**Power Apps Component Framework (PCF)**

## 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.

**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.

## 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).

**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):**
2. Graphical user interface, application

   Description automatically generated