**Power Apps Component Framework (PCF)**

**(all experiments below tried on Power Platform Trial environment “Test” in sintranetdemo tenant, environment is now expired and removed)**

## To allow PCF code controls in Power Apps, perform these 2 actions:

Ref: <https://d365demystified.com/2021/04/13/enable-custom-code-components-pcf-controls-to-be-imported-in-a-canvas-power-app-quick-tip/>

1. Enable setting on **Environement** level:

Admin Center >> Select Environement >> Settings >> Product >> Features >> "Power Apps component framework for canvas apps"

1. Enable setting on **Power App** level:

Select Power App >> Edit App >> Settings >> Upcoming features >> Preview >> Components

## Steps to create PCF component:

Ref: <https://docs.microsoft.com/en-us/powerapps/developer/component-framework/implementing-controls-using-typescript>

**Prerequisites:**

* VS Code
* Node.js
* [Microsoft Power Platform CLI](https://docs.microsoft.com/en-us/powerapps/developer/data-platform/powerapps-cli#install-power-apps-cli) (Use either the Visual Studio Code extension or the MSI installer)
  + [Visual Studio 2019 for Windows & Mac](https://visualstudio.microsoft.com/downloads/). Select at minimum the workload .NET build tools.
  + [Build Tools for Visual Studio 2019](https://visualstudio.microsoft.com/downloads/#build-tools-for-visual-studio-2019). Select at minimum the workload .NET build tools.
* **To deploy your code component using Microsoft Power Platform CLI, you must have a Microsoft Dataverse environment with system administrator or system customizer privileges.**

**Steps**

1. Open VS Code
2. Open new terminal
3. Run commands:
   1. mkdir LinearInput
   2. cd LinearInput
   3. pac pcf init --namespace SampleNamespace --name LinearInputControl --template field

(Update code changes as per instructions in MS reference article)

* 1. npm run build
  2. npm start watch
  3. mkdir MyPCFSolutions
  4. cd MyPCFSolutions
  5. pac solution init --publisher-name MyPCFSamples --publisher-prefix pcfctrls
  6. pac solution add-reference --path ..\

(Open Visual Studio – Continue without code >> Tools >> Command Line >> Developer PowerShell)

* 1. navigate to MyPCFSolutions folder in Visual Studio – Developer PowerShell
  2. msbuild /t:restore
  3. msbuild

1. Upload solution zip file (from debug folder) to Power Platform environment
2. Use code component in Power App (Click on “+” icon >> Get more components >> Code tab >> Select “LinearInputControl” >> Import >> Add to app >> Publish app and test)
3. For updating existing code component, update “version” number in control manifest. Then build code and solution and import solution again and publish all customizations. Close app before updating code component in app. Updating code component dialog will appear while opening app where code component is already added and then it will automatically update code component.
4. <https://docs.microsoft.com/en-us/powerapps/developer/component-framework/component-framework-for-canvas-apps#add-components-to-a-canvas-app>

## Adding code components in model-driven apps

To add a code component like a linear input component, follow the steps mentioned in the article [Add components to columns and tables](https://docs.microsoft.com/en-us/powerapps/developer/component-framework/add-custom-controls-to-a-field-or-entity).

## Adding code components to a canvas app

To add the code components to a canvas app, follow the steps in the article [Add code components to a canvas app](https://docs.microsoft.com/en-us/powerapps/developer/component-framework/component-framework-for-canvas-apps#add-components-to-a-canvas-app).

## Adding code components to a portal

To add the code component to a portal, follow the steps in the article [Use code components in portals](https://docs.microsoft.com/en-us/powerapps/maker/portals/component-framework-tutorial).

**Git Code Repository:**

<https://github.com/shrirampophali/MyPowerAppsPCFControls/tree/master>

**Examples:**

1. **PCF User contact control => Tried, but not possible as cannot get user AD information to canvas or model driven or portal app.**

Limitation is that only this object is available which has very limited properties and are not from AD but from User Dataverse Table:

**context.userSettings**

Property Mapping from User Dataverse Table:

* 1. context.userSettings.userId => systemuserid
  2. context.userSettings.userName=> fullname

1. **Custom Linear Number Control to Custom Number Column in Model Driven App (using Classic Form Editing):**

(Also canvas app option can be used for entity field)

Graphical user interface, application

Description automatically generated

1. Custom Canvas App can be shown as Page in Model Driven App like this (where **User()** function is used in canvas app to show user image and fullname)

Graphical user interface, application

Description automatically generated

1. Embed Canvas App to Model driven app field (binding to field is not mandatory in this case, only viewing) article [here](https://docs.microsoft.com/en-us/powerapps/maker/model-driven-apps/embedded-canvas-app-add-classic-designer)

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Note: Also tried using **Classic** mode, but canvas form is not correctly visible for any field. It keeps loading always and never loads.

Text

Description automatically generated

**Dt: 04-Jan-2022**

Custom PCF Control

“Custom PCF Text Formatter”

<https://docs.microsoft.com/en-us/powerapps/developer/component-framework/manifest-schema-reference/property>

<https://www.w3schools.com/tags/tag_input.asp>

1. Run commands:
   1. mkdir CustomPCFInputTextFormatter
   2. cd CustomPCFInputTextFormatter
   3. pac pcf init --namespace CustomPCFInputTextFormatterNamespace --name CustomPCFInputTextFormatterControl --template field

(Update code changes as per instructions in MS reference article)

* 1. npm run build
  2. npm start watch
  3. mkdir CustomPCFInputTextFormatterSolution
  4. cd CustomPCFInputTextFormatterSolution
  5. pac solution init --publisher-name ShriramPophali --publisher-prefix pcfctrls
  6. pac solution add-reference --path ..\

(Open Visual Studio – Continue without code >> Tools >> Command Line >> Developer PowerShell)

* 1. navigate to CustomPCFInputTextFormatterSolution folder in Visual Studio – Developer PowerShell
  2. msbuild /t:restore
  3. msbuild

1. Upload solution zip file (from debug folder) to Power Platform environment and publish all customizations in solution
2. Use code component in Power App (Click on “+” icon >> Get more components >> Code tab >> Select “LinearInputControl” >> Import >> Add to app >> Publish app and test)
3. For updating existing code component, update “version” number in control manifest. Then build code with “npm run build” and build solution in (Visual Studio 2019 – Developer Powershell), run “msbuild /t:restore” and then “msbuild” and import solution again and publish all customizations. Close app before updating code component in app. Updating code component dialog will appear while opening app where code component is already added and then it will automatically update code component.

Usage of the JavaScript eval function should be limited where possible. The eval function can be a dangerous function as it allows strings to be executed as scripts within the context of the caller. This can be exploited to run malicious code. Eval is also usually slower than other options due to the lack of optimizations of the script text passed to eval. If this error is reported for a Power Apps component framework code component created using CLI tooling, package your control with **'msbuild /p:configuration=Release' or 'npm run build -- --buildMode production'** to produce a release build that does not include 'eval' usage.

1. <https://docs.microsoft.com/en-us/powerapps/developer/component-framework/component-framework-for-canvas-apps#add-components-to-a-canvas-app>

## Adding code components in model-driven apps

To add a code component like a linear input component, follow the steps mentioned in the article [Add components to columns and tables](https://docs.microsoft.com/en-us/powerapps/developer/component-framework/add-custom-controls-to-a-field-or-entity).

## Adding code components to a canvas app

To add the code components to a canvas app, follow the steps in the article [Add code components to a canvas app](https://docs.microsoft.com/en-us/powerapps/developer/component-framework/component-framework-for-canvas-apps#add-components-to-a-canvas-app).

## Adding code components to a portal

To add the code component to a portal, follow the steps in the article [Use code components in portals](https://docs.microsoft.com/en-us/powerapps/maker/portals/component-framework-tutorial).