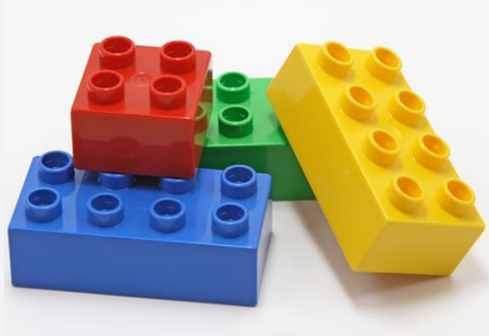




CI/CD for Dynamics/PowerApps

The Building Blocks



About myself

Name: Alex Shlega

Title: Dynamics 365 Consultant/Developer/Solution Architect

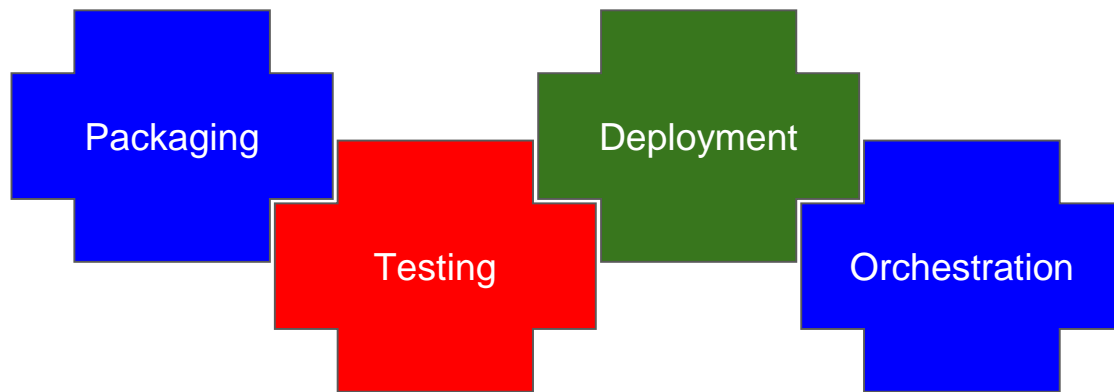
Blog: <https://www.itaintboring.com>

Linkedin: <https://www.linkedin.com/in/alexandershlega/>

Twitter: <https://twitter.com/ashlega>



CI/CD Puzzle



- **Solutions**
- **Solution Packager**

- **Easy Repro**
- **Fake XRM**

- **Build Tools**
- **Configuration Data**

- **DevOps**

Packaging: Managed vs Unmanaged

- Microsoft recommends managed solutions for any environment other than dev
- At least two reasons for that: ability to prevent modifications of certain solution components, and, also, ability to delete components by installing a newer version of the managed solution
- Does this settle the old argument? Not necessarily, but, since managed solutions are recommended, it's a good enough reason to at least try using them

← Export this solution ×

Version number* ⓘ

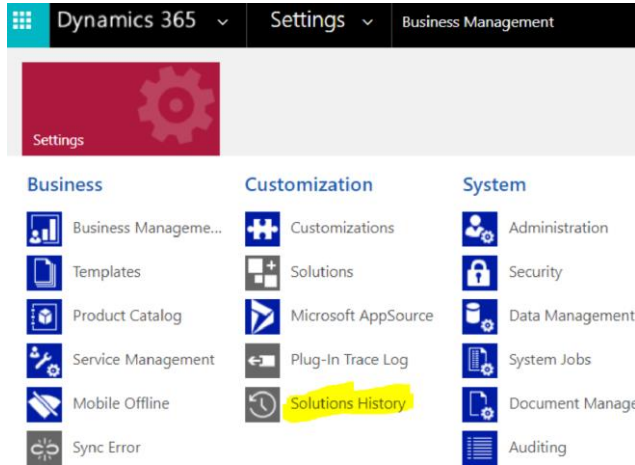
Current version 1.0.0

Export as

Managed (recommended) ⓘ
The solution is moving to a test or production environment. [Learn more](#)

Unmanaged
The solution is moving to another development environment or source control. [Learn more](#)

Packaging: Solution History



The screenshot shows the Dynamics 365 Settings navigation pane. The top bar includes 'Dynamics 365', 'Settings', and 'Business Management'. Below this, there are three main categories: Business, Customization, and System. The 'Solutions History' option under the Customization category is highlighted in yellow.

- Business**
 - Business Managem...
 - Templates
 - Product Catalog
 - Service Management
 - Mobile Offline
 - Sync Error
- Customization**
 - Customizations
 - Solutions
 - Microsoft AppSource
 - Plug-In Trace Log
 - Solutions History
- System**
 - Administration
 - Security
 - Data Management
 - System Jobs
 - Document Manage
 - Auditing

- Which solutions were involved
- What happened to them (export, import, etc)
- When did it happen

Custom Solutions History

<input type="checkbox"/>	Solution Name	Start Time	End Time	Solution Version	Publisher Name	Operation
	TestProfileMigration	2019-09-02 7:54 PM	2019-09-02 7:54 PM	1.0.0.0	itaintboring	Export
	ItAintBoringPCFControls	2019-08-21 8:33 PM	2019-08-21 8:33 PM	1.0	ItAintBoring	Import
	ItAintBoringPCFControls	2019-08-21 8:30 PM	2019-08-21 8:30 PM	1.0	ItAintBoring	Import
	ItAintBoringPCFControls	2019-08-21 9:12 AM	2019-08-21 9:12 AM	1.0	ItAintBoring	Import
	ItAintBoringPCFControls	2019-08-20 9:21 AM	2019-08-20 9:21 AM	1.0	ItAintBoring	Import

Packaging: Solution Layers

The screenshot shows the Microsoft Dynamics 365 Solution Explorer interface. The 'Account' entity is selected in the left-hand tree view. The right-hand pane displays the 'Entity Definition' for 'Account' with the following details:

- Display Name: Account
- Plural Name: Accounts
- Name: account
- Primary Image: Default Image
- Color: #794300
- Description: Business that represents a customer or potential customer. The

For the individual solution components (entities, forms, fields, etc), we can now see the layers of solutions which included changes to that component

The 'Solution Layers' pane for the 'Account' entity shows the following information:

- Name: Account
- Type: SystemForm

Below this, a table lists the solutions that include changes to this component:

Solution	Publisher	Order
Active	Default Publisher for treecatsoftware	12
LinkedInSalesNavigatorControlsForUnifiedClie	Microsoft Dynamics 365	11
FieldService	Microsoft Dynamics 365	10

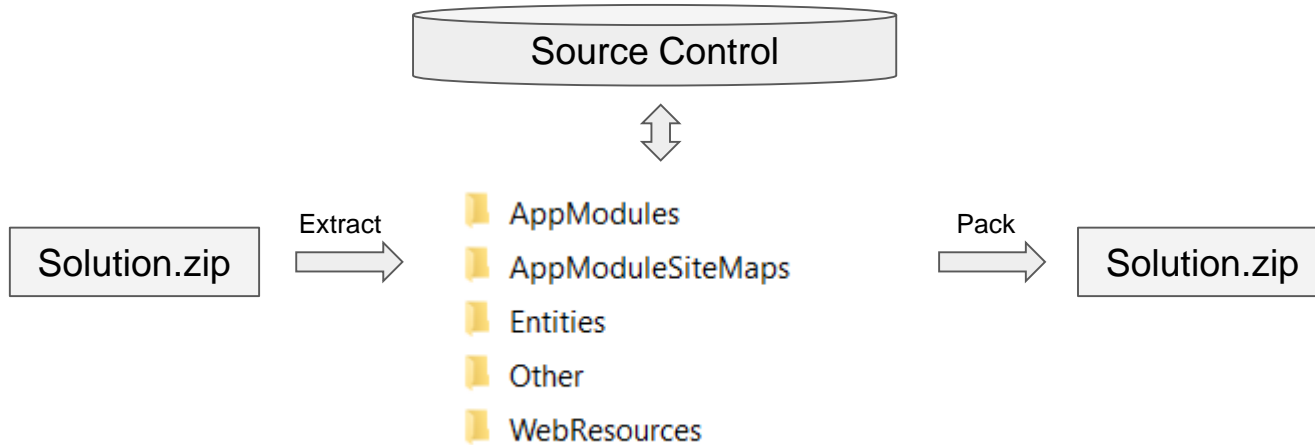
The 'Solution Layers' pane for the 'Account' entity shows the following information:

- Name: Account
- Type: Entity

Below this, a table lists the solutions that include changes to this component:

Solution	Publisher	Order
AppforOutlookPatch	Microsoft Dynamics 365	13
Active	Default Publisher for treecatsoftware	12
MicrosoftCrmPortalDependencies	Microsoft	11

Packaging: SolutionPackager



- Recommended by Microsoft for source control “integration”
- Makes XML merge somewhat easier
- While packing, provides self-validation
- Deployment to the target environment ensures ultimate solution validation
- Download script: <https://docs.microsoft.com/en-us/dynamics365/customer-engagement/developer/download-tools-nuget>

Configuration Migration Tool

- Works for configuration data migration
- Earlier, it would have to be started manually. As an alternative, package deployer can utilize configuration migration tool files
- There is a powershell module now:

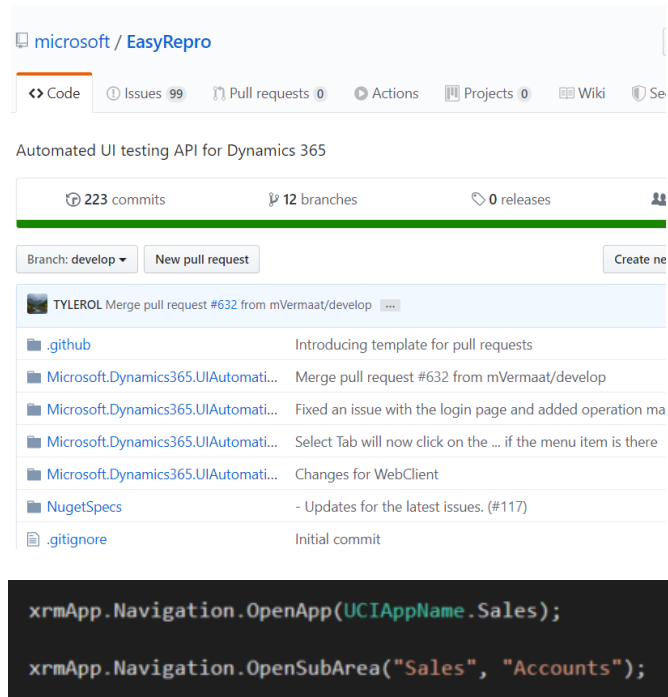
<https://www.powershellgallery.com/packages/Microsoft.Xrm.Tooling.ConfigurationMigration>

- Advanced filtering and/or data transformation is still not supported



Testing: EasyRepro

- The purpose of this library is to provide Dynamics customers the ability to facilitate automated UI testing for their **model-driven** app projects
- It is built on top of Selenium
- It knows how to work with PowerApps/Dynamics - usually, there is no need to work with HTML directly
- Can be easily added to the DevOps pipelines
- <https://github.com/microsoft/EasyRepro>



microsoft / EasyRepro

Code Issues 99 Pull requests 0 Actions Projects 0 Wiki Security

Automated UI testing API for Dynamics 365

223 commits 12 branches 0 releases

Branch: develop New pull request Create new pull request

TYLEROL Merge pull request #632 from mVermaat/develop


- .github Introducing template for pull requests
- Microsoft.Dynamics365.UIAutomati... Merge pull request #632 from mVermaat/develop
- Microsoft.Dynamics365.UIAutomati... Fixed an issue with the login page and added operation ma
- Microsoft.Dynamics365.UIAutomati... Select Tab will now click on the ... if the menu item is there
- Microsoft.Dynamics365.UIAutomati... Changes for WebClient
- NugetSpecs - Updates for the latest issues. (#117)
- .gitignore Initial commit


```
xrmApp.Navigation.OpenApp(UCIAppName.Sales);  
xrmApp.Navigation.OpenSubArea("Sales", "Accounts");
```


Orchestration: DevOps

- Build and release pipelines
- Source code repository (git)
- Ability to run PowerShell in the pipelines
- Ability to trigger the pipelines automatically
- Hosted agents
- Public and private projects





 **CDSDemo** +


 Overview


 Boards


 Repos

 Pipelines

 Builds


 Releases

 Library

 Task groups

Orchestration: PowerApps Build Tools

Automate your application lifecycle management (ALM) with PowerApps Build Tools (Preview)

 Per Mikkelsen, Principal Program Manager, Wednesday, July 10, 2019



- Public Preview since June 20, 2019
- <https://powerapps.microsoft.com/en-us/blog/automate-your-application-lifecycle-management-alm-with-powerapps-build-tools-preview/>

Export Solution

Pack Solution

Unpack Solution

Import Solution

Demo pipelines

Export and Unpack

- Will export solution from the branch dev environment
- Will unpack it
- Will check it into the repo

Build and Test

(triggers automatically on the master branch)

- Will build managed solution from the source control
- Will import it into the branch QA environment
- Will run the test
- Will create build artefacts

Prepare Dev

- Will build unmanaged solution from the source control
- Will import it into the branch Dev environment

Assumptions for the demo

Everything is already in the source control, always deploying complete solution, no configuration data

Building a pipeline

Get the tools: <https://marketplace.visualstudio.com/items?itemName=microsoft-lsvExpTools.PowerApps-BuildTools>

The screenshot displays the Azure DevOps pipeline editor interface. At the top, a navigation bar includes tabs for 'Tasks', 'Variables', 'Triggers', 'Options', 'Retention', and 'History'. To the right of these tabs are action buttons: 'Save & queue', 'Discard', 'Summary', 'Queue', and a menu icon. Below the navigation bar, the left sidebar shows the pipeline configuration: 'Pipeline' (Build pipeline), 'Get sources' (CDSDemo, master), and 'Agent job 1' (Run on agent). The main workspace is titled 'Add tasks' and features a 'Refresh' button and a search bar containing the text 'powerapps'. Below the search bar, a list of tasks is displayed, each with a PowerApps icon and a title: 'PowerApps Set Solution Version', 'PowerApps Create Environment', 'PowerApps Tool Installer', and 'PowerApps Publish Customizations'.

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue ...

Pipeline
Build pipeline

Get sources
CDSDemo master

Agent job 1
Run on agent

Add tasks | Refresh

powerapps

- PowerApps Set Solution Version
PowerApps Set Solution Version
- PowerApps Create Environment
PowerApps Create Environment
- PowerApps Tool Installer
PowerApps Tool Installer
- PowerApps Publish Customizations

Demo scenario

A new developer who just joined the team is tasked with adding a new feature. There are a few steps involved:

- Setting up a dev environment
- Setting up a test environment
- Getting unmanaged solution from the source control into the dev environment
- Implementing new feature in the dev environment
- Running regression tests
- Pushing unpacked solution to the source control (on the branch)
- Re-testing managed solution in the test instance
- Merging into master branch
- Re-testing in the master test instance and creating a build artefact

LET'S REVIEW THE PIPELINES,
PREPARE DEV ENVIRONMENT FOR THE BRANCH,
& MAKE A CHANGE IN THE SOLUTION

Demo Steps

- Create a branch
- Set up branch connection
- Deploy solution from the source control to the new dev environment using a pipeline

```
ash1e@DESKTOP-MV5HBTU MINGW64 /c/Work/Projects/CDSDemo (master)
$ git checkout -b Feature1
Switched to a new branch 'Feature1'

ash1e@DESKTOP-MV5HBTU MINGW64 /c/Work/Projects/CDSDemo (Feature1)
$ git push origin Feature1
Total 0 (delta 0), reused 0 (delta 0)
To https://dev.azure.com/itaintboring/_git/CDSDemo
* [new branch]      Feature1 -> Feature1

ash1e@DESKTOP-MV5HBTU MINGW64 /c/Work/Projects/CDSDemo (Feature1)
$ |
```

git branch -D Feature1

git push origin --delete
Feature1

Job

Started: 9/23/2019, 9:21:31 PM

Pool: Hosted VS2017 · Agent: Hosted Agent

... 2m 51s

✓	Prepare job · succeeded	<1s
✓	Initialize job · succeeded	1s
✓	Checkout · succeeded	12s
✓	Replace solution code files · succeeded	1s
✓	PowerApps Tool Installer · succeeded	26s
✓	PowerApps Pack Solution (Unmanaged) · succeeded	3s
✗	PowerApps Import Solution · 1 error	2m 4s
✗ The request channel timed out while waiting for a reply after 00:02:00. Increase the timeout value passed to the call to Request or increase the SendTimeout value on the Binding. The time allotted to this operation may have been a portion of a longer timeout.		
✓	Post-job: Checkout · succeeded	<1s
✓	Finalize Job · succeeded	<1s
✓	Report build status · succeeded	<1s

Make a change (Add a new field, put it on the form)



NEW ENTITY

New New Entity

General

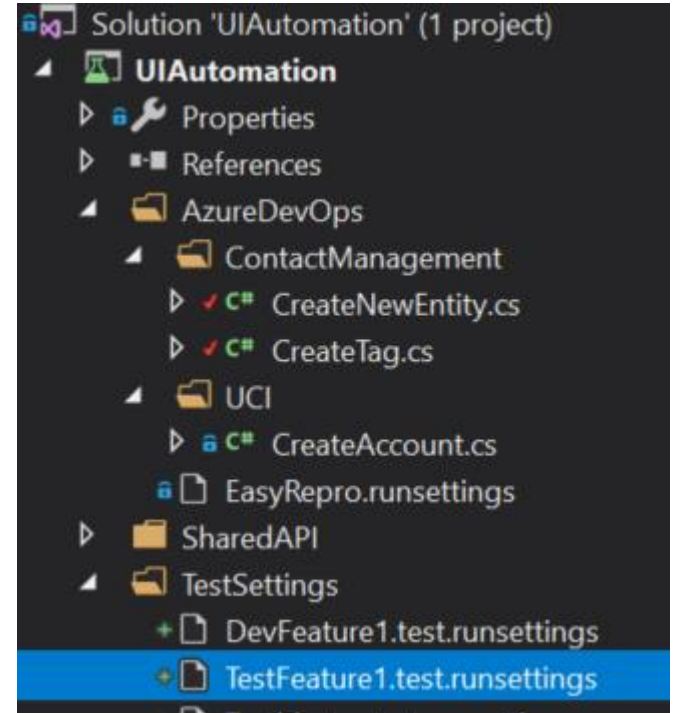
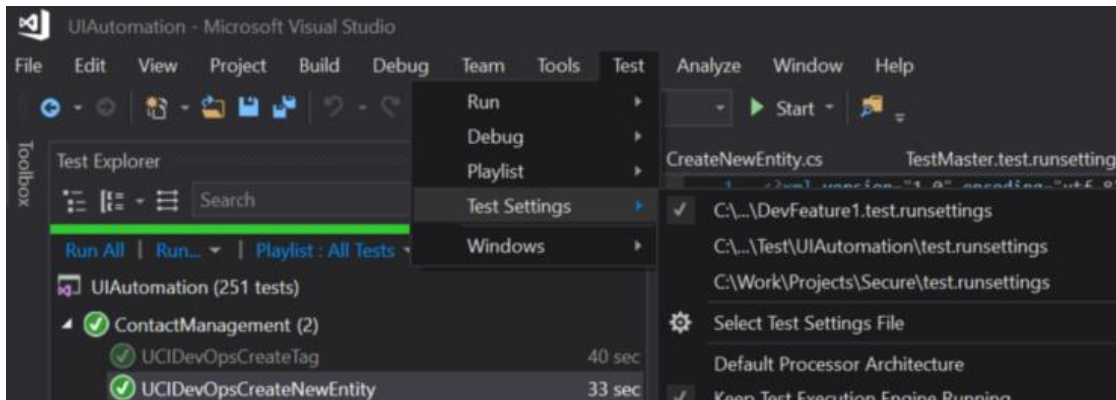
Name

*

Demo Sep 23

Add test.runsettings files to the UIAutomation

- For Dev environment
- For Test environment



Update the tests and run them manually in the feature Dev environment

The image shows a screenshot of the Visual Studio interface. On the left, the Test Explorer window is open, displaying a tree view of test categories. The 'ContactManagement' category is expanded, showing two tests: 'UCIDevOpsCreateTag' and 'UCIDevOpsCreateNewEntity', both of which have passed (indicated by green checkmarks) and took 32 seconds to run. Below the tree view, a 'Summary' section indicates that the 'Last Test Run Passed' with a total run time of 0:01:05.175343 and that 2 tests passed.

On the right, the Code Editor window shows the source code for 'CreateNewEntity.cs'. The code is written in C# and includes a class definition and a test method. The line numbers 23 through 41 are visible on the left side of the code editor.

```
23
24 [Class]
25 public
26 {
27     _te
28
29     _us
30     _pa
31     _xr
32     _br
33 }
34
35 [TestCa
36 [TestMe
37 public
38 {
39     var
40     usj
41 {
```

Commit and push updated tests to the remote

- `git commit -am "test settings"`
- `git push origin Feature1`

Demo Steps

- Export solution and push it into the branch using a pipeline
- Build and test Feature 1 branch
- Merge into Master
- A test on the master branch will start automatically

Export and unpack solution on Feature1 branch

Set up to be started manually - there is no trigger for this one since solution updates are happening in the CDS/Dynamics environment

itaintboring / CDSDemo / Pipelines / Builds / **Export and Unpack** / #20190925.1

✓ #20190925.1: Updated test

Manually run today at 9:30 pm by Alex Shlega ↗ CDSDemo ↗ **Feature1** ↗ cd866fb

[Logs](#) [Summary](#) [Tests](#)

Job

Pool: [Azure Pipelines](#) · Agent: Hosted Agent

- ✓ Prepare job · succeeded
- ✓ Initialize job · succeeded
- ✓ Checkout · succeeded
- ✓ PowerApps Tool Installer · succeeded
- ✓ PowerApps Export Solution · succeeded
- ✓ PowerApps Export Managed Solution · succeeded
- ✓ PowerApps Unpack Solution · succeeded
- ✓ Git push to origin · succeeded
- ✓ Post-job: Checkout · succeeded
- ✓ Finalize Job · succeeded
- ✓ Report build status · succeeded

Run Build and Test pipeline

Set up to trigger automatically on the master branch only. Build artefacts will be created on the master branch only as well

The screenshot shows a GitHub Actions workflow run for the 'Build and Test' pipeline. The run was manually triggered by Alex Shlega at 10:18 pm on 9/24/2019. The job, named 'Build and Test', is running on a 'Hosted VS2017' agent. The pipeline consists of 15 steps, all of which have succeeded. The steps include preparing the job, initializing, checking out, replacing solution code files, installing PowerApps tools, packing the solution, importing the solution, building the test solution (with one warning), installing the Visual Studio Test Platform, running the tests, and publishing pipeline artifacts. The total duration of the job is 4m 25s.

itaintborring / CSDSDemo / Pipelines / Builds / Build and Test / #20190925.4

Manually run today at 10:18 pm by Alex Shlega @ CSDSDemo @ Feature1 @ 481ecd5

Logs Summary Tests

Job Started: 9/24/2019, 10:18:20 PM
Pool: Hosted VS2017 · Agent: Hosted Agent ... 4m 25s

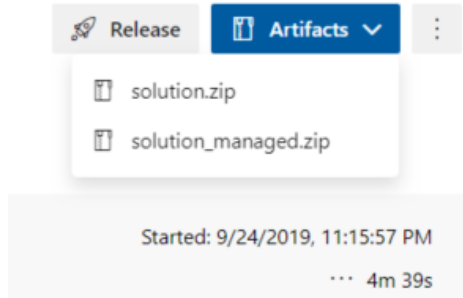
✓	Prepare job · succeeded	<1s
✓	Initialize job · succeeded	4s
✓	Checkout · succeeded	9s
✓	Replace solution code files · succeeded	1s
✓	PowerApps Tool Installer · succeeded	23s
✓	PowerApps Pack Solution · succeeded	3s
✓	PowerApps Pack Solution (Unmanaged) · succeeded	1s
✓	PowerApps Import Solution · succeeded	59s
✓	Build test solution · succeeded 1 warning	23s
✓	Visual Studio Test Platform Installer · succeeded	8s
✓	Run the tests · succeeded	2m 10s
⌚	Publish Pipeline Artifact (solution_managed.zip) · skipped ⓘ	
⌚	Publish Pipeline Artifact (solution.zip) · skipped ⓘ	
✓	Post-job: Checkout · succeeded	<1s
✓	Finalize Job · succeeded	<1s

Merge Feature1 changes into Master

- `git checkout Feature1`
- `git pull origin Feature1`
- `git checkout master`
- `git merge -X theirs Feature1`
- `git add .`
- `git commit -m "..."`
- `git push origin master`

Run Build and Test pipeline

- Once there is a commit on the master branch, the test starts automatically (no need to start it manually)
- Build artefacts are created as a result



✓ #20190925.5: merged
Rolling build triggered today at 11:15 pm for Alex Shlega @CSDemo master 481ecd5

Logs Summary Tests

Job

Pool: Hosted VS2017 · Agent: Hosted Agent

- ✓ Prepare job · succeeded
- ✓ Initialize job · succeeded
- ✓ Checkout · succeeded
- ✓ Replace solution code files · succeeded
- ✓ PowerApps Tool Installer · succeeded
- ✓ PowerApps Pack Solution · succeeded
- ✓ PowerApps Pack Solution (Unmanaged) · succeeded
- ✓ PowerApps Import Solution · succeeded
- ✓ Build test solution · succeeded 1 warning
- ✓ Visual Studio Test Platform Installer · succeeded
- ✓ Run the tests · succeeded
- ✓ Publish Pipeline Artifact (solution_managed.zip) · succeeded
- ✓ Publish Pipeline Artifact (solution.zip) · succeeded

What have we achieved?

- Started with creating a branch and corresponding dev/test environments
- Used a pipeline to deploy unmanaged solution to the dev environment from the source control
- Made a configuration change
- Updated the tests
- Used a pipeline to export solution from CDS and put it in the repo
- Used a pipeline to re-test managed solution in the branch test environment
- Merged all those changes into master
- Build and test pipeline started automatically on the master branch
- As a result, we have everything in the source control, and we have a managed solution for release (as an artifact on the pipeline)

Known limitations of the build tools

- 2 minutes timeout
- No dedicated “solution upgrade” task

Other notes

- It's a relatively involved process
- Requires understanding of git & devops

Q & A