Training Slides Review for QACRMDEV

Author: Roger Hill

Course: QACRMDEV.

Start Date: 14/01/2019

Location: QA Offices, International House, St Katherines Dock.

# General points

## Classroom setup

The PCs had clearly not been set up with a specific image for this course. There were packages for other courses (for example Power BI course) but none for QACRMDEV.

So no PowerPoints, Labs or CRM SDK on trainees’ machines.

I was able to resolve this by sharing the required resources from the Instructor machine.

## Trainees

Two of the trainees did not meet the prerequisites for the course (.Net/C# programming experience and Dynamics CRM customization/management experience).

This was primarily due to confusion within their organization on a suitable course, but I feel that the course was of limited value to Keith and Dan from the NFU.

## Labs out of date

It was noted that a number of Labs were out of date.

This is not necessarily an issue with the Labs as originally written, but the environment in which the labs are run is quite dynamics and tends to change and become out of date quite quickly. This is becoming increasingly obvious (even between the last time I ran the course in November ’18 and last week)

For example –

* Web services referred to in some of the labs no longer existed.
* The way in which the Labs instructed trainees to operate CRM was not always up to date and could cause confusion.
* The Labs should be recommending trainees to use nuget to resolve dependencies on SDK library files. The Labs should never be telling trainees to hardcode references to SDK library files on their local machine.

## Topics Covered

Coverage of BPFs needs to be in module 4. Integration with Actions and workflows. Develop a specific lab to cover this.

Put coverage of Azure topics into standalone module

* Azure Service Bus
* Azure SQL DB replication
* Virtual Entities. (OK not strictly an Azure topic, but would fit better).

# Module 1 – Extensible framework

No issues with the slides.

However, the LABS were incorrect. The SDK bin and plugin profiler should not be updated with nuget to v9. When this is done the two directories do not work with the Microsoft Dynamics 365 Developer toolkit for Visual Studio.

A separate v9 copy of the plugin registration tool is required.

The LABS should say: -

* Unpack the SDK. (v8)
* Set tool paths in Visual studio to this v8 version of the SDK.
* Use nuget to obtain a separate V9 instance of the Plugin registration tool.

# Module 2 – Data Access

References to the Microsoft dll’s should be added using nuget. Hard coding paths to the .dll files is poor practice, and will cause configuration/consistency issues.

# Module 3 – Data Access

No issues with this module.

# Module 4 – Code Activity

This module should stress that Code Activities and Plugins should normally be deployed from the Visual Studio framework project. This ensures consistency and good configuration management.

The Plugin registration tool should only be used to deploy Code Activities and Plugins for test purposes. The normal method should be to deploy from within Visual Studio. This is the only way to guarantee consistency in the build and deployment process.

There are errors in the Code Activity implementation– in particular it is querying CRM for attribute values where it should be obtaining them from the Execution Context.

I would suggest to add a Lab to demonstrate the creation of a BPF and the way in which action buttons can be added. Also the way that pre and post stage workflows can be added.

Lab A – A few issues: -

* A code template for the workflow activity should be generated using the Dynamics 365 toolkit.
* The workflow activity can get the account name from the execution context. A CRM query is not required.
* Normally code activities should be deployed from visual studio using the Dynamics 365 toolkit.
* Code activities should be debugged using the profiler by saving to an entity.
* Mention other approaches to testing and unit testing (see Extra Module below)

# Module 5 – Plugins

There needs to be emphasis that best practice is to deploy plugins from the Microsoft D365 template project. Deploying production code using the Plugin Registration tool is poor practise and will cause numerous consistency and reliability problems in the long term. See below re testing of plugins and use of Fakes.

Not sure that the detailed description of the Azure pipeline is relevant here – it feels a bit out of place on this course. A more limited description might be preferable.

The plugin labs could be improved: -

* Add an initial lab to create a plugin without external dependencies. There was too much focus in the existing lab in resolving issues with the external web service used in the lab.
* The LAB B refers to a web service that no longer exists. I would suggest rewriting to use <https://postcodes.io> web service.
* Add a lab for setting up Azure service bus. Make it an optional lab for trainees to complete in their own time.

On the subject of Azure, it is also worth mentioning the possibility of replicating CRM entities into an Azure SQL database.

# Extra Module – post 5- testing techniques

I think there should be a module on approaches to testing CRM code (This is a developer course after all).

There are a number of approaches that can be applied to testing .Net code in CRM: -

* Structuring plugins/code activities to facilitate unit testing.
* Use of CRM specific faking framework (FakeXrmEasy).
* Using faking framework to write tests for CodeActivities and Plugins.
* Use of Profiler is already covered.

The course should include a description of these and relevant Labs.

# Module 6 – Client side scripting.

Needs to be clearer and more coherent on the subject of Xrm.Page (which is deprecated) vs passing the execution context as a function parameter.

Need to have emphasis on the use of the Microsoft D365 solution template to manage and deploy web resources.

The module should be talking more about best working practises and debugging using the Chrome Developer Tools.

Mention tools such as Level-up for working with CRM in chrome.

With Labs, I would suggest a Lab that focuses initially on using Xrm.Page, then a second lab that focuses on how this would be converted to the new v9 world of passing execution context as a parameter. (This is likely to reflect trainees’ real world experience in any event)

# Module 7 – Web resources and accessing Odata services.

The module should be guiding developers to create web resources through the Visual Studio template solution. Creating and maintaining resources through the UI is poor practice, and leads to configuration issues, errors and poor reliability.

Lab 7A should point the class towards the Microsoft D365 template solution rather than an ASP.Net Web application. The Web Application adds only very limited value in this lab.

Point developers towards using the CRMRestBuilder and Xrm.WebApi interface.

I would reintroduce the autocomplete/postcode lab that was in the November course. I felt this was a useful exercise. Use postcodes.io for UK postcode checker.

Discussion of SOAP vs ODATA is not relevant to the current situation where WebApi is the recommended approach. Module needs rewriting to reflect this.

Lab 7B – information on Chrome Postman is out of date. Chrome app is deprecated. Should use Windows app. Example is hard to follow.

The end of this module is rambling and incoherent – do we really need to discuss JQuery here? SDK Rest API superseded by Xrm.WebAPI

Consider examples where Xrm.Page is converted to a parameter based execution context. (See Module 6 above)

# Module 8 – Navigation.

This needs to be quite app centric – this is the way that CRM navigation is headed.

There is a specific technique for editing the default sitemap – this needs to be discussed. (i.e Add sitemap to a solution, which then makes it available for editing with the App designer)

Discuss why we would create new Apps and approaches to be used for this.

The class all said that the Lab A did not work – in particular it is no longer possible to create a web interface type App.

Use of Ribbon Editor still quite relevant, but need to check to make sure that labs are up to date with the tool. Also need to note that Enable Rules are no longer needed/relevant.

# Module 9 – Virtual Entities

I have attempted to follow the guidelines in the PowerPoint, but feel that we need to check this to make sure it still works.

There does not appear to be any Lab 9 in the material provided.

I would reduce the amount of time devoted to Virtual Entitles and spend more time on client side issues (BPF Example with Action buttons and triggering of actions via pre/post stages.).

I would combine the module on Virtual entities with the discussion of Azure Service Bus, and Azure DB replication. I think this would be a better fit.

Labs could be improved – check to make sure that they exist, still work, and can be followed by trainees.

Note that some of the labs cannot be completed with a free trial account. (Interaction with Azure SQLDB seems to lead to requiring credit card details at some point…)

# Module 10 – Summary of Course and V9 changes.

Module 10 summary (was Module 9)

Summary of V9 changes – not sure how much this is needed now – but…

Still useful to have a summary of what has been covered on the course.  
Ideally ensure that improvements have been covered on the course as well as dependencies.

V9 changes to cover -

* Make coverage of API more coherent
* JavaScript web resource dependencies.
* NOT IN query in advanced find
* Wrong Link fix
* Multi select option sets.
* Virtual Entities.