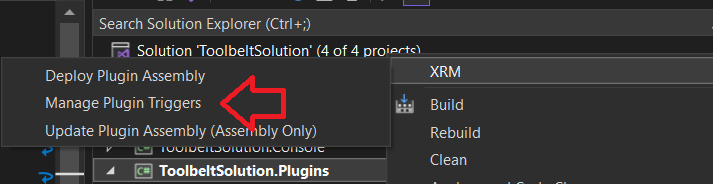


The Deploy Plugin Assembly process may be run again on the plugin project as required to add new plugins or workflow activities, delete the plugins or workflow activities, or update their configurations

### Create and manage SDK Message Processing Steps

This feature of the extension enables configuration of plugin triggers (SDK Message Processing Steps) for a plugin assembly in the Dataverse instance

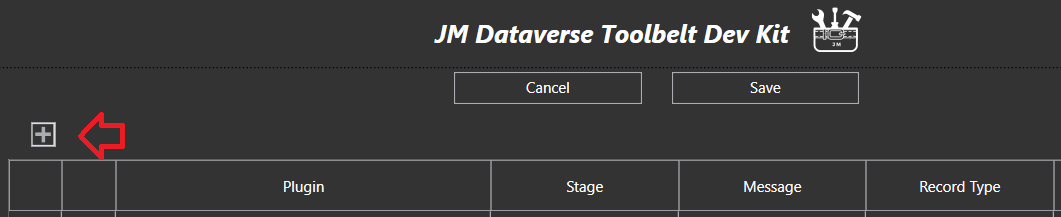
The Manage Plugin Triggers button is made available in the XRM menu when a project is right-clicked in solution explorer



When the Manage Plugin Triggers process is initiated the extension will

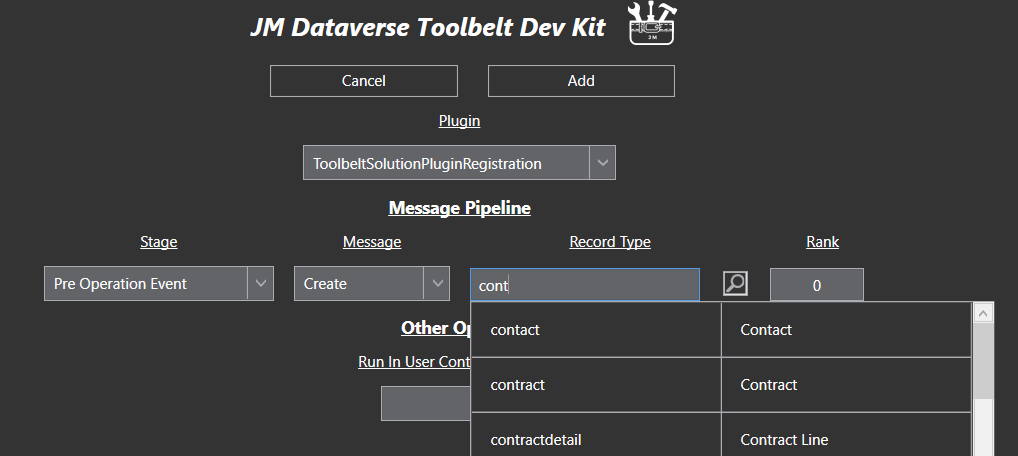
1. Verify there is a matching Plugin Assembly deployed to the active instance
2. Query [SDK Message Processing Steps](https://learn.microsoft.com/en-us/dynamics365/customerengagement/on-premises/developer/entities/sdkmessageprocessingstep?WT.mc_id=DX-MVP-5004940) in the active instance for that Plugin Assembly
3. Load the form for creating and managing the plugin triggers

To add a new plugin trigger (SDK Message Processing Step) click the + button at the top left of the grid

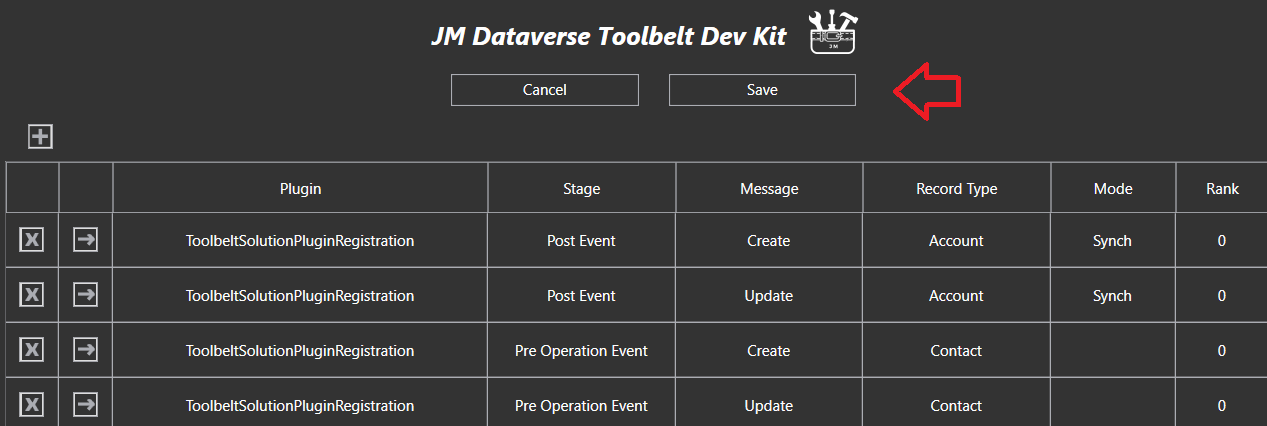


This loads the form for entering details for the new plugin trigger. The properties should be familiar for a learned Dataverse programmer, otherwise see Microsoft documentation for further details on Dataverse Plugins   
<https://learn.microsoft.com/en-us/power-apps/developer/data-platform/plug-ins?WT.mc_id=DX-MVP-5004940>

Note only PreImages are currently available for configuration in the extension as PostImages are considered an [anti-pattern](https://en.wikipedia.org/wiki/Anti-pattern) by the publisher, and the Target and PreImage in plugins is used to achieve the same result



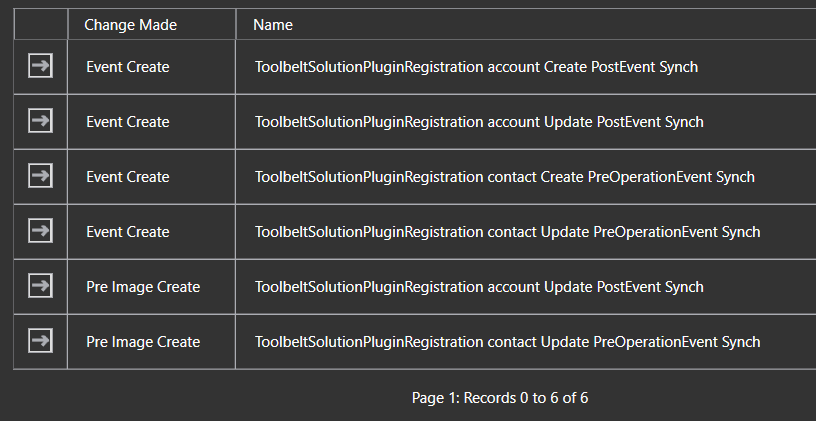
Once all required SDK Message information has been entered click Save to commit the changes into the Dataverse instance for the active connection



Progress information will be displayed as changes are save



At completion a summary of updates will be displayed

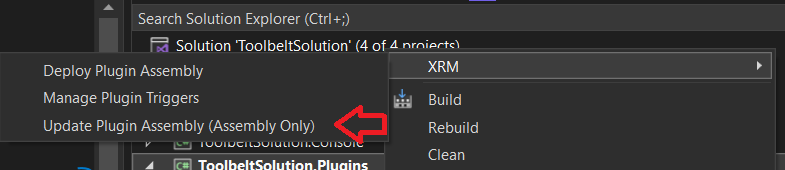


The Manage Plugin Triggers process may be run again on the plugin project as required to add new plugin triggers, delete plugin triggers, or update plugin trigger configurations

### One Click Plugin Assembly Update

This well-used feature of the extension deploys plugin code changes into the Dataverse instance with one-click from the XRM menu in solution explorer

The Update Plugin Assembly button is made available in the XRM menu when a project is right-clicked in solution explorer



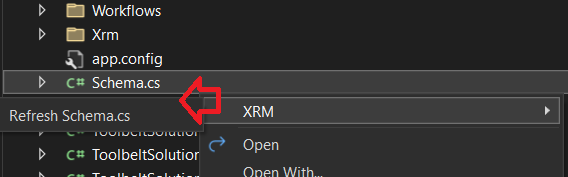
When clicked the process initiates a build of the project assembly, and if successful immediately updates the assembly binary into the Dataverse platform

No entry form or confirmation is needed with this feature to enable efficiency when developing plugin code for the platform

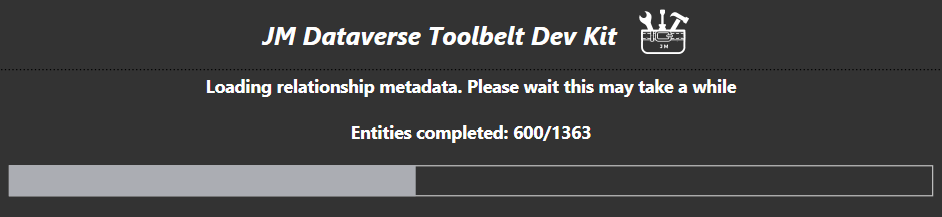
## Generate C# Constants for Data Model

This is another well-used feature of the extension which generates static classes and constants for the types, fields, option sets, and action endpoints which are configured in the Dataverse instance

The Refresh Schema button is made available by right clicking a file which has the name “Schema.cs” in solution explorer



Running this process will refresh the schema.cs file with classes and constants based on the current state of the Dataverse instance. The refresh may take a while as all metadata is refresh from the source instance to ensure customisations are up to date



Classes in the generated file enable greater efficiency writing code with the Dataverse SDK, by using the Intellisense feature of Visual Studio, instead of manually typing strings for customisation values

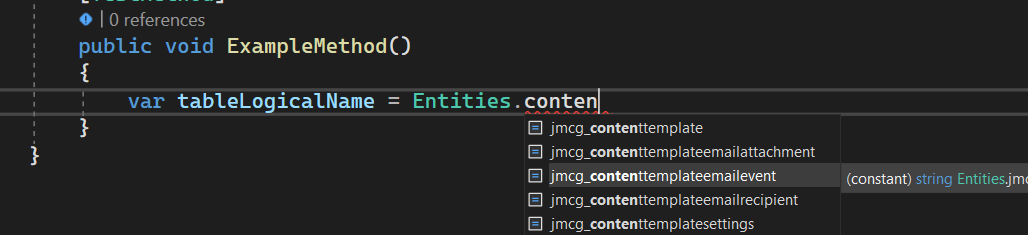
Code will also naturally become more maintainable as constants have managed code references, option values are represented as labels, as well as other benefits of using constants instead of hard values

Constants types generated include the following

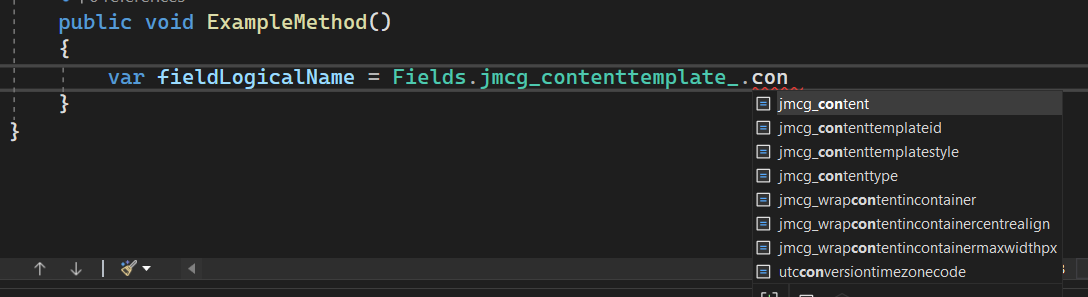
* Table logical names
* Column logical names
* Field option sets
* Shares option sets
* Custom actions along with their input and output arguments
* Many-to-many relationship logical names

The screenshots below show example constants and type ahead from the Schema.cs file

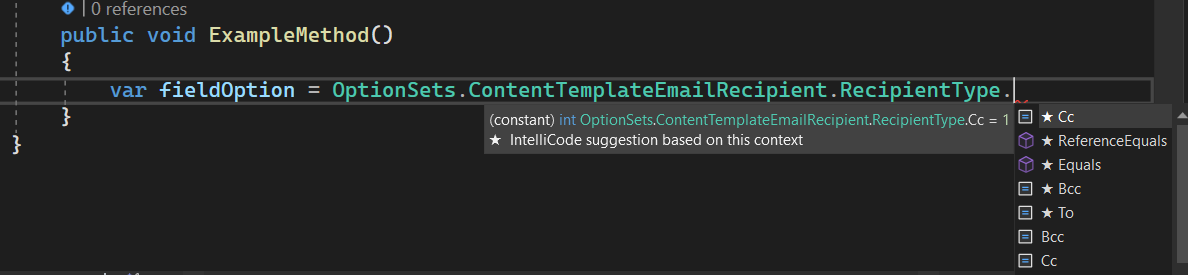
Constants in the static Entities class for table logical names



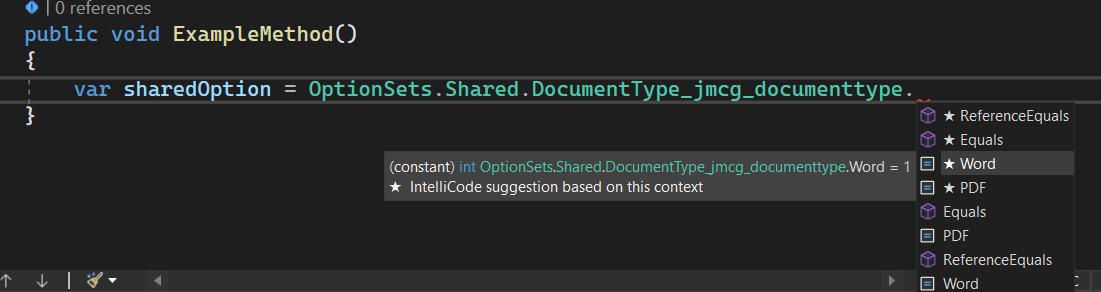
Constants in the static Fields class for column logical names



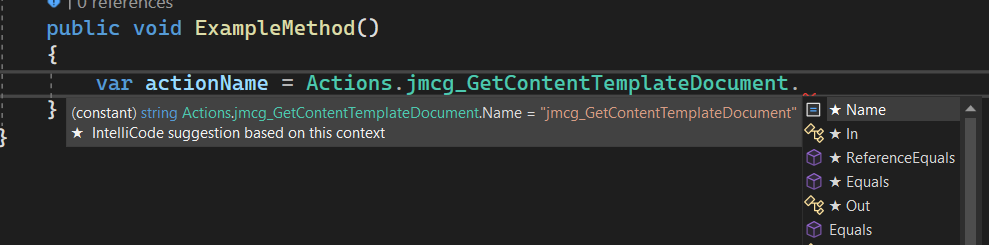
Constants in the static OptionSets class for column option values



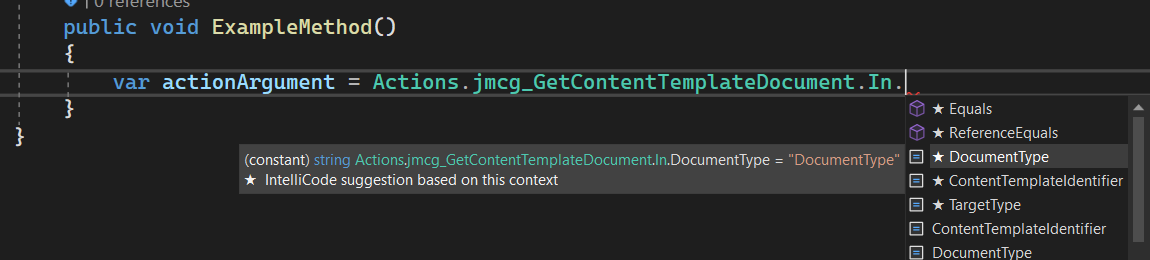
Constants in the static OptionSets.Shared class for shared option values



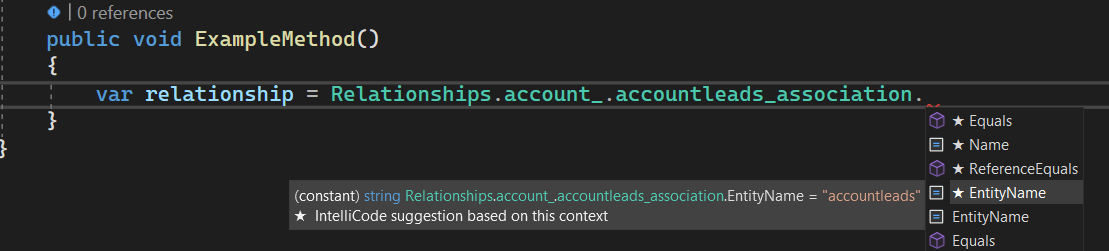
Constants in the Action class for custom action names



Constants in the static Actions class for custom action inout and output arguments



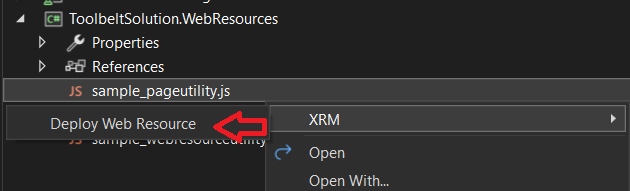
And finally in the static Relationships class many-to-many relationship names and intersect table names



## Deploy Web Resources

This feature of the extension uploads files directly from solution explorer iinto Dataverse web resources. It is a very useful for efficiently creating and updating JavaScript or other web resource files into the platform without having to upload manually in the web UI

The Deploy Web Resource button is made available when a compatible file type is right clicked in solution explorer



This will run a process to import the selected file(s) into Web Resources



Matching to existing web resources is done by unique name, else display name. That is

* If an existing web resource has the same unique name as the file it is updated with the contents
* Else if an existing web resource has the same display name as the file it is updated with the contents
* Else a new Web Resource is created

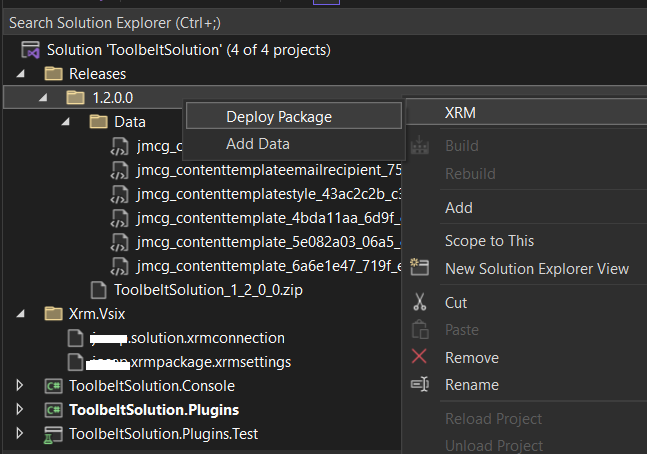
Web Resources with paths in their name are not supported by the feature

The following lists file type extensions are enabled for the Deploy Web Resource feature

* css
* gif
* htm
* html
* ico
* jpeg
* jpg
* js
* png
* xap
* xml
* xsl

## Release Packages

The release package features facilitates creation and deployment of release packages within a visual studio solution. Release package folders are created in the visual studio solution with Dataverse solution zip files, and XML files for table data to include in the deployment



The features provide a quick and simple way to package solutions and data for release. Along with efficiency in moving solutions and configuration data between environments, source control integration in Visual Studio means the release package files are safely persisted for future releases when moving through multiple environments

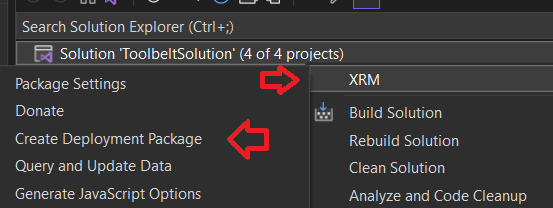
The release package features composed of the following

* Create Release Package – this will export a solution from the active connection into a solution zip file. Configuration data may optionally be included and saved as XML files in a release subfolder
* Add Data to Release Package – this feature enables quickly appending or updating configuration data in the release package
* Deploy Release package – this feature does a big bang deployment of sequentially importing all solution and configuration data from the release package into a saved connection
* Deploy Solution – this feature enables importing a specific solution from a release package into a saved connection. This is an alternative to the big bang deployment where release tasks for solutions and configuration data need to be done in more of a controlled sequential order
* Deploy Records - this feature enables importing specific configuration data from a release package into a saved connection. Again this is an alternative to the big bang deployment where release tasks for solutions and configuration data need to be done in more of a controlled sequential

### Create Release Package

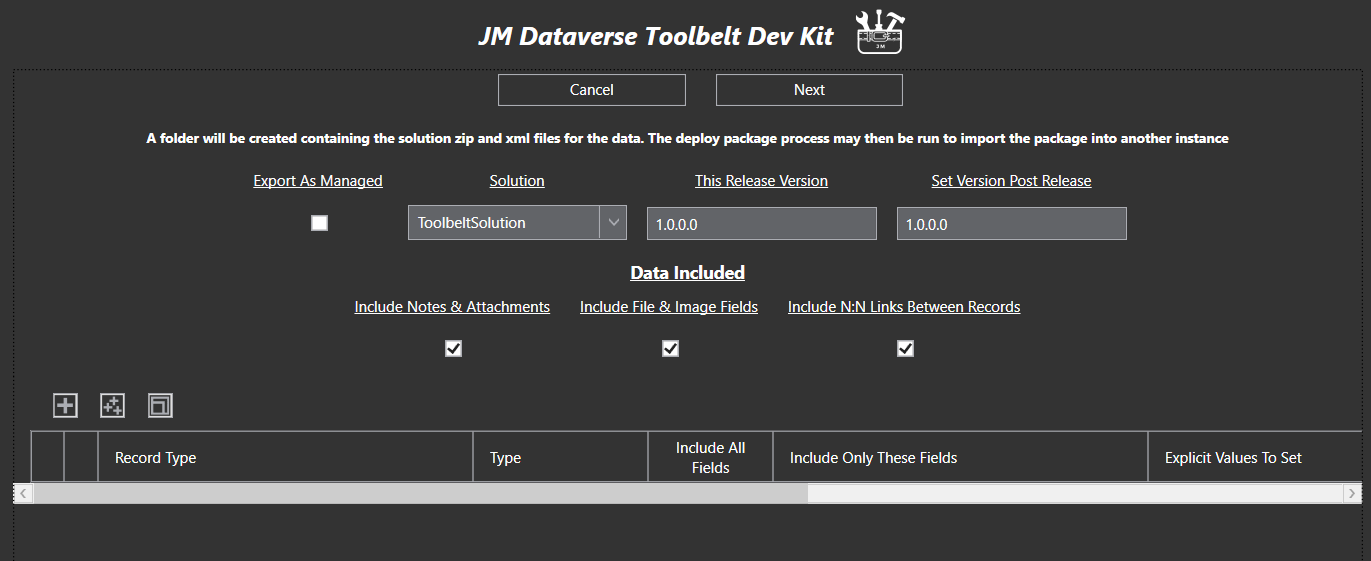
This feature is the main process for creating a release package in a visual studio solution

The Create Deployment Package button is made available by right clicking a solution in solution explorer, then locating it in the XRM submenu



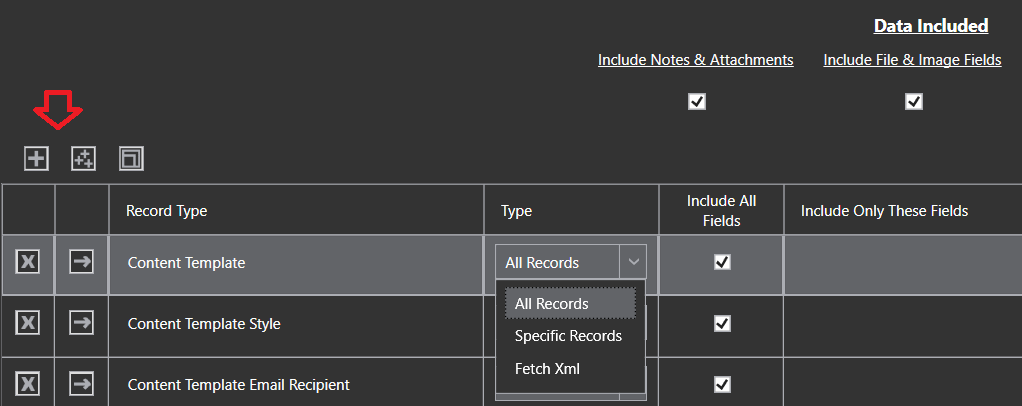
The Create Deployment Package form has main options for the solution to add into the package, then further down a grid with options for inclusion of data along with the release

* Setting This Release Version will set the selected solution version prior to the export
* Setting Set Version Post Release will update the solution version after export

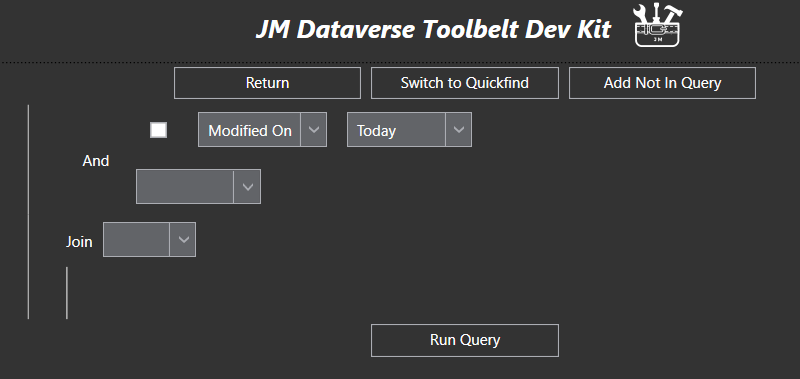


To include data in the package click the + button at the top left of the grid, or use the multi + button to select and add multiple types in one go

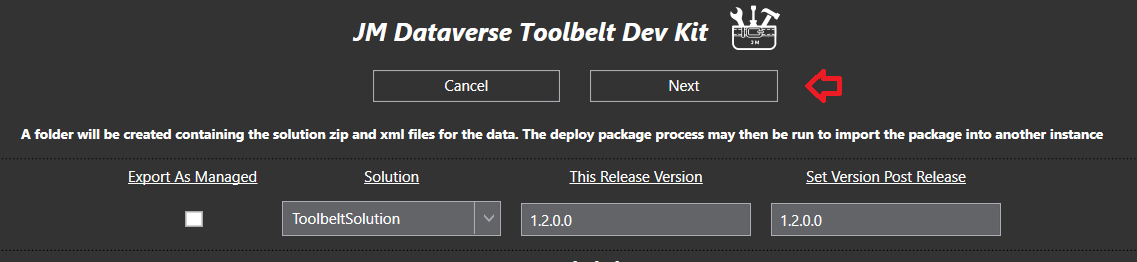
Once types are selected for inclusion the set of table rows and columns included in the package can be filtered. Change the Type selection from All Records to limit records included, and deselect Include All Fields to select a limited columns in each record



Where Specific Records is selected for the type, the extensions query dialog will be opened for selecting which specific records to include



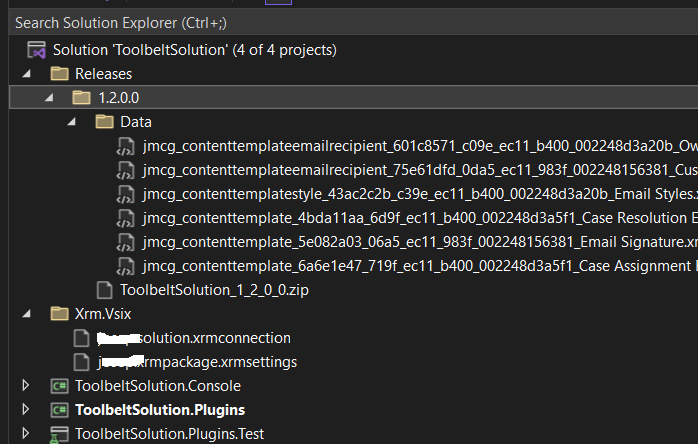
Once satisfied with the solution details and data to include, click Next to initiate creation of the release package



The process may take a minute to export the release package depending on performance of the instance



This screenshot shows an example release folder created by the process. Note the folder has been named according to the solution version, and there are xml files in the Data folder for each record which has been included in the package

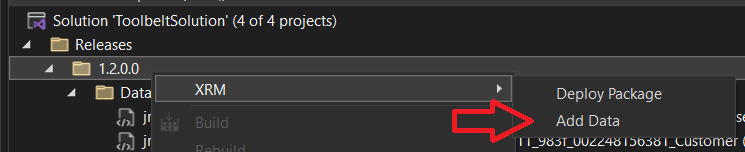


The Create Deployment Package process may be run any number of times to update the solution and data in the package

As is often the case additional solutions, files, and release notes need to be included in a holistic release package. There is no issue in adding additional folders and files into the release version folder to keep these all together. This won’t break any deployment process unless zip files are added, or incorrect xml files are added to the Data folder

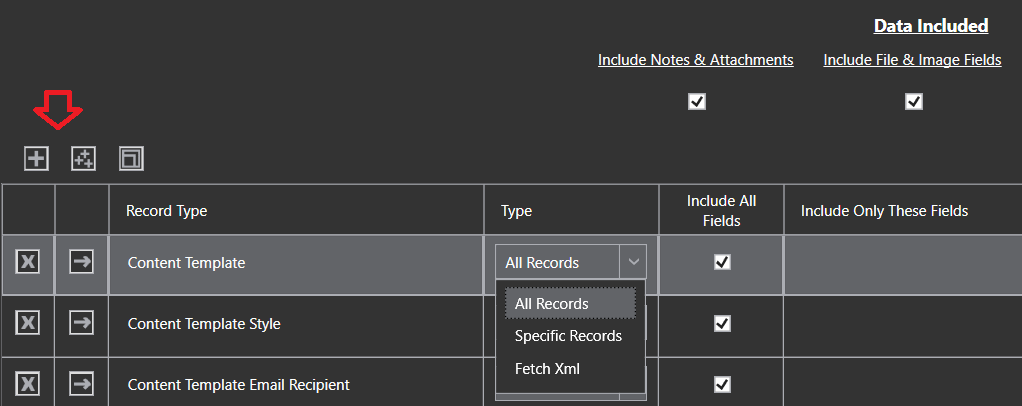
### Add Data to Release Package

This micro feature “Add Data” can be used to add additional data into a release package. This button is made available by right clicking the release version folder, and is particularly useful if where changes only involve configuration data and you want to append them into an existing release package, without the overhead of waiting for a full solution export

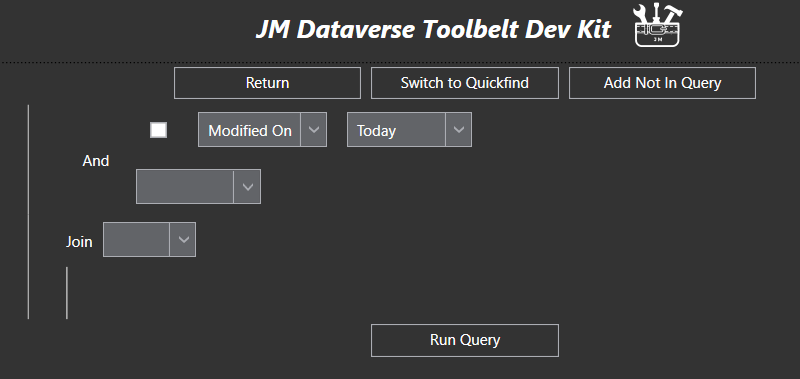


To include data in the process, click the + button at the top left of the grid, or use the multi + button to select and add multiple types in one go

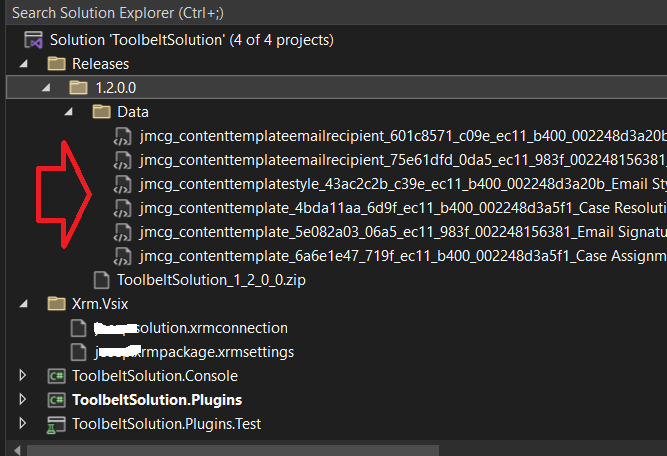
Once types are selected for inclusion the set of table rows and columns included can be filtered. Change the Type selection from All Records to limit records included, and deselect Include All Fields to select a limited columns in each record



Where Specific Records is selected for the type, the extensions query dialog will be opened for selecting which specific records to include



Once satisfied with the included data click Next to initiate the process. Once completed the included data will be appended or overwritten into any records already present in the release folder

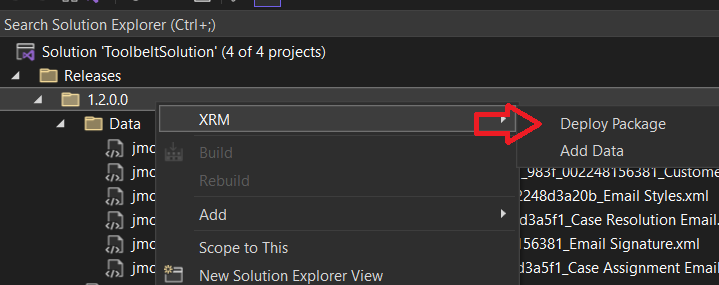


### Deploy Release Package

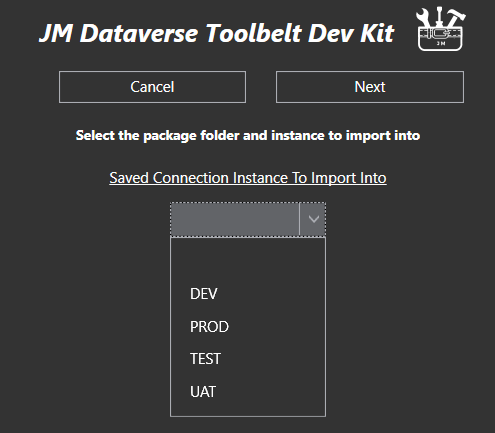
This feature of the extension Deploys a release package into a saved connection instance

Depending on preference items in the release package may be deployed selectively, or as a big bang deployment process. If selective deployment is preferred which enables importing the solutions and records separately see details for the Deploy Records and Deploy Solution features

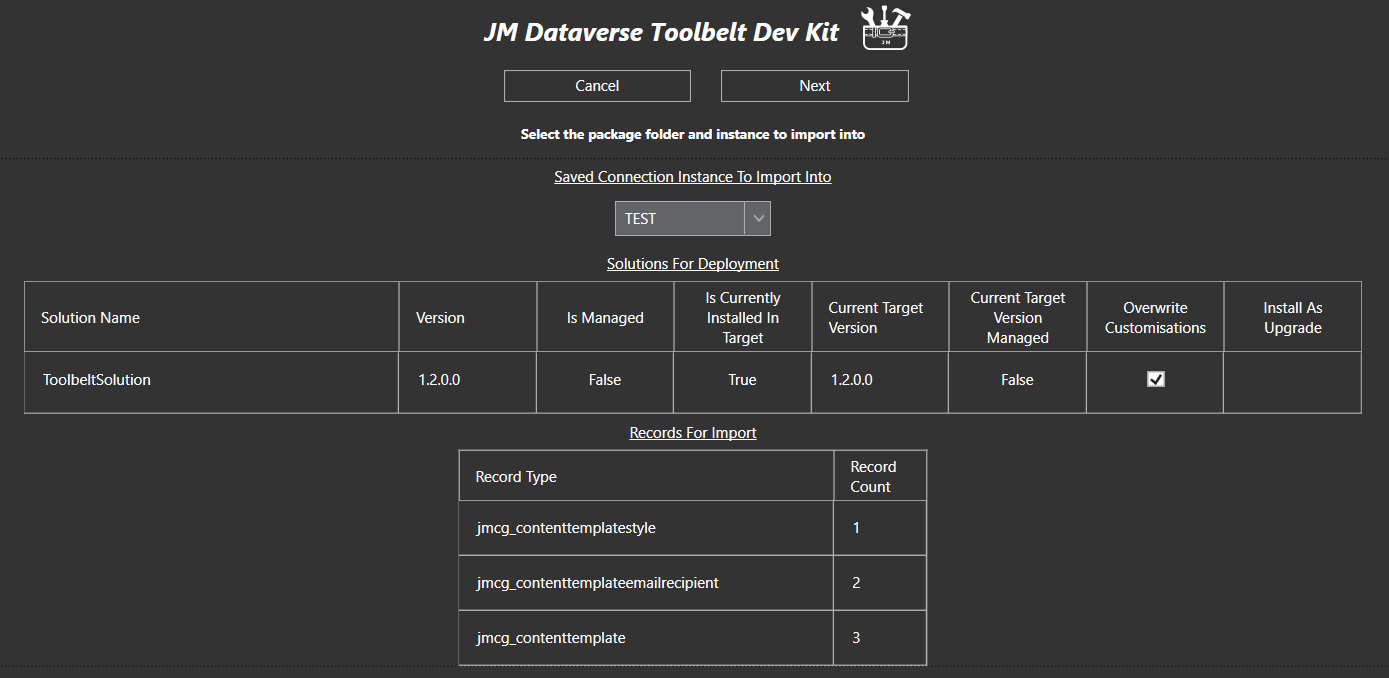
The Deploy Package button is made available by right clicking the release folder in solution explorer



The Deploy Package form will first prompt for the saved connection to deploy into

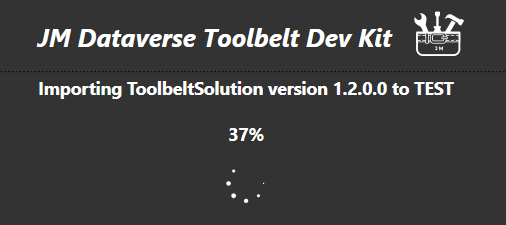


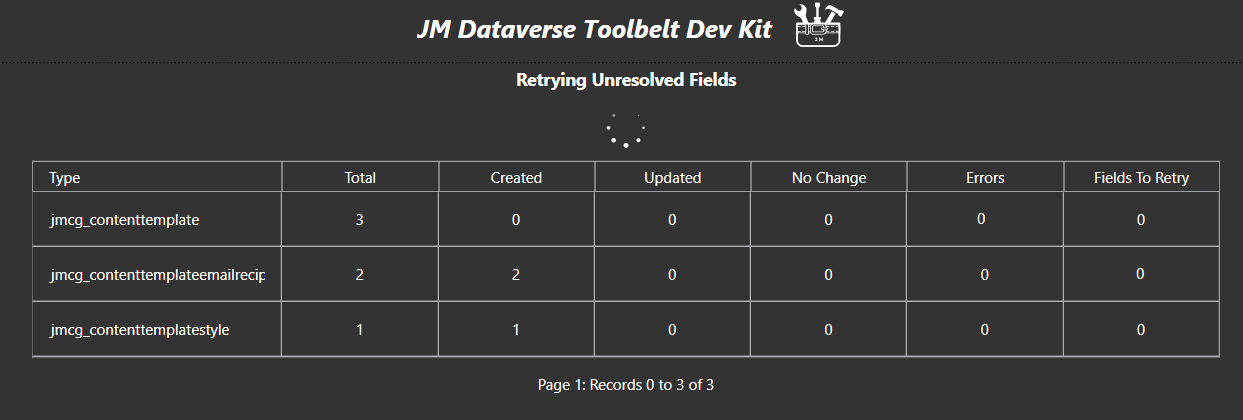
And when selected will display details of the loaded package, along with options for the solution import. Different options will be displayed for each solution in the package depending on it being unmanaged or managed, and what version if any is installed in the target instance

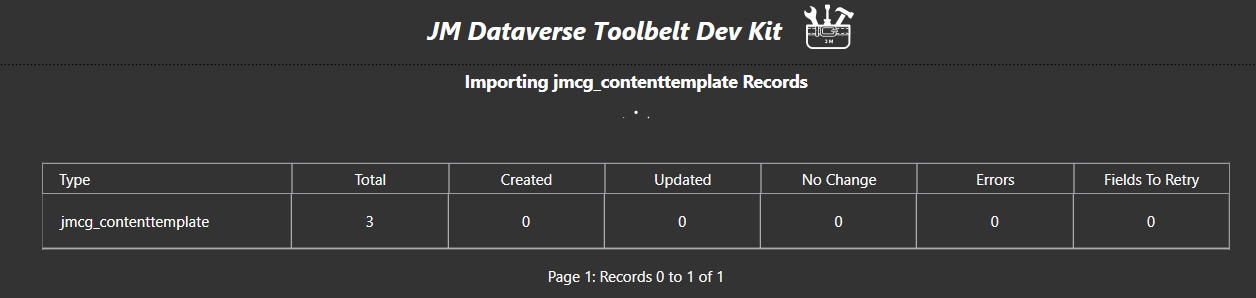


Click Next to initiate the deployment. The extension will display progress of deployment before a summary at completion





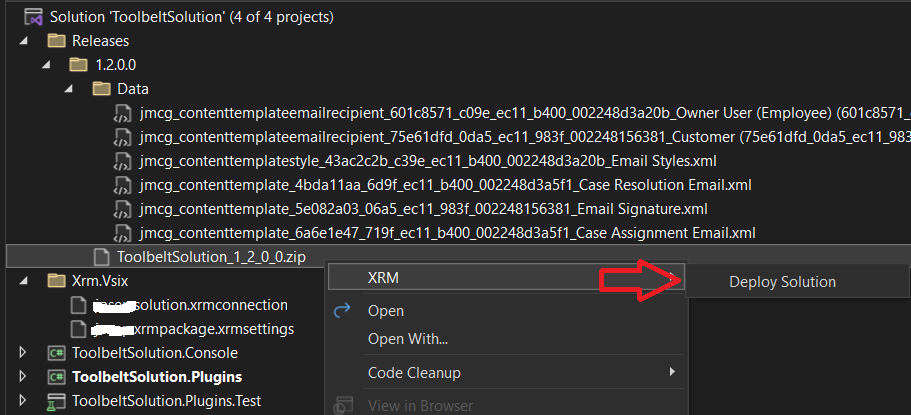




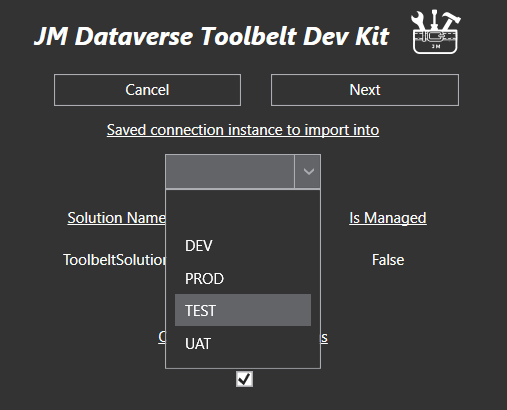
### Deploy Solution in Release package

This Deploy Solution feature of the extension enables import of solutions from a release package independent to other items in the release package

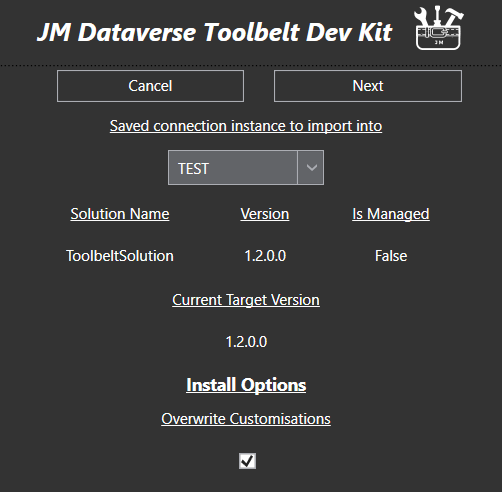
To run the Deploy Solution process right click a solution in solution explorer then click select Deploy records in the XRM menu



First select the connection to import into on the Deploy Solution form

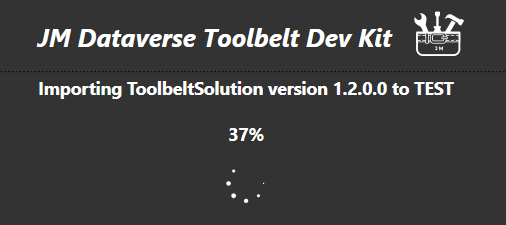


Different options will be displayed for the import depending on it being unmanaged or managed, and what version if any is installed in the target instance. When selections are completed select Next to initiate the import

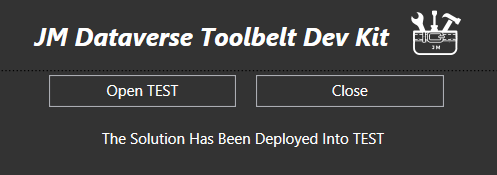


Regular progress will be displayed during the import process





And at completion a useful link to open the instance the solution was imported into

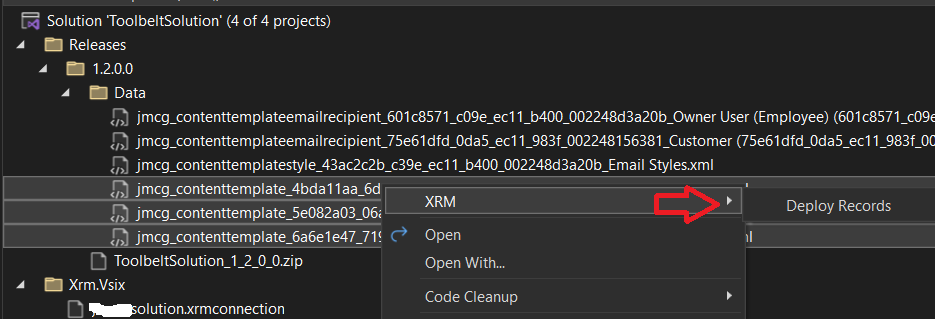


### Deploy Records in Release package

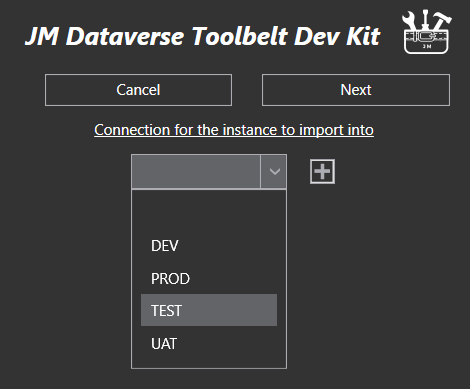
This Deploy Records feature of the extension enables deployment of release package data in a more granular fashion

Often when performing releases data needs to be installed in a particular sequence, and other tasks may need to be performed between each item being installed. When this is the case, the bug bang approach of Deploy Package is not feasible

To run Deploy Records select one or more records in the Data folder of a release package and select Deploy records in the XRM menu



After selecting the connection to import into click Next to initiate the import



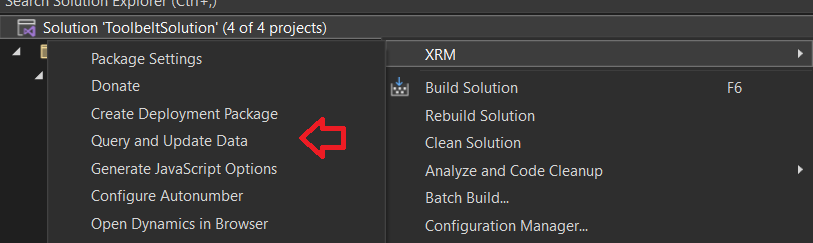
The system will display progress before a summary at completion

## Query & Update Data

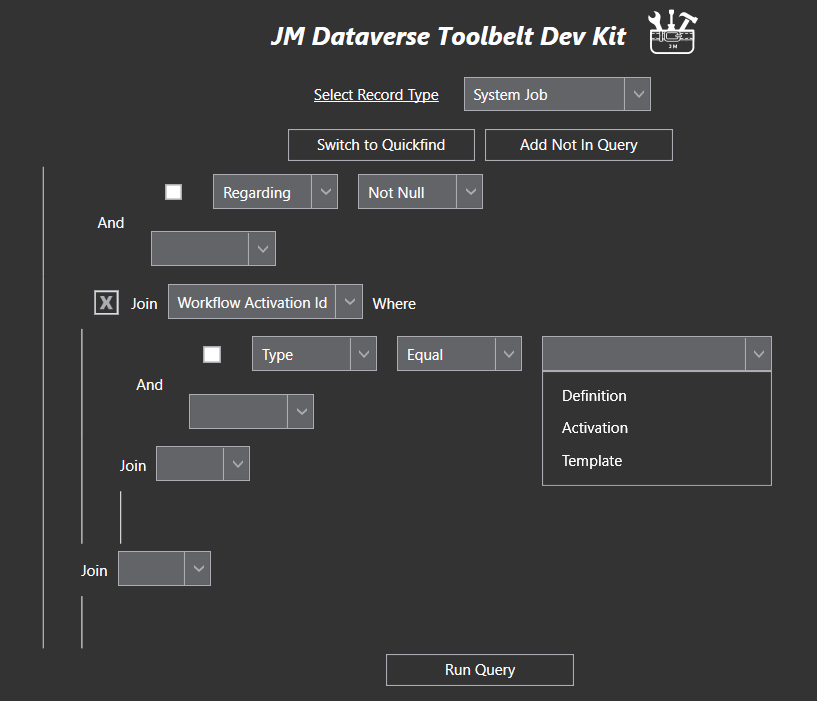
More commonly used in the toolbelt desktop app, this feature can be used to achieve any of the following

* Arbitrarily query data for any record type in the Dataverse instance
* Bulk update columns for the results of a query
* Bulk delete results of a query
* Bulk replace text in string fields for the results of a query
* Copy FetchXml for the query to the clipboard either raw or formatted for JavaScript
* Display total number of records for a query
* Export query results to Excel
* Open result in an editable form containing all columns in the table
* Open results in the Model Driven Web UI

To run the query dialog right click the solution and select Query and Update Data in the XRM menu

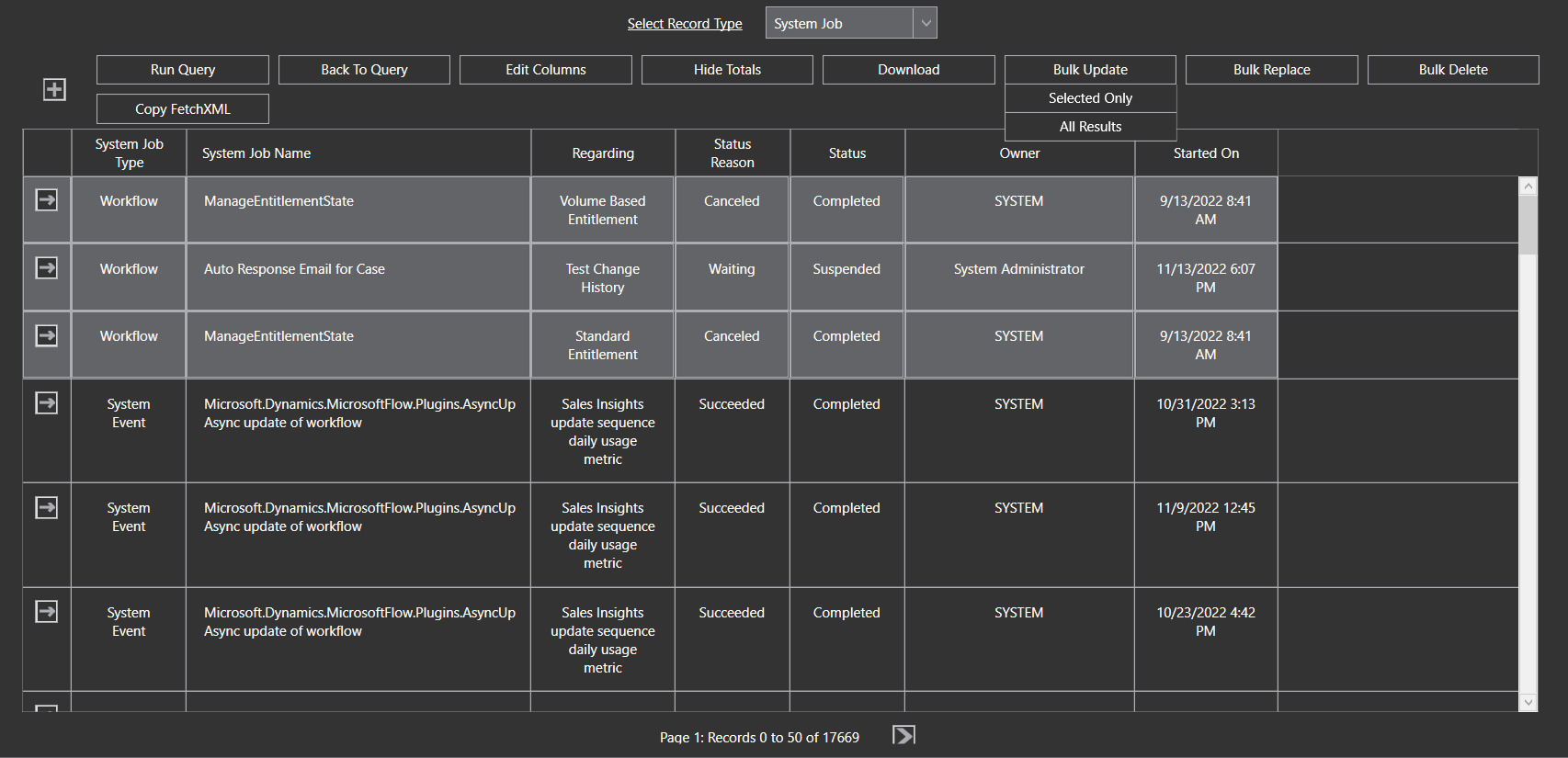


Once loaded select a record type and a query can then be created using a range of conditions, operators and joins



After the Query is run results are displayed in grid form with operations available to perform on the results

In this screenshot I have also clicked Display Totals and the total number of records is displaying in the age count at the bottom of the grid



## Solution Template

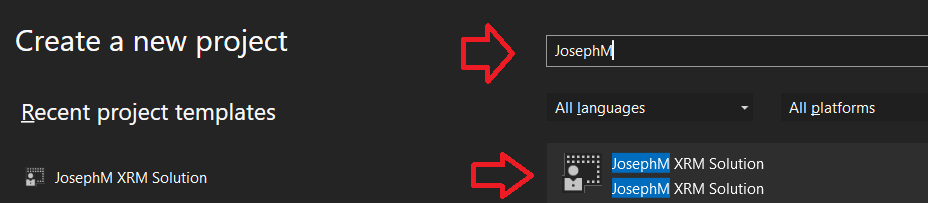
The Solution Template feature of the extension is a visual studio starter solution containing several projects for Dataverse development purposes.

New solutions created from the template contain the following projects all connected to an active Dataverse instance

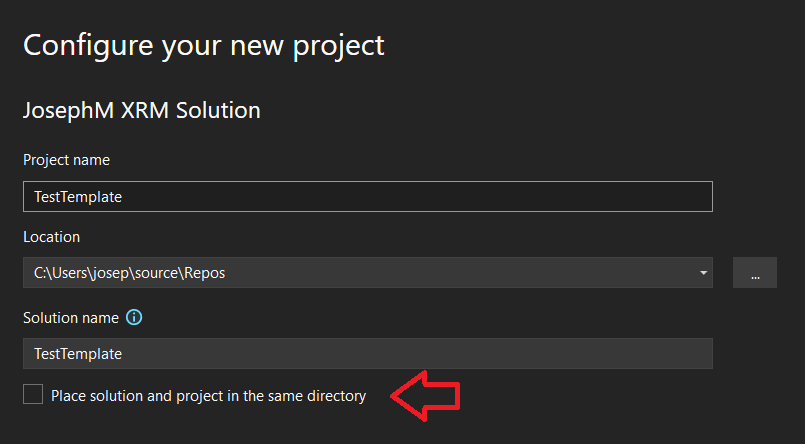
* console app
* plugin project -
* web resource project
* automated test project

As well as the plugin framework which abstracts out some of the low level plugin code, a range of class libraries and service classes simplify coding against the SDK to enable quicker outcomes and more maintainable code

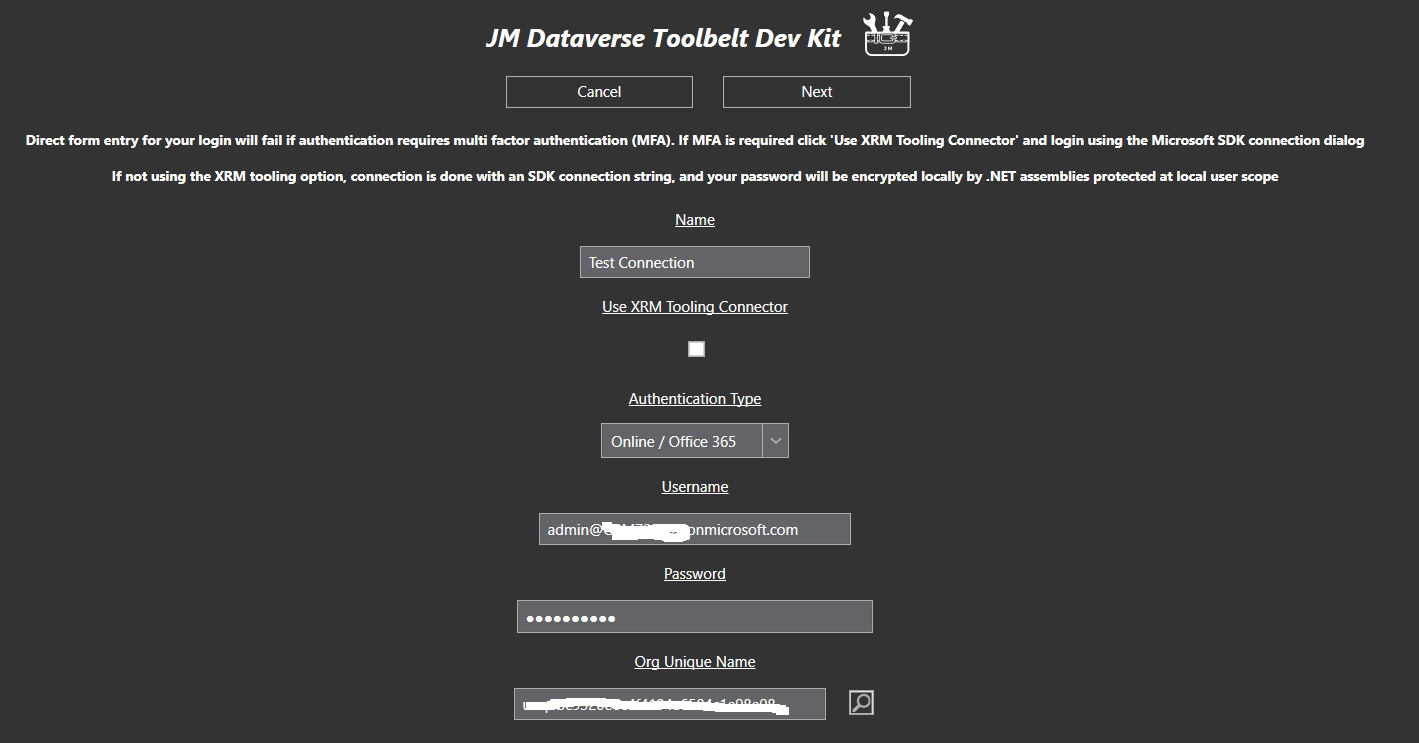
To create a solution from the template use the standard new project dialog in Visual Studio. Typing JosephM into the search box for the new project type will bring uo the solution template for selection



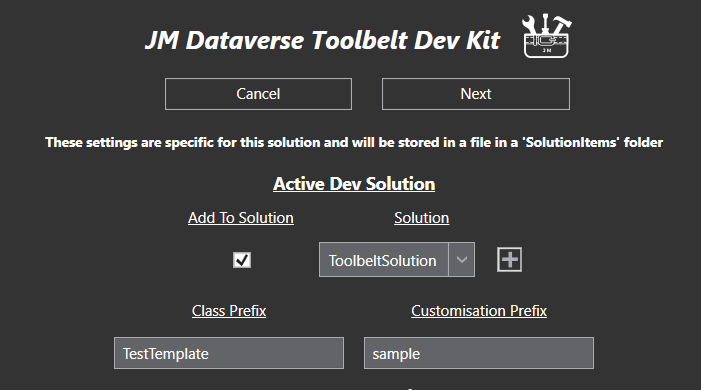
When entering the project name and location ensure “Please solution and project in the same directory” is unchecked. If it is checked the solution will not provision correctly



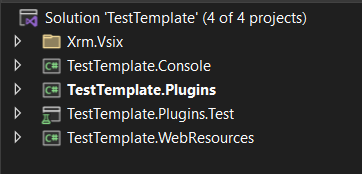
Provisioning of the project first requires the Dataverse instance connection to be entered



Then a form which includes details for Class Prefix and Customisation Prefix. These are used for initialising names of web resources and classes in the new projects



Click Next to provision the new solution. The screenshot below shows an example new solution created form the template

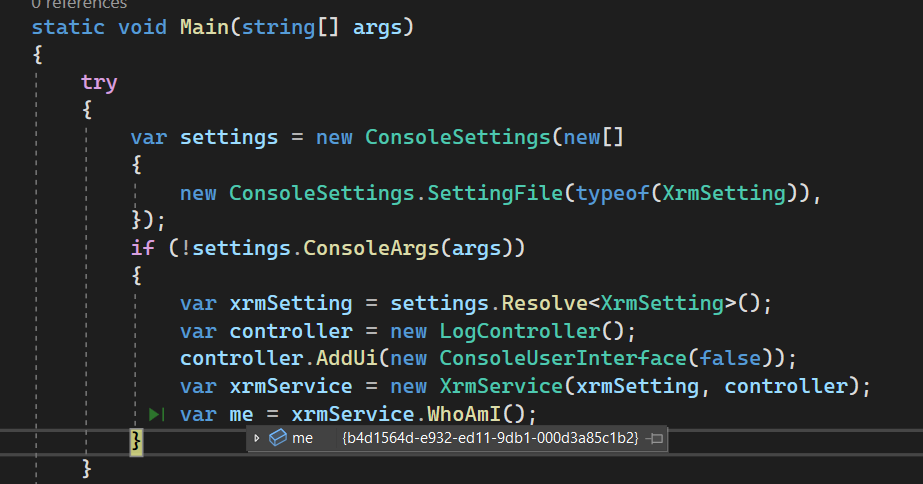


### Template Console Project

The console project in the template provides a console app with a preloaded connection to the Dataverse instance

As the project has a reference to the Plugin project for the template, it has access to a range of classes which simplify coding activities using the Dataverse SDK

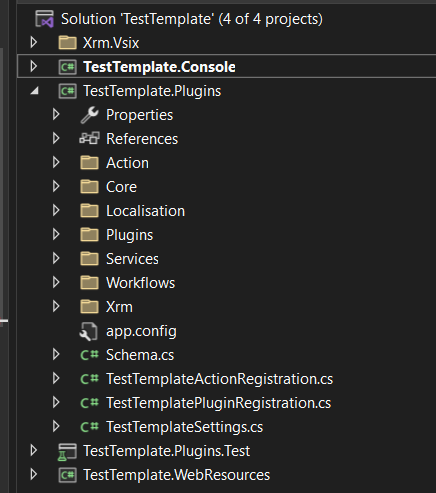
The screenshot below shows the initial state of the console app. As you can see the app is immediately connected to the Dataverse so provides a great starting point for a console app



### Plugin Project

The plugin project in the template provides a framework for custom plugins, workflow activities, and custom actions, as well as a range of classes which simplify coding activities using the Dataverse SDK.

The screenshot below shows the plugin project a created by the solution template



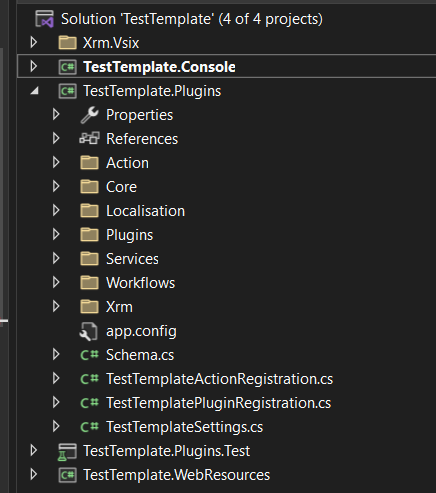
Classes and details of the plugin project will not be documented here and other options should be well thought through before committing to writing new custom plugin code

### Test Project

The test project in the template provides a framework for automated testing of behaviour in the Dataverse instance. The testing framework uses the active connection of the extension so is great for quickly writing new tests, and running regressions scripts, of plugins and other behaviours which have been customised into the platform

As the project has a reference to the Plugin project for the template, it has access to a range of classes which simplify coding activities using the Dataverse SDK

The screenshot below shows the test project a created by the solution template

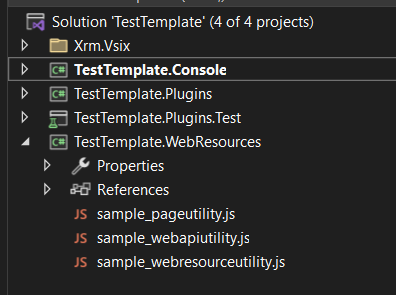


Classes and details of the test project will not be documented here and other options should be well thought through before committing to writing custom code for the platform

### Web Resources Project

The web resources project in the template provides a JavaScript utility classes for customising behaviour in the Dataverse forms

The screenshot below shows the web resource project a created by the solution template

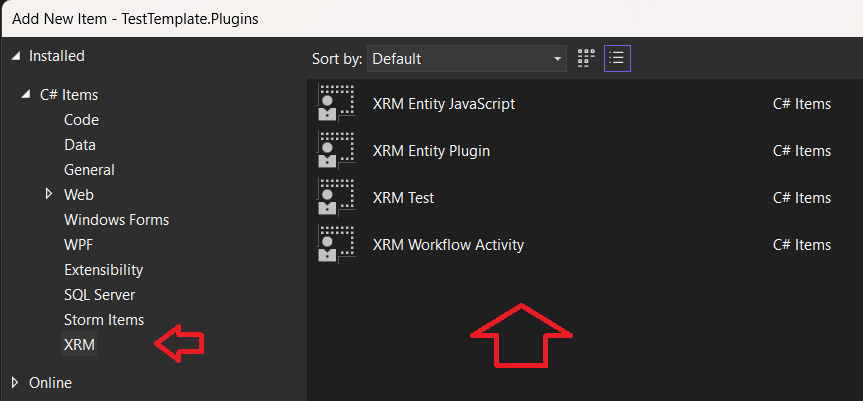


The JavaScript files will not be documented here and other options should be well thought through before committing to writing custom code for the platform

### Item Templates

The extension also contains several classes intended for use with frameworks in the solution template project

These are accessible in the XRM category of the new project item dialog



These items will not be documented here and other options should be well thought through before committing to writing custom code for the platform