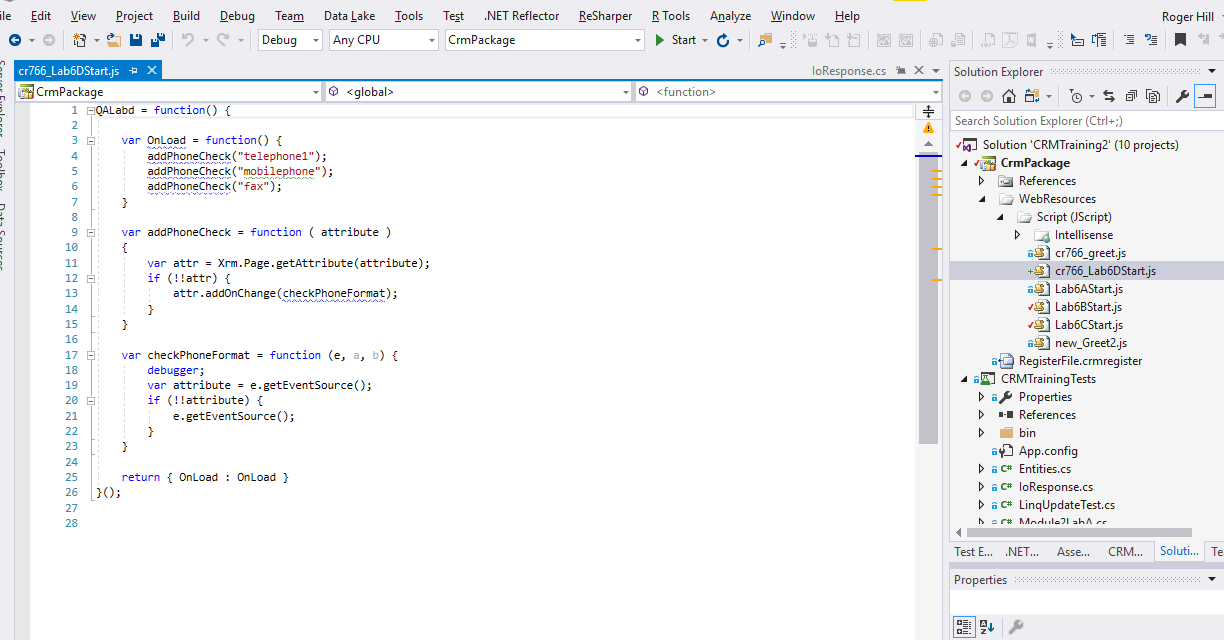
CRM Training Notes

# Solution

A Solution with all of my working to the LABS is available here: <https://github.com/rogerhillgsy/CRMTraining>

Pull this using GIT (Note this does not seem to work from within QA) and you should see something like this: -



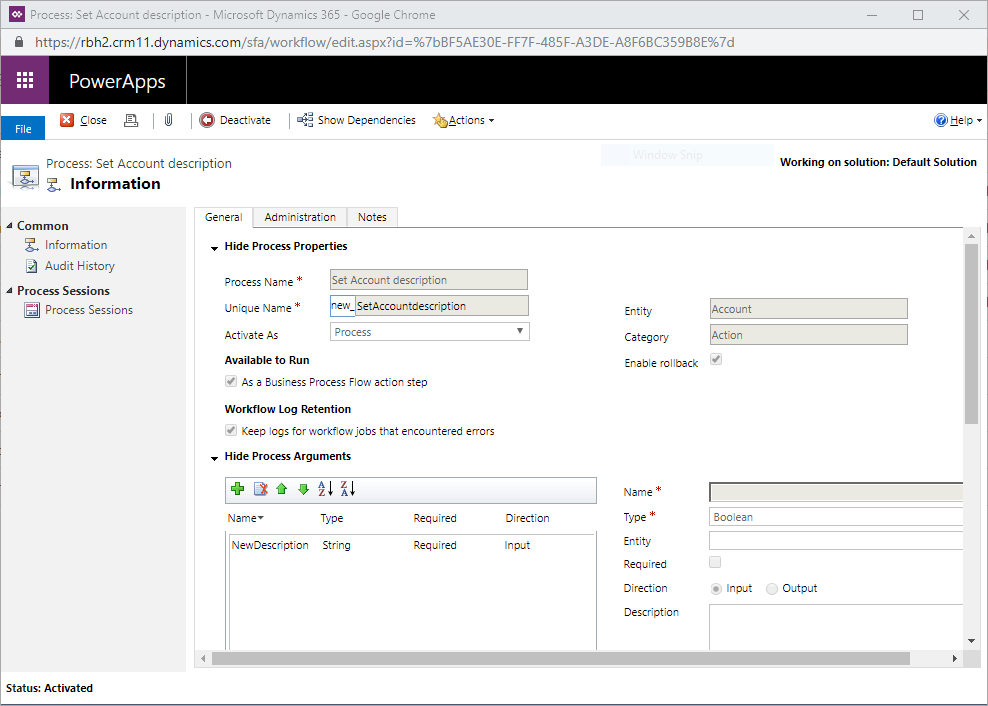
# Additional Labs

* Action and Business Process Flow
* Write Unit Test using FakeXrmEasy.
* Use CRM Developer Toolkit to add a Plugin template to your plugins project.
* Plugin to run on create and update of an entity, to write to trace the details of the PreImage, postImage and Target entity (where available).  
  This should include a list of attributes + values  
  Register against a contact and create and update a contact.  
  Observe which images are present in each case, and the attributes that appear in each.
* Microsoft Flow
  + Create a Flow to interact with CRM.

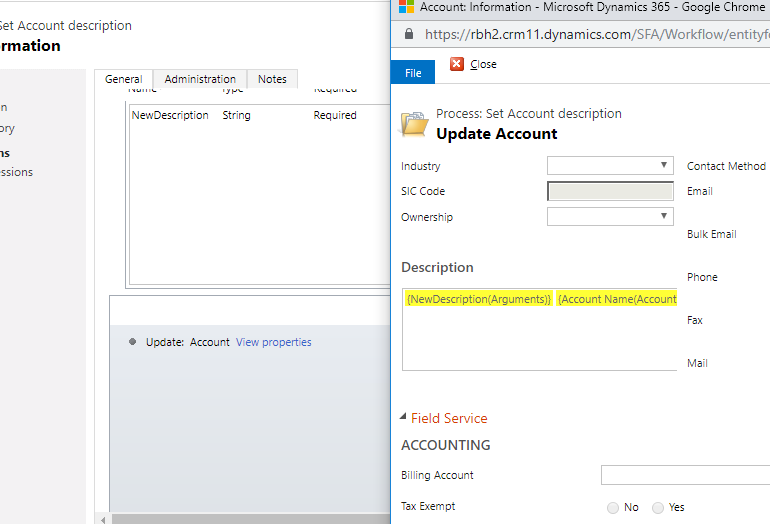
# Actions and Business Flow

Create an Action on an account to update the description field.

Give it one argument of “New Description

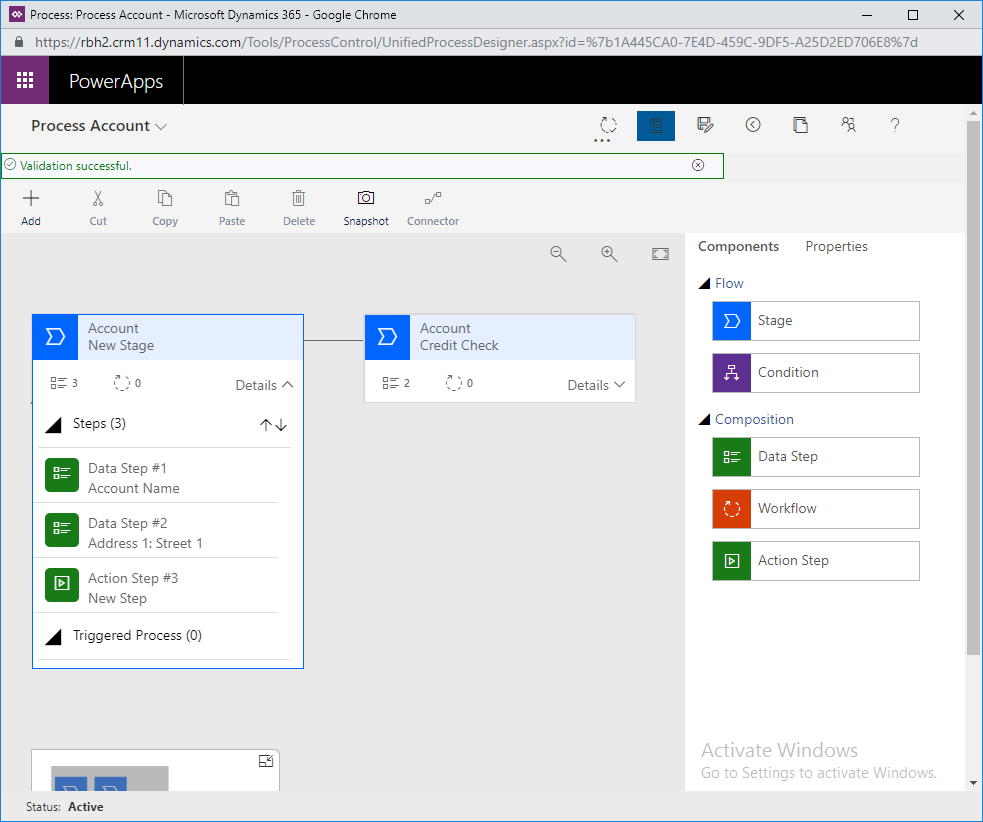


Add a step to update the account record with the new description parameter and the account name.



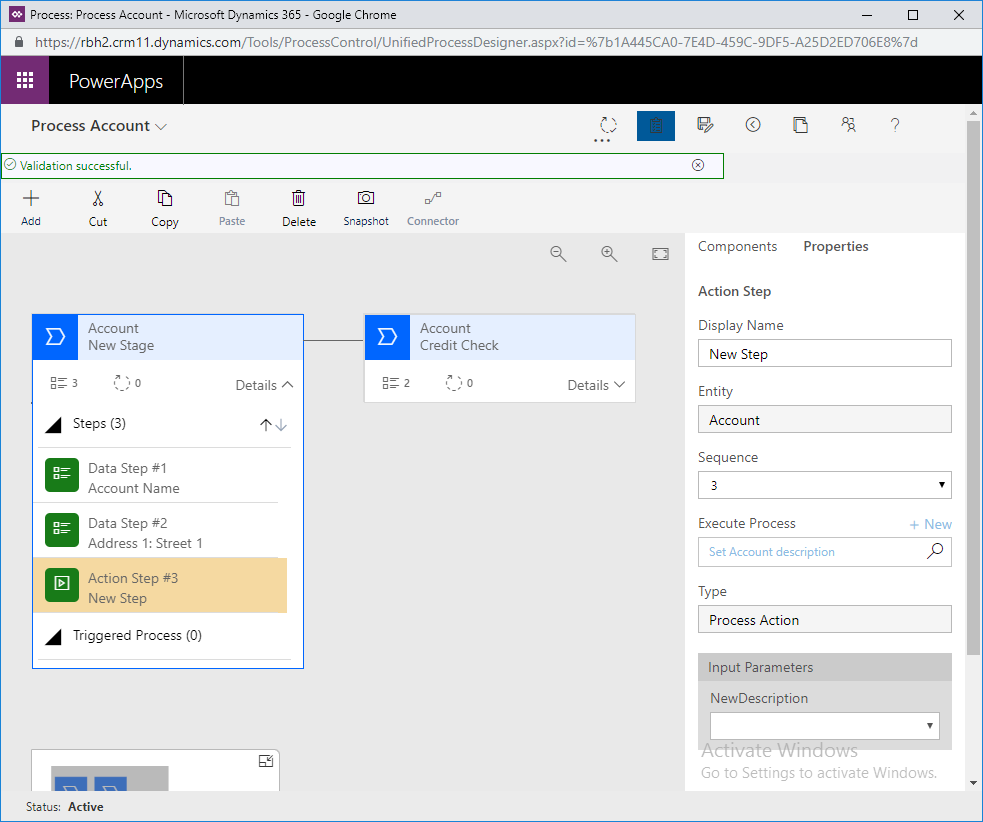
Create a new Business Process flow on the Account.

Create a first stage with a couple of fields.

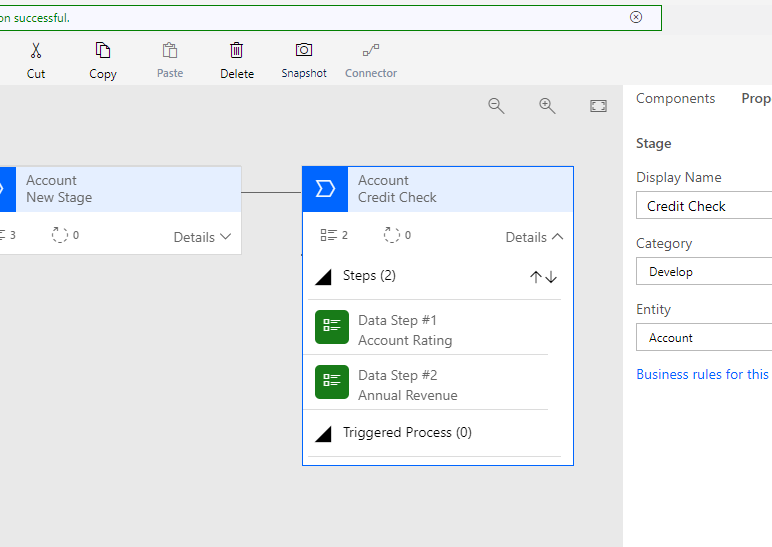


Then add an action step.

Set the action to run the action you have just created.



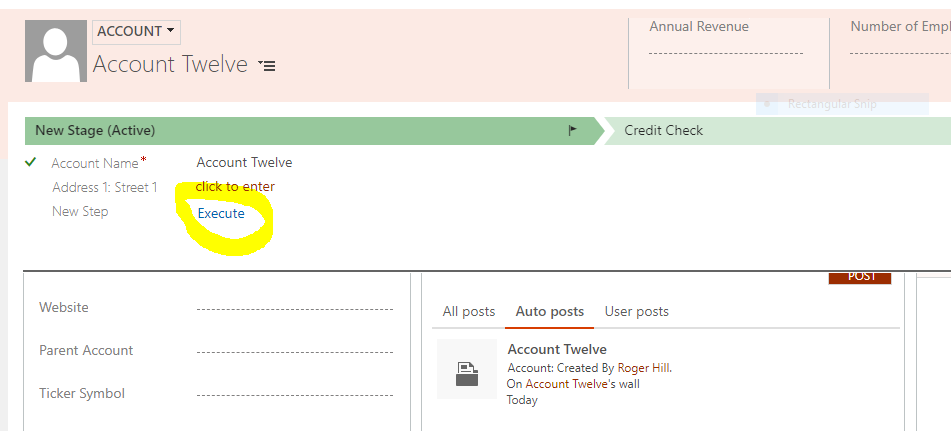
Add a second step with a couple more fields.

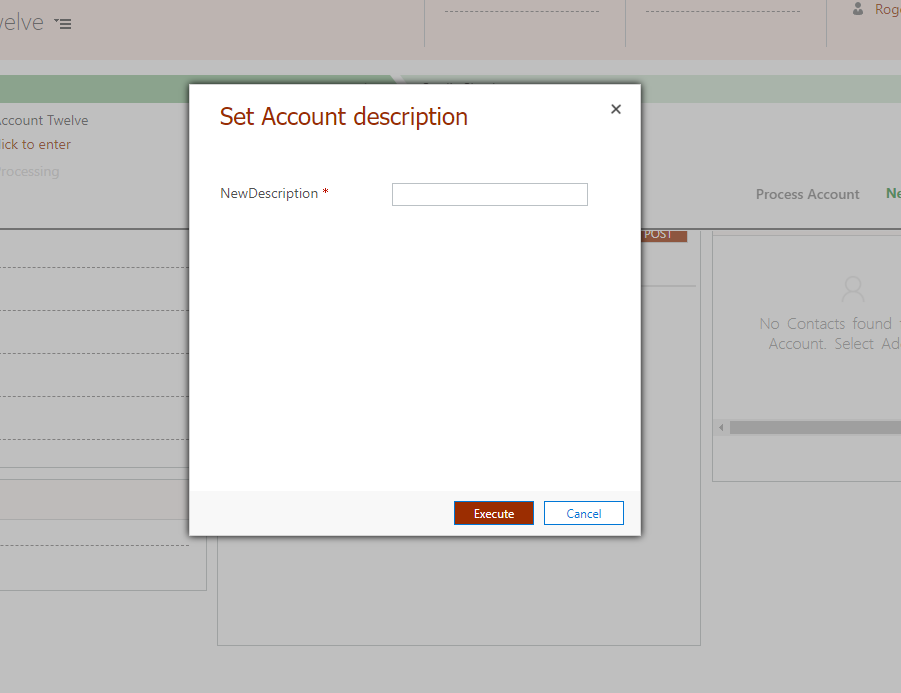


Validate and publish the BPF.

Create a new account records.

Test your new Action button in the BPF.





Enter values and observe the update to your Account records.

# Lab 7D – Validate Postcode Interactively

Add a postcode field to the Bank Account entity.

Create a change event on the Bank Account postcode field.

Use the attached IoResponse class to validate the postcode. (<https://postcodes.io/> - use the validation call)

# Write unit test using FakeXrmEasy

We will write a unit test to exercise our code activity from Module 4 Lab A.

Create a unit test, use Nuget to get FakeXrmEasy.9.

Set up a fake context containing three accounts.



Run the unit test and debug through your code activity.

# Use CRM Developer Toolkit to Add a Plugin Template

You should be able to add a plugin template from within Visual Studio…

# Javascript Intellisense

Look for MSXRMTOOLS.Xrm.Page.2016.js

Currently available here: <http://msxrmtools.com/Xrm.Page/MSXRMTOOLS.Xrm.Page.2016.js>

|  |
| --- |
| ///<reference path="Intellisense/MSXRMTOOLS.Xrm.Page.2016.js"/>  function Greet( e ) {  var firstName = Xrm.Page.getAttribute("firstname").getValue();  console.log(firstName + " display in the debug environmnet console");  alert("Welcome " + firstName );  } |

# Visual Studio Online Build tools.

For continuous integration and deployment. Look at:

**Dynamics 365 Build Tools**

<https://marketplace.visualstudio.com/items?itemName=WaelHamze.xrm-ci-framework-build-tasks>

# XrmToolbox

Very useful toolbox to manage many aspects of CRM – 100’s of plugins for any task you can think of.

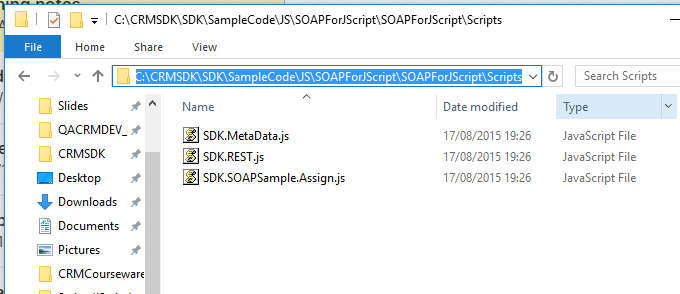
<https://www.xrmtoolbox.com/>

# CRM Rest Builder

Takes the old OData Query tool to a new level with support for WebApi.

<https://github.com/jlattimer/CRMRESTBuilder/releases>

# SDK JavaScript libraries



​

Use SDK.Rest to simplify